AMS-511 Foundations of Quantitative Finance

Fall 2020 — Lecture 05 — 2020-09-30 Wednesday

Introduction to Derivatives
Binomial Lattice Models of Asset Dynamics
Binomial Pricing of European Options
Introduction to Stochastic Calculus and Itô's Lemma

Robert J. Frey, Research Professor
Stony Brook University, Applied Mathematics and Statistics

Robert.Frey@StonyBrook.edu
http://www.ams.sunysb.edu/~frey

Arbitrage and Linear Pricing

Type A Arbitrage and Linear Pricing

Type A arbitrage is said to exist if an investment pays an immediate payoff with no future payoff. Given two securities with prices p_1 and p_2, The price of a portfolio of the two securities must be $p_1 + p_2$. If it were not then it would be possible, then it would be possible to buy (sell) the portfolio, break it up into its constituent assets and sell (buy) then individually, realizing an immediate riskless profit. It is hard to imagine a liquid and transparent market allowing such a situation to exist because the prices would rapidly adjust to make such a payoff impossible.

Type B Arbitrage

A related form of arbitrage is *type B arbitrage* in which an investor pays nothing or a negative amount up front and has the prospect of receiving something in the future. This is equivalent to receiving a free lottery ticket.

Introduction to Derivatives

Derivative Securities

Derivative: A financial contract is a *derivative security* , or a *contingent claim*, if its value at expiration is exactly determined by the market price of one or more cash instruments called the *underlying*.

Notation

T	=	expiration date, the expiry, of the derivative
t	=	time index (typically at some time $t < T$)
$S(t)$	=	the price of the underlying at time t
$F(S(t), t)$	=	the derivative price at time t given $S(t)$
$F(t)$	=	$F(S(t), t)$, when the context is clear

Underlying Assets

Mathematica's FinancialData[ ] function provides a uniform interface for a wide variety of data sources. However, in cases where FinancialData[ ] does not provide the necessary interface, *Mathematica*'s Import[ ] function is a powerful tool for "scraping" data off of web sites. Using Import[ ] for web scraping requires a bit of trial and error and the solution breaks easily if the page format changes, but it is nevertheless a powerful tool for getting data.

Stocks

Stocks or *equities* are claims against the real returns of companies. For example, here are some basic data on Microsoft:

```
In[•]:= FinancialData["MSFT", #] & /@ {"Name", "OHLCV", "MarketCap"}
Out[•]= {Microsoft,
  {$138.97, $140.01, $136.26, $136.37, 27 935 270 shares}, $1.04124 × 10^12}
```

Currencies

The *currencies* of different nations are typically "priced" in terms of exchange rates. For example, here is the Euro/US Dollar exchange rate; *i.e.*, the number of €s it takes to "buy" a $.

```
In[•]:= FinancialData["EUR/USD"]
Out[•]= 1.1123

In[•]:= FinancialData["JPY/EUR"]
Out[•]= 0.00828981

In[•]:= FinancialData["GBP/USD"]
Out[•]= 1.28727
```

Interest Rates

Interest rates are based on some notional asset, e.g., a Treasury Bill, Note or Bond. For example, here is a table of the US Treasury yield curves for the current month scraped off the US Treasury web site:

```
In[•]:= TableForm[Import[
          "https://www.treasury.gov/resource-center/data-chart-center/interest-rates/
            Pages/TextView.aspx?data=yield", {"Data", {2, 1, 2}}][[1, 1, 2]]]
```

Out[•]//TableForm=

Date	1 mo	2 mo	3 mo	6 mo	1 yr	2 yr	3 yr	5 yr	7 yr	10 y
09/01/20	0.09	0.11	0.12	0.13	0.12	0.13	0.14	0.26	0.46	0.6
09/02/20	0.1	0.1	0.12	0.12	0.13	0.14	0.16	0.26	0.45	0.6
09/03/20	0.1	0.11	0.11	0.12	0.12	0.13	0.15	0.24	0.43	0.6
09/04/20	0.09	0.1	0.11	0.12	0.13	0.14	0.18	0.3	0.5	0.7
09/08/20	0.1	0.1	0.13	0.14	0.15	0.14	0.17	0.28	0.47	0.6
09/09/20	0.1	0.11	0.12	0.14	0.14	0.14	0.17	0.28	0.48	0.7
09/10/20	0.1	0.11	0.12	0.12	0.15	0.14	0.17	0.26	0.46	0.6

Commodities

Commodities are usually physical goods such as metals (gold, silver), agricultural goods (pork bellies, soybeans) or energy products (oil, natural gas). For example, here is the gold price scraped off the Markets Insider web site:

```
In[•]:= Import["https://markets.businessinsider.com/commodities/gold-price",
          {"Data", 2, 2, 2, 1, 1}]
Out[•]= 1 Troy Ounce ≈ 1,097 Ounce
```

Market Indices

Market indices are computed from the prices of a basket of securities. For example, here is the S&P 500 stock index:

```
In[•]:= FinancialData["^GSPC"]
Out[•]= 3340.97

In[•]:= FinancialData["^DJI"]
Out[•]= 27 665.6
```

Types of Markets

Cash and Carry

In a cash and carry market traders can borrow at the risk-free rate to buy and then store and insure until expiration.

A buy-and-hold strategy is an alternative to a forward or futures contract; thus, anticipated future demand is reflected in the price and the spread between the forward or future depends only on holding costs.

Price Discovery

In a price discovery market it is impossible to store the underlying because it is perishable or does not yet exist, e.g., spring wheat.

Thus, the current "market" price must be "discovered" via the pricing of the derivative.

Arbitrage & Pricing

Arbitrage is taking simultaneous positions in different assets so that one generates a riskless profit higher than the risk-free rate.

If arbitrage opportunities exist, then this drives traders to sell over-valued assets, decreasing their price, and buy under-valued assets, increasing their price. This continues until an equilibrium is reached in which no further arbitrage is possible.

Arbitrage Free Pricing - We say an asset is fairly or correctly priced if there are no arbitrage opportunities with respect to it, *i.e.*, when its price reflects a market equilibrium.

This is a reasonable model but it is only approximately true. However, in liquid markets with freely available information it is difficult for prices to drift too far from equilibrium.

Forward and Futures Contracts

Forward and Futures Contracts

Forward Contract

A *forward contract* is an obligation to buy or sell an underlying asset at a specified forward price on a known future date T. Forwards are traded over-the-counter (OTC).

Futures Contract

A *futures contract* is a forward that is traded on an organized exchange. They involve standardized contract prices and expiration dates and are marked-to-market daily.

Pricing a Forward in a Cash-and-Carry Market

Consider someone at time t who needs one unit of a commodity at time T where $t < T$. This can be accomplished two ways: Either buy now at price $S(t)$ for consumption at T and forego the interest on the cash used in the purchase. Or buy a forward at $F(t)$ for delivery at T.

Thus, in the absence of other effects these two situations produce identical results. If $r(t, T)$ is the interest rate charged from t to T, then

$$S(t) + r(t, T) S(t) = F(t)$$

If this were not so then an arbitrage involving selling (buying) the underlying while buying (selling) the forward would exist.

If there are additional costs $c(t, T)$, such as storage and insurance, or benefits $b(t, T)$, such as dividend payments, associated with the underlying then

$$S(t) + r(t, T) S(t) - b(t, T) + c(t, T) = F(t)$$

$$S(t)(1 + r(t, T)) - b(t, T) + c(t, T) = F(t)$$

Note that the difference between S(t) and F(t), called the spread, is not related to the uncertainty of the future price of the underlying, but the costs and benefits of a buy-and-hold strategy.

Example: A Commodities Contract

Suppose the spot price of gold is $360/oz., the yearly interest rate is 10% and there are no storage and insurance costs or any benefits from holding gold. The fair value of a future for delivery one year hence is

$$F(0) = S(0) + r(0, 1) S(0) = 360 + 0.10 \times 360 = \$396/\text{oz}.$$

Options

```
In[•]:= FinancialData["IBM", "OHLCV"]
```

```
Out[•]= { $132.55 , $134.05 , $131.61 , $133.96 , 4 195 803 shares }
```

European Options

A *European-style option* on a security is the right (not obligation) to buy in a call option or to sell in a put option the security at strike price K at expiry T.

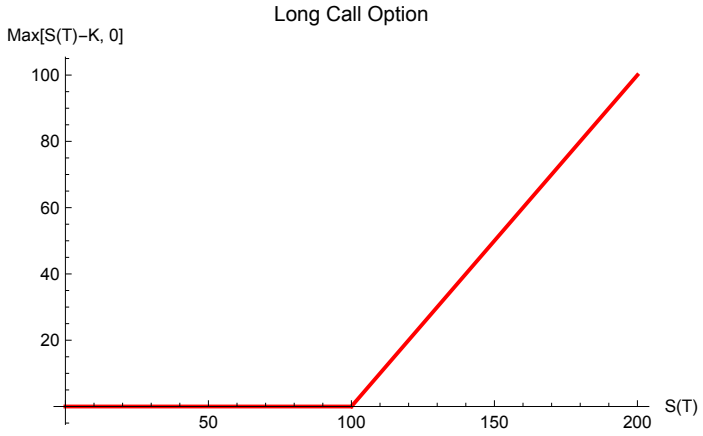
The fact that an option involves a right means that the option only has value at expiration when there is some economic benefit associated with its exercise. Thus, the relationship between $S(T)$ and $F(T)$ is non-linear.

Pay-Off Diagrams at Expiration (t = T)

Consider a case in which a trader is long--has bought--a call with strike $K = 100$, then $F(T) = \max[S(T) - K, 0]$.

In[]:=
```
Plot[Max[s - 100, 0], {s, 0, 200},
 PlotStyle → {Red, Thick},
 PlotLabel → "Long Call Option",
 AxesLabel → {"S(T)", "Max[S(T)-K, 0]"}
]
```

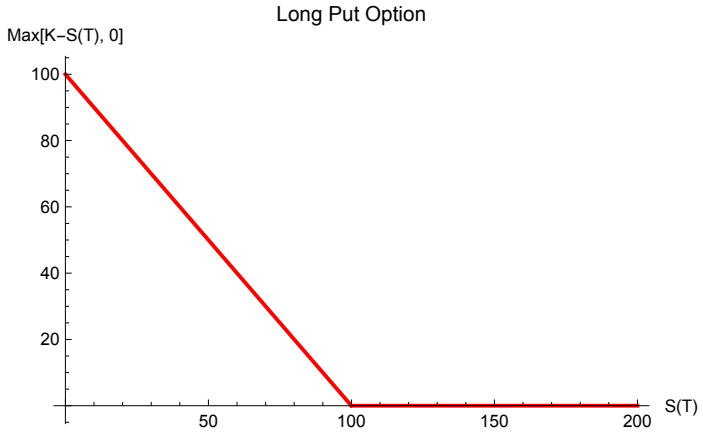
Out[]=



Consider a case in which a trader is long--or has bought--a put with strike $K = 100$, then $F(T) = \max[K - S(T), 0]$.

In[]:=
```
Plot[Max[100 - s, 0], {s, 0, 200},
 PlotStyle → {Red, Thick},
 PlotLabel → "Long Put Option",
 AxesLabel → {"S(T)", "Max[K-S(T), 0]"}
]
```

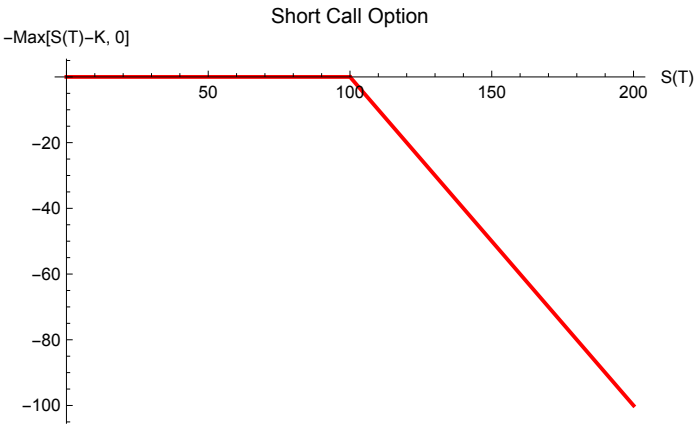
Out[]=



Consider a case in which a trader is short--has sold or written--a call with strike $K = 100$, then $F(T) = -\max[S(T) - K, 0]$.

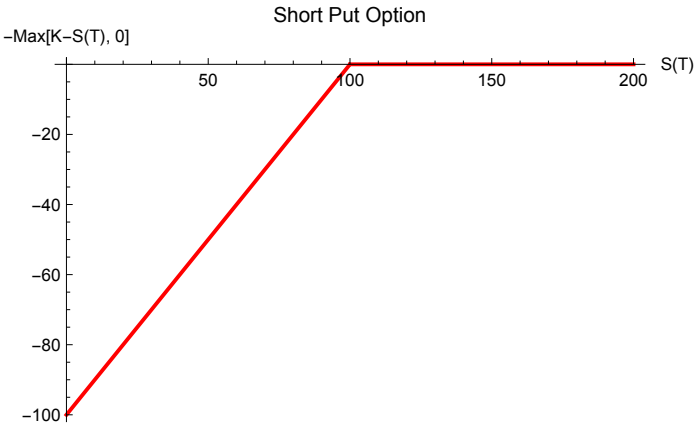
In[•]:=
```
Plot[-Max[s - 100, 0], {s, 0, 200},
 PlotStyle → {Red, Thick},
 PlotLabel → "Short Call Option",
 AxesLabel → {"S(T)", "-Max[S(T)-K, 0]"}
]
```

Out[•]=



Consider a case in which a trader is long--or has sold or written--a put with strike $K = 100$, then $F(T) = -\max[K - S(T), 0]$

```
Plot[-Max[100 - s, 0], {s, 0, 200},
 PlotStyle → {Red, Thick},
 PlotLabel → "Short Put Option",
 AxesLabel → {"S(T)", "-Max[K-S(T), 0]"}
]
```



Option Valuation Prior to Expiration

An option is said to be *out-of-the money* if

Put: $S(t) > K$

Call:$S(t) < K$

An option is said to be *at-the-money* if $S(t) = K$.

An option is said to be *in-the-money* if

Put: $S(t) < K$

Call:$S(t) > K$

Prior to expiration even an out-of-the-money option has a positive premium because there is a chance that the price will drift past the strike in the time remaining to expiration.

Swaps & Swaptions

A *swap* is the simultaneous selling and purchasing of cash flows involving various currencies, interest rates and a number of other financial assets.

A typical example is an interest rate swap, say, to swap a variable rate with a fixed rate. An interest rate swap could be used to effectively convert an adjustable rate loan into a fixed rate loan without renegotiating the loan or refinancing the debt.

A *swaption* is a contract that gives the owner the right, but not the obligation, to enter into or exit out of a swap.

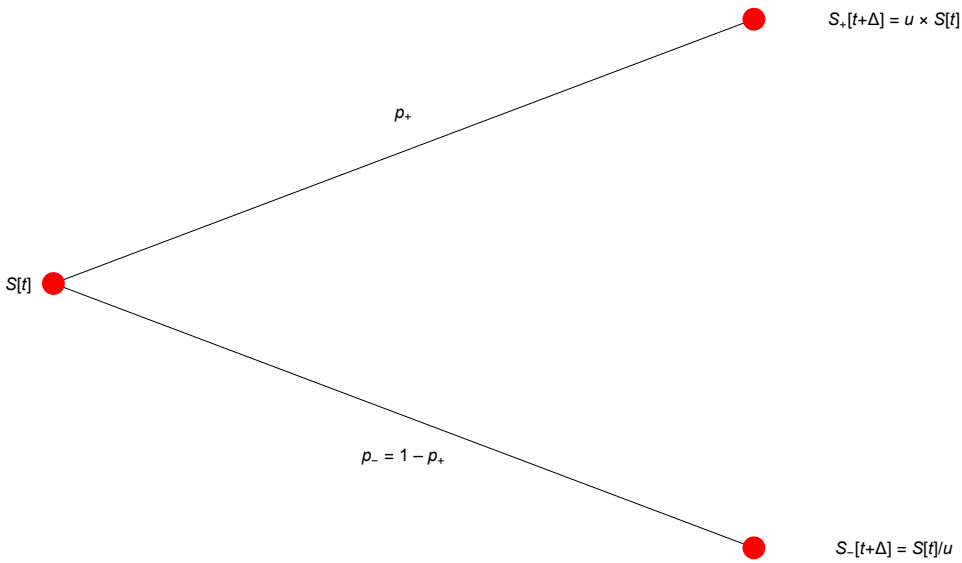
Returning to interest rates. A swaption could be used to construct a situation in which a loan and swaption operates like a variable rate loan until the variable rate exceeds some trigger. The swaption is exercised which kicks in a swap that swaps out the variable rate for a fixed rate. This strategy produces the equivalent of an adjustable rate loan with a rate ceiling.

Binomial Lattice Models

Geometric Binomial Lattices

A useful special case of a lattice is the *binomial lattice*. At each step one of the branches of a node combines with one of the branches of another. Thus, instead of 2^n leaf nodes, there are only $n + 1$ leaves.

Let $S(t)$ denote the price of the asset at time t. At time $t + \Delta$ with probability p_+ the asset will be $u \times S[t]$ and with probability $p_- = 1 - p_+$ the asset will be $S[t]/u$.



Following this model, after n up steps and m down steps the price will be $u^{n-m}\, S[t]$ regardless of the specific sequence of up and down steps it took to get there. Thus, over multiple time steps the process forms a recombinant structure called a lattice.

Example: Stock Price Model

Consider the price model above in which $S(0) = 1, \Delta = 0.1, p_+ = 0.5$, and $u_\Delta = 1.1$. We can plot a full ten steps on graph with time along the x-axis and price along the y-axis.

TRepresenting a lattice as a sequence of vectors, this function takes a lattice x and adds one step using the parameters $p_+, \Delta,$ and u:

```
In[•]:= xAddGeoStep[x_, {p_, Δ_, u_}] := Append[x, Append[# u, Last[#] / u] &[Last[x]]];
```

With $S(0) = 1$, represented by the root {{1}}, a single step is then

```
In[•]:= xAddGeoStep[{{1}}, {0.5, 0.01, 1.1}]
Out[•]= {{1}, {1.1, 0.909091}}
```

A lattice with two steps is the recursion

```
In[•]:= xAddGeoStep[xAddGeoStep[{{1}}, {0.5, 0.01, 1.1}], {0.5, 0.01, 1.1}]
Out[•]= {{1}, {1.1, 0.909091}, {1.21, 1., 0.826446}}
```

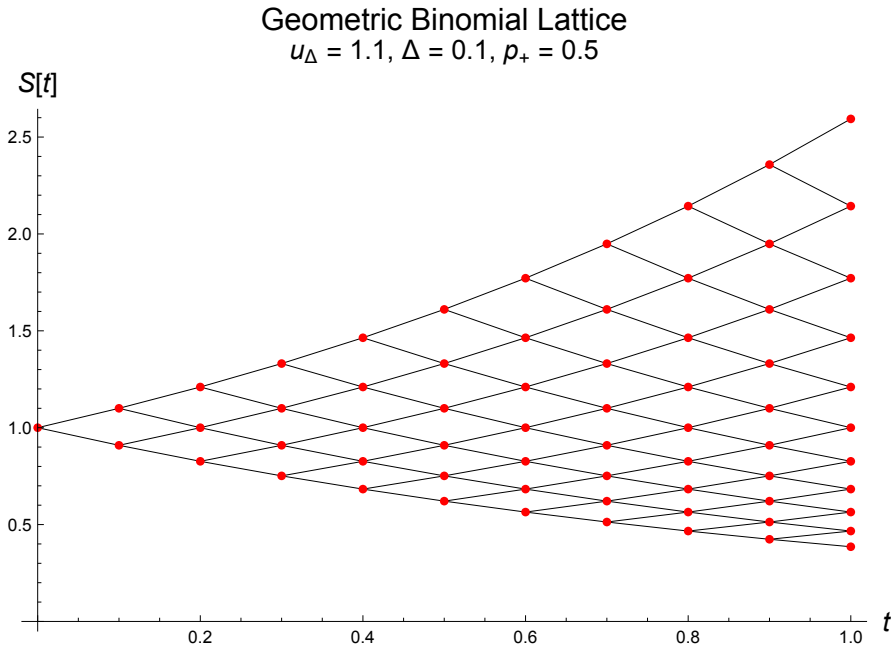
And over ten steps we can use Nest to recursively apply the function "xAddGeoStep[#, {0.5, 0.01, 1.1}] &". We also Map (/@) MatrixForm to provide an easier to read output.

```
In[•]:= MatrixForm /@ Nest[xAddGeoStep[#, {0.5, 0.01, 1.1}] &, {{1}}, 10]
```

$$\left\{(1),\begin{pmatrix}1.1\\0.909091\end{pmatrix},\begin{pmatrix}1.21\\1.\\0.826446\end{pmatrix},\begin{pmatrix}1.331\\1.1\\0.909091\\0.751315\end{pmatrix},\begin{pmatrix}1.4641\\1.21\\1.\\0.826446\\0.683013\end{pmatrix},\begin{pmatrix}1.61051\\1.331\\1.1\\0.909091\\0.751315\\0.620921\end{pmatrix},\right.$$

$$\begin{pmatrix}1.77156\\1.4641\\1.21\\1.\\0.826446\\0.683013\\0.564474\end{pmatrix},\begin{pmatrix}1.94872\\1.61051\\1.331\\1.1\\0.909091\\0.751315\\0.620921\\0.513158\end{pmatrix},\begin{pmatrix}2.14359\\1.77156\\1.4641\\1.21\\1.\\0.826446\\0.683013\\0.564474\\0.466507\end{pmatrix},\begin{pmatrix}2.35795\\1.94872\\1.61051\\1.331\\1.1\\0.909091\\0.751315\\0.620921\\0.513158\\0.424098\end{pmatrix},\begin{pmatrix}2.59374\\2.14359\\1.77156\\1.4641\\1.21\\1.\\0.826446\\0.683013\\0.564474\\0.466507\\0.385543\end{pmatrix}\Bigg\}$$

A plot of this lattice is

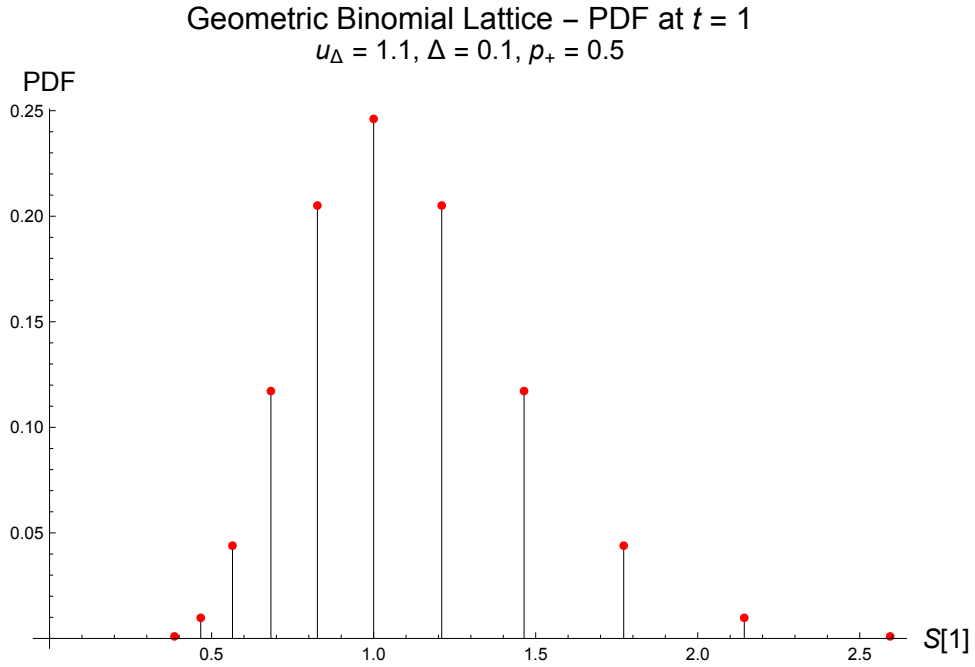


Limiting Distribution

Consider a geometric binomial lattice as described above with n steps and each step being of duration $\Delta = 1/n$. If we number the nodes from top to bottom in step $n > 0$ from 0 to n, then the probability that node i in step n is reached is determined by the binomial distribution:

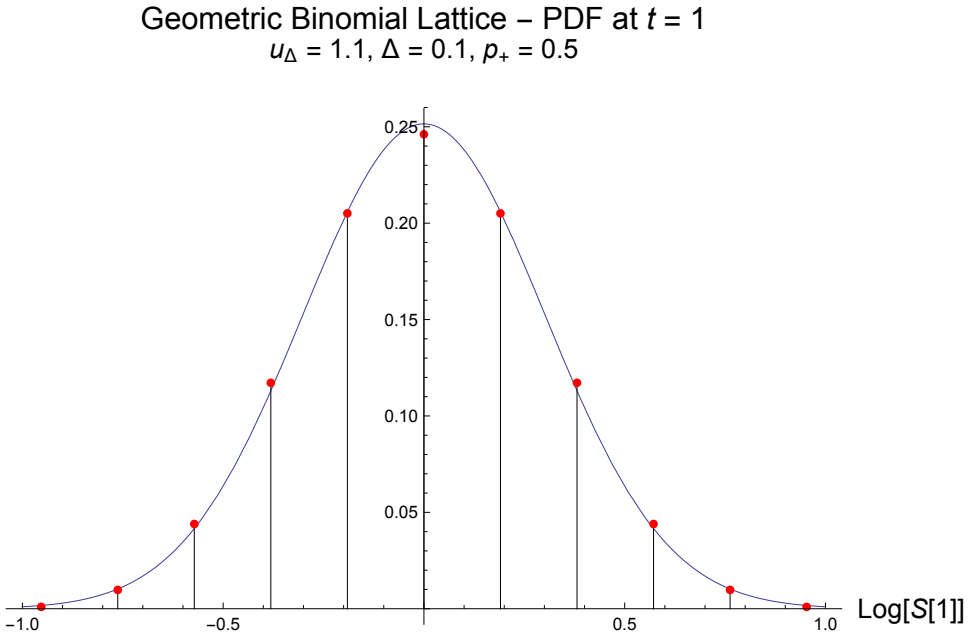
$$\mathbb{P}_n[i] = \binom{n}{i} p^i (1-p)^{n-i}, \quad i \in \{0, \ldots, n\}$$

A plot of the pdf at the leaf nodes for $p = 0.5$, $u = 1.1$ and $t = 1$ (*i.e.*, $n = 10$) is:



If we take the log of the x-values we get a the usual binomial distribution, something that looks very much like a

discrete approximation to a Normal distribution. The plot below has the distribution of log price with a Normal distribution overlaid on it.



As the number of steps $n \to \infty$ or the size of the time step $\Delta \to 0$, the distribution of log prices asymptotically approaches a Normal distribution. When log[X] is Normally distributed, then X is said to be *log Normally distributed*.

In[•]:=
```
PDF[LogNormalDistribution[μ, σ], r]
```

Out[•]=
$$\begin{cases} \frac{e^{-\frac{(-\mu+\text{Log}[r])^2}{2\sigma^2}}}{\sqrt{2\pi}\, r\, \sigma} & r > 0 \\ 0 & \text{True} \end{cases}$$

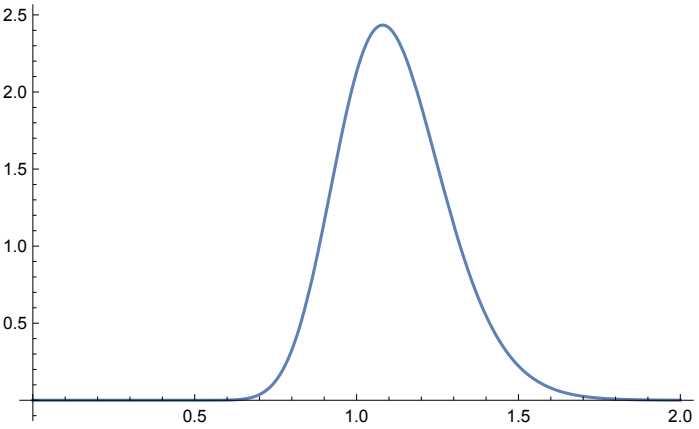
In[•]:=
```
TableForm[Through[
   {Mean, StandardDeviation, Skewness, Kurtosis}[LogNormalDistribution[μ, σ]]],
 TableHeadings → {{"mean", "sdev", "skew", "kurt"}}]
```

Out[•]//TableForm=

mean	$e^{\mu+\frac{\sigma^2}{2}}$
sdev	$\sqrt{e^{2\mu+\sigma^2}\left(-1+e^{\sigma^2}\right)}$
skew	$\sqrt{-1+e^{\sigma^2}}\left(2+e^{\sigma^2}\right)$
kurt	$-3+3\,e^{2\sigma^2}+2\,e^{3\sigma^2}+e^{4\sigma^2}$

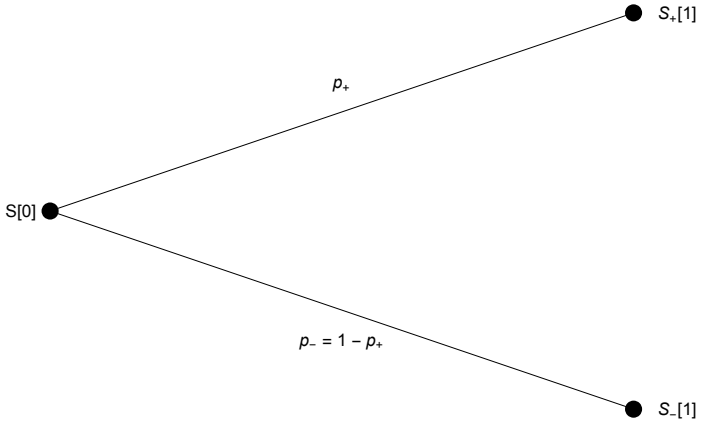
In[]:=
```
Plot[PDF[LogNormalDistribution[0.1, 0.15], r], {r, 0, 2}]
```

Out[]=



Finite State Models

We'll work through a simple binomial case here with $p_+,\ p_- > 0$.



In the absence of arbitrage and in the presence of a risk-free asset we have

$$\begin{pmatrix} 1 \\ S[0] \end{pmatrix} = \begin{pmatrix} 1+r_f & 1+r_f \\ S_+[1] & S_-[1] \end{pmatrix} \begin{pmatrix} \psi_+ \\ \psi_- \end{pmatrix}, \quad \begin{pmatrix} \psi_+ \\ \psi_- \end{pmatrix} > \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

In[]:=
```
Solve[{1., 100.} == {{1.02, 1.02}, {105., 97.}}.{ψ1, ψ2}, {ψ1, ψ2}]
```

Out[]=
```
{{ψ1 → 0.612745, ψ2 → 0.367647}}
```

The state prices and risk neutral measure are:

In[]:=
```
{nPsi1, nPsi2} = {ψ1, ψ2} /. First[%]
```

Out[]=
```
{0.612745, 0.367647}
```

In[]:=
```
{nQ1, nQ2} = {nPsi1, nPsi2} / Total[{nPsi1, nPsi2}]
```

Out[]=
```
{0.625, 0.375}
```

Under the risk neutral measure the current price is the discounted expected value of the future price.

$$S(0) = (1 + r_f)^{-1}\, \mathbb{E}_Q[S(1)]$$

```
In[•]:= 1/1.02 {nQ1, nQ2}.{105., 97.}
Out[•]= 100.
```

Note that $S_+[1]/S[0] > 1 + r_f > S_-[1]/S[0]$ implies that $\psi_+,\ \psi_- > 0$. If that were not the case, then arbitrage opportunities would exist.

For example, if $S_+[1]/S[0] > S_-[1]/S[0] > (1 + r_f)$, then it would be possible to borrow the amount $S[0]$ of cash and buy one unit of the asset. After one period the asset is guaranteed to make more than $(1 + r_f)$; hence, it will be possible to sell the asset, repay the borrowed cash and reap a guaranteed profit.

On the other hand, if $(1 + r_f) > S_+[1]/S[0] > S_-[1]/S[0]$, then it would be possible to place the amount $S[0]$ of cash into a savings account and short one unit of the asset. After one period the asset is guaranteed to make less than $(1 + r_f)$; hence, it will be possible to unwind the short by buying a share in the market using savings. The cash in the savings account will be greater than $S[1]$ and, hence, a guaranteed profit results.

The Binomial Option Pricing Model

General Single Step Solution

The geometric binomial model has many advantages. First, over a reasonable number of steps it represents a surprisingly realistic model of price dynamics. Second, the state price equations at each step can be expressed in a form independent of $S(t)$ and those equations are simple enough to solve in closed form.

$$\begin{pmatrix} 1 \\ S(t) \end{pmatrix} = \begin{pmatrix} e^{r\,\Delta} & e^{r\,\Delta} \\ S_+(t+\Delta) & S_-(t+\Delta) \end{pmatrix} \begin{pmatrix} \psi_+ \\ \psi_- \end{pmatrix} = \begin{pmatrix} e^{r\,\Delta} & e^{r\,\Delta} \\ u_\Delta\, S(t) & S(t)/u_\Delta \end{pmatrix} \begin{pmatrix} \psi_+ \\ \psi_- \end{pmatrix}$$

If we divide the bottom row by $S(t)$, then we see that the state prices for a geometric binomial lattice are the same throughout.

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} e^{r\,\Delta} & e^{r\,\Delta} \\ u_\Delta & 1/u_\Delta \end{pmatrix} \begin{pmatrix} \psi_+ \\ \psi_- \end{pmatrix}$$

As we will see shortly, we can solve the general problem by solving a sequence of single-step problems on the lattice. That sequence of solutions can be efficiently computed because we only have to solve for the state prices once.

Valuing an Option with One Period to Expiration

Let the current value of a stock be $S(t) = 105$ and let there be a call option with unknown price $C(t)$ on the stock with a strike price of 100 that expires the next three month period. The *annual* risk free rate is $r = 0.04$. We'll use a single binomial step, so $\Delta = 0.25$ years. For the period of t to $t + \Delta$, $u_\Delta = 110\%$; the binomial probability is $p_+ = 0.50$. In the equations below, known quantities are in blue and quantities to be solved for are in red.

$$\begin{pmatrix} 1 \\ S(t) \\ C\,(t) \end{pmatrix} = \begin{pmatrix} e^{r\,\Delta} & e^{r\,\Delta} \\ u\, S(t) & S(t)/u \\ \max[u\, S(t) - K, 0] & \max[S(t)/u - K, 0] \end{pmatrix} \begin{pmatrix} \psi_+ \\ \psi_- \end{pmatrix}$$

$$\Rightarrow \quad \begin{pmatrix} 1 \\ 105 \\ C\,(t) \end{pmatrix} \simeq \begin{pmatrix} 1.01 & 1.01 \\ 115.50 & 95.46 \\ 16.40 & 0 \end{pmatrix} \begin{pmatrix} \psi_+ \\ \psi_- \end{pmatrix}$$

Note that each column in the matrix above represents a different possible future state of nature and those states collectively are all inclusive.

Let

- $\mathbf{m}$ = current prices of cash and the asset
- $\mathbf{M}$ = forward state price matrix of cash and the asset
- $\mathbf{c}$ = forward state price vector of the option
- ψ = state price vector
- v = current option price

These equations can be summarized as

$$\begin{pmatrix} \mathbf{m} \\ v \end{pmatrix} = \begin{pmatrix} \mathbf{M} \\ \mathbf{c}^T \end{pmatrix} \psi$$

The values of the state prices are determined by the first two rows. The state prices can then be used to calculate the option's price based on its two outcomes.

$$\psi = \mathbf{M}^{-1}\,\mathbf{m} \quad \Rightarrow \quad v = \mathbf{c}^T\,\psi \quad \Rightarrow \quad v = \mathbf{c}^T\,M^{-1}\,\mathbf{m}$$

We can solve the first two equations for the state prices. Note that the quarterly rate is $e^{0.25\times 0.04}$

```
In[•]:= ψ = Inverse[{{e^0.01, e^0.01}, {105 × 1.10, 105 / 1.10}}].{1, 105}
Out[•]= {0.523572, 0.466478}
```

If we take these values and substitute them into the last row we realize a call premium of $C(t) \simeq 8.11$:

```
In[•]:= {Max[1.1 × 105 - 100, 0], Max[105 / 1.1 - 100, 0]}.ψ
Out[•]= 8.11537
```

The risk neutral measure Q is:

```
In[•]:= q = ψ e^0.01
Out[•]= {0.528834, 0.471166}

In[•]:= Total[q]
Out[•]= 1.
```

It is remarkable but, other than meeting the requirement that $0 < p_+ < 1$, the ordinary probability measure does not play a role in the solution.

Replicating Portfolio

An alternate approach to pricing the option, one that is in a sense dual to that of risk neutral pricing, is to build a replicating portfolio. Under the Law of One Price, if the pay-offs of the replicating portfolio are identical to those of the option, then the current price of the replicating portfolio and the option must also be identical.

Let the current value of a stock be $S(t) = 105$ and let there be a call option with unknown price $C(t)$ on the stock with a strike price of 100 that expires the next three-month period. The annual risk free rate is $r = 0.04$. We'll use a single binomial step, so $\Delta = 0.25$ years. For the period of t to $t + \Delta$, $u = 110\times\%$; the binomial probability is $p_+ = 0.50$.

As before, in the equations below, known quantities are in blue and quantities to be solved for are in red.

Let x_B represent the cash position (The "B" is the "bank" balance.), and x_S represents the underlying position. We seek a replicating portfolio of cash and underlying $\mathbf{x} = (x_B, x_S)^T$ such that

$$\begin{pmatrix} \max[u\,S(t) - K, 0] \\ \max[S(t)/u - K, 0] \end{pmatrix} = \begin{pmatrix} e^{r\,\Delta} & u\,S(t) \\ e^{r\,\Delta} & S(t)/u \end{pmatrix} \begin{pmatrix} x_B \\ x_S \end{pmatrix} \quad \Rightarrow \quad C(t) = \begin{pmatrix} 1 \\ S(t) \end{pmatrix}^T \begin{pmatrix} x_B \\ x_S \end{pmatrix}$$

In terms of the linear algebra for the risk neutral approach above, we see that the result of determining the replicating portfolio ends up with the same relationship as that of the risk neutral solution. Note the matrix above is the transpose of the state-price matrix used in the risk neutral solution. Thus, using the same notation

$$\mathbf{c} = \mathbf{M}^T \mathbf{x} \quad \& \quad v = \mathbf{m}^T \mathbf{x} \quad \Rightarrow \quad v = \mathbf{c}^T \mathbf{M}^{-1} \mathbf{m}$$

$$\psi = \mathbf{M}^{-1} \mathbf{m} \quad \Rightarrow \quad v = \mathbf{c}^T \psi \quad \Rightarrow \quad v = \mathbf{c}^T M^{-1} \mathbf{m}$$

We can solve for the replicating portfolio:

```
In[•]:= x = Inverse[Transpose[{{e^0.01, e^0.01}, {1.1 × 105, 105 / 1.1}}]].
          {Max[1.1 × 105 - 100, 0], Max[105 / 1.1 - 100, 0]}
Out[•]= {-73.0751, 0.773243}
```

Then use the replicating portfolio to price the option:

```
In[•]:= {1, 105}.x
Out[•]= 8.11537
```

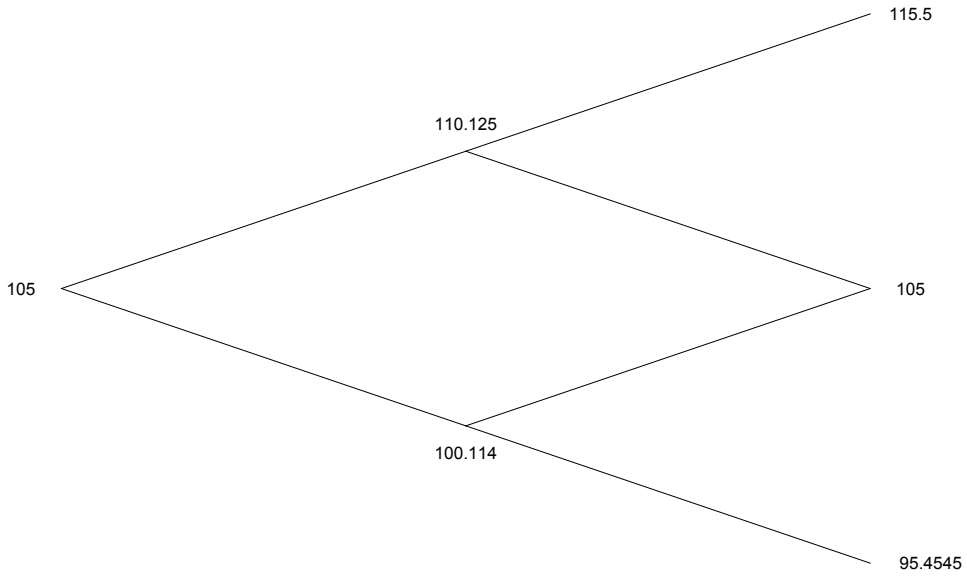
and get the same result as the risk neutral pricing.

The General Binomial Option Pricing Model

We'll return to the example above but now assume that the option expires is 3 months, but we wish to model the stock price evolution using two lattice steps;*i.e.*, $\Delta = 0.125$. The binomial lattice for the stock is then

```
In[•]:= MatrixForm /@ Nest[xAddGeoStep[#, {0.5, 0.125, √1.10}] &, {{105}}, 2]
Out[•]= {(105), (110.125
                 100.114), (115.5
                            105.
                            95.4545)}
```

Note that we have adjusted $u_{0.125} = \sqrt{u_{0.25}}$ so that its product over the interval is the same as the single step case.

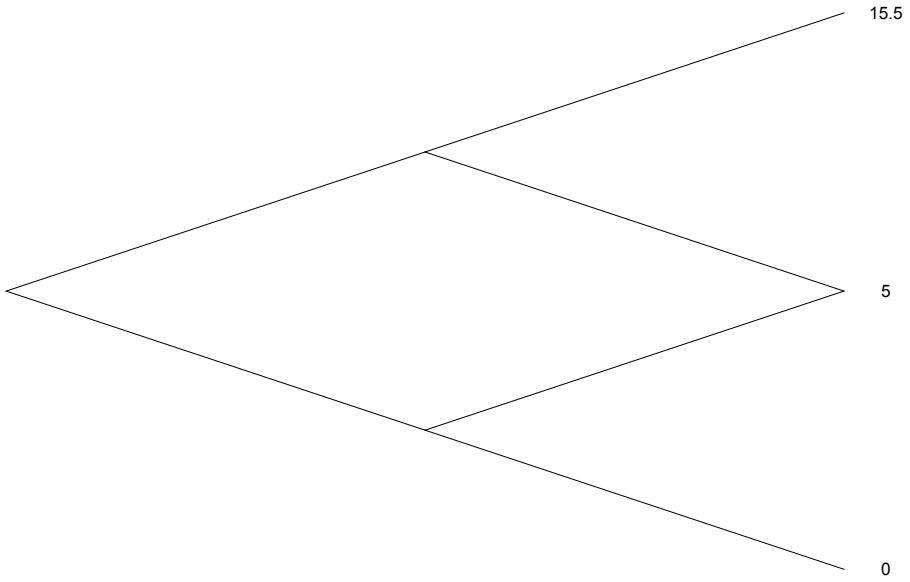



```
In[•]:= ψ = Inverse[{{e^0.005, e^0.005}, {√1.1, 1/√1.1}}].{1, 1}
Out[•]= {0.537964, 0.457049}
In[•]:= q = ψ e^0.005
Out[•]= {0.54066, 0.45934}
```

The risk neutral measure is

$$\begin{pmatrix} q_+ \\ q_- \end{pmatrix} = \begin{pmatrix} 0.541 \\ 0.459 \end{pmatrix}$$

The "leaves" of the lattice above occur at expiry, so for each price node we can calculate the value of the call option. This gives us a binomial pricing lattice for the option:



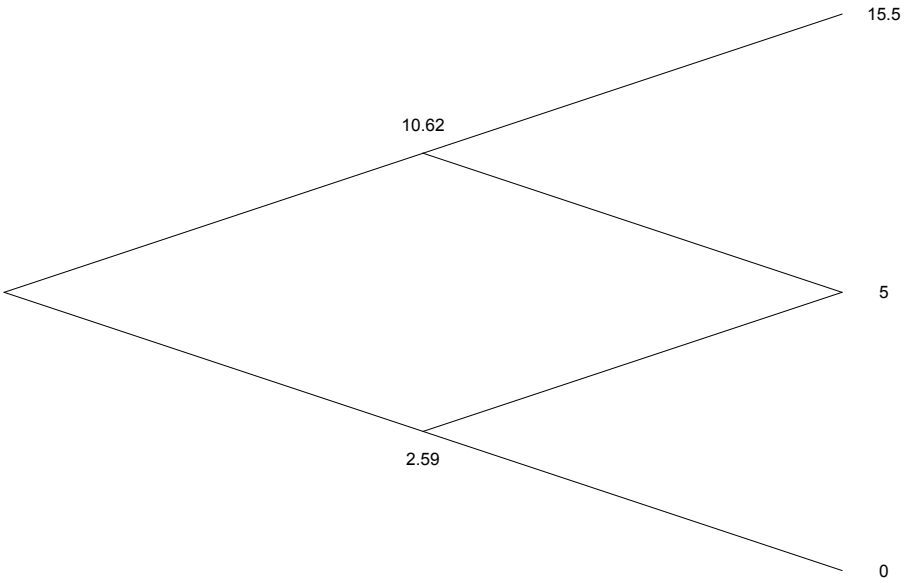
We now apply this to the next step back.

```
In[•]:= q.{15.5, 5} / 1.005
        q.{5, 0} / 1.005
Out[•]= 10.6238
Out[•]= 2.68985
```

$$e^{r_f \Delta}\begin{pmatrix} q_+ \\ q_- \end{pmatrix}^T \begin{pmatrix} 15.5 \\ 5.00 \end{pmatrix} = e^{-0.005}\begin{pmatrix} 0.541 \\ 0.459 \end{pmatrix}^T \begin{pmatrix} 15.5 \\ 5.00 \end{pmatrix} = 10.62$$

$$e^{r_f \Delta}\begin{pmatrix} q_+ \\ q_- \end{pmatrix}^T \begin{pmatrix} 5.00 \\ 0.00 \end{pmatrix} = e^{-0.005}\begin{pmatrix} 0.541 \\ 0.459 \end{pmatrix}^T \begin{pmatrix} 5.00 \\ 0.00 \end{pmatrix} = 2.69$$

We can now add this information to the lattice.

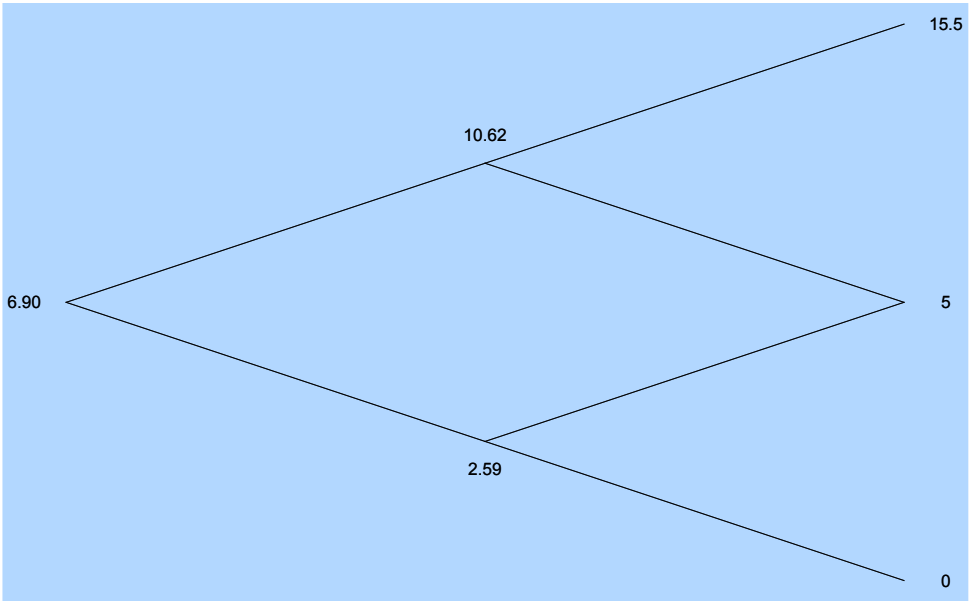


Finally, we add this to complete the lattice for the option.

```
In[•]:= e^-0.005 q.{10.62, 2.59}
Out[•]= 6.89693
```

$$e^{r_f \Delta}\begin{pmatrix} q_+ \\ q_- \end{pmatrix}^T \begin{pmatrix} 10.62 \\ 2.59 \end{pmatrix} = e^{-0.005}\begin{pmatrix} 0.541 \\ 0.459 \end{pmatrix}^T \begin{pmatrix} 10.62 \\ 2.59 \end{pmatrix} = 6.90$$

This allows us to complete the lattice. The value at the “root”, 6.90, is the value of the call option.



Calibrating the Model to Market Data

For a given step-size Δ we have two free parameters in the geometric binomial model: p_+ and u. As was shown earlier, with a sufficient number of steps the log price tends towards a Normal distribution. The calibration of the geometric binomial model at step-size Δ, therefore, involves selecting a set of parameters p_+ and u so that the mean and standard deviation of the log price are matched.

$$\mu = \mathrm{E}\left[\log\left[\frac{S(t+1)}{S(t)}\right]\right]$$

$$\sigma^2 = \mathrm{Var}\left[\log\left[\frac{S(t+1)}{S(t)}\right]\right]$$

$$p_+ = \frac{1}{2}\left(1 + \frac{\mu}{\sigma}\sqrt{\Delta}\right)$$

$$u = e^{\sigma\sqrt{\Delta}}$$

Mathematica Code

Code

This code is not the most efficient; it's meant to illustrate the computations in the simplest terms. Each step or level in the lattice is represented by a list. The entire lattice is a list of such lists, e.g., {{x11},. {x21, x22}, {x31, x32, x33}, ...}.

In[•]:=
```
xLatticeForwardStep[vnOneLevel_, nUpFactor_] :=
  Append[vnOneLevel nUpFactor, Last[vnOneLevel] / nUpFactor];

xRiskNeutralMeasure[nRiskFree_, ΔTime_, nUpFactor_] :=
```

$$\frac{\text{Exp[nRiskFree}\,\Delta\text{Time]} - 1 / \text{nUpFactor}}{\text{nUpFactor} - 1 / \text{nUpFactor}};$$

```
xCallExercise[nNowPrice_, nStrikePrice_] := Max[nNowPrice - nStrikePrice, 0];

xPutExercise[nNowPrice_, nStrikePrice_] := Max[nStrikePrice - nNowPrice, 0];

xLatticeBackStep[vnOneLevel_, nRiskFree_, ΔTime_, nRiskNeutralProb_] :=
  Exp[-nRiskFree ΔTime]
   (nRiskNeutralProb Most[vnOneLevel] + (1 - nRiskNeutralProb) Rest[vnOneLevel]);

xEuroCallOptGeoBin[nNowPrice_, nVolatility_,
   nStrikePrice_, nExpiry_, nRiskFree_, iIntervals_] := Module[
   {vvnBackwardTree, vnExpiryPrice, vvnForwardTree,
    nRiskNeutralProb, nUpFactor, ΔTime},
   ΔTime = nExpiry / iIntervals;
   nUpFactor = Exp[√ΔTime nVolatility];
   vvnForwardTree = NestList[
     xLatticeForwardStep[#, nUpFactor] &,
     {nNowPrice},
     iIntervals
    ];
   nRiskNeutralProb = xRiskNeutralMeasure[nRiskFree, ΔTime, nUpFactor];
   vnExpiryPrice = xCallExercise[#, nStrikePrice] & /@ Last[vvnForwardTree];
   vvnBackwardTree = NestList[
     xLatticeBackStep[#, nRiskFree, ΔTime, nRiskNeutralProb] &,
     vnExpiryPrice,
     iIntervals
    ];
   {nRiskNeutralProb, vvnForwardTree, vvnBackwardTree}
  ];

xEuroPutOptGeoBin[nNowPrice_, nVolatility_,
   nStrikePrice_, nExpiry_, nRiskFree_, iIntervals_] := Module[
   {vvnBackwardTree, vnExpiryPrice, vvnForwardTree,
    nRiskNeutralProb, nUpFactor, ΔTime},
   ΔTime = nExpiry / iIntervals;
   nUpFactor = Exp[√ΔTime nVolatility];
   vvnForwardTree = NestList[
     xLatticeForwardStep[#, nUpFactor] &,
     {nNowPrice},
     iIntervals
    ];
```

```
  nRiskNeutralProb = xRiskNeutralMeasure[nRiskFree, ΔTime, nUpFactor];
  vnExpiryPrice = xPutExercise[#, nStrikePrice] & /@ Last[vvnForwardTree];
  vvnBackwardTree = NestList[
    xLatticeBackStep[#, nRiskFree, ΔTime, nRiskNeutralProb] &,
    vnExpiryPrice,
    iIntervals
   ];
  {nRiskNeutralProb, vvnForwardTree, vvnBackwardTree}
 ];
```

Example — Pricing Put and Call Options

Consider a stock with current price $S(0)$=125. It's log mean and variance are $\mu = 0.10$ and σ^2=0.01. There is a put option on the stock with expiry T=0.5 and a strike price $K = 120$. The annual risk free rate is 4%. Estimate the price of the put option using a 5-step geometric binomial option model.

In[•]:=
```
{nRiskNeutral, vvnStockLattice, vvnOptionLattice} =
  xEuroPutOptGeoBin[125, 0.1, 120, 0.5, 0.04, 5];
```

In[•]:=
```
MatrixForm[{#, 1 - #} &[nRiskNeutral]]
MatrixForm /@ vvnStockLattice
MatrixForm /@ vvnOptionLattice
```

Out[•]//MatrixForm=

$$\begin{pmatrix} 0.555457 \\ 0.444543 \end{pmatrix}$$

Out[•]=
$$\left\{(125), \begin{pmatrix} 129.016 \\ 121.109 \end{pmatrix}, \begin{pmatrix} 133.161 \\ 125. \\ 117.339 \end{pmatrix}, \begin{pmatrix} 137.439 \\ 129.016 \\ 121.109 \\ 113.687 \end{pmatrix}, \begin{pmatrix} 141.855 \\ 133.161 \\ 125. \\ 117.339 \\ 110.148 \end{pmatrix}, \begin{pmatrix} 146.412 \\ 137.439 \\ 129.016 \\ 121.109 \\ 113.687 \\ 106.719 \end{pmatrix}\right\}$$

Out[•]=
$$\left\{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 6.31342 \\ 13.2809 \end{pmatrix}, \begin{pmatrix} 0. \\ 0. \\ 0. \\ 2.79538 \\ 9.37321 \end{pmatrix}, \begin{pmatrix} 0. \\ 0. \\ 1.23771 \\ 5.69668 \end{pmatrix}, \begin{pmatrix} 0. \\ 0.548019 \\ 3.20706 \end{pmatrix}, \begin{pmatrix} 0.242645 \\ 1.72317 \end{pmatrix}, (0.897208)\right\}$$

The put is $\simeq 0.90$.

Warning: We're reusing some variables... Estimate the price of the call option using a 5-step geometric binomial option model.

In[•]:=
```
{nRiskNeutral, vvnStockLattice, vvnOptionLattice} =
  xEuroCallOptGeoBin[125, 0.1, 120, 0.5, 0.04, 5];
```

```
In[•]:= MatrixForm[{#, 1 - #} &[nRiskNeutral]]
        MatrixForm /@ vvnStockLattice
        MatrixForm /@ vvnOptionLattice
```

Out[•]//MatrixForm=

$$\begin{pmatrix} 0.555457 \\ 0.444543 \end{pmatrix}$$

Out[•]= $\left\{(125), \begin{pmatrix} 129.016 \\ 121.109 \end{pmatrix}, \begin{pmatrix} 133.161 \\ 125. \\ 117.339 \end{pmatrix}, \begin{pmatrix} 137.439 \\ 129.016 \\ 121.109 \\ 113.687 \end{pmatrix}, \begin{pmatrix} 141.855 \\ 133.161 \\ 125. \\ 117.339 \\ 110.148 \end{pmatrix}, \begin{pmatrix} 146.412 \\ 137.439 \\ 129.016 \\ 121.109 \\ 113.687 \\ 106.719 \end{pmatrix}\right\}$

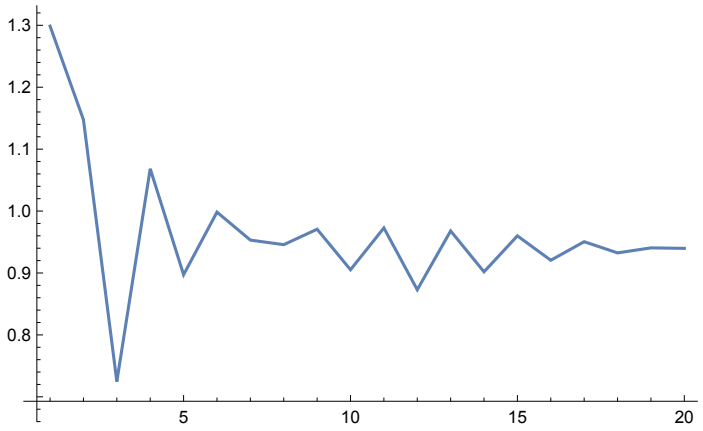
Out[•]= $\left\{\begin{pmatrix} 26.4124 \\ 17.4393 \\ 9.01601 \\ 1.109 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 22.334 \\ 13.6401 \\ 5.47904 \\ 0.613542 \\ 0. \end{pmatrix}, \begin{pmatrix} 18.3954 \\ 9.97218 \\ 3.30288 \\ 0.339435 \end{pmatrix}, \begin{pmatrix} 14.5924 \\ 6.97941 \\ 1.97757 \end{pmatrix}, \begin{pmatrix} 11.1634 \\ 4.73689 \end{pmatrix}, (8.27337)\right\}$

The call is $\simeq 8.27$.

We can see the convergence as the number of intervals increases.

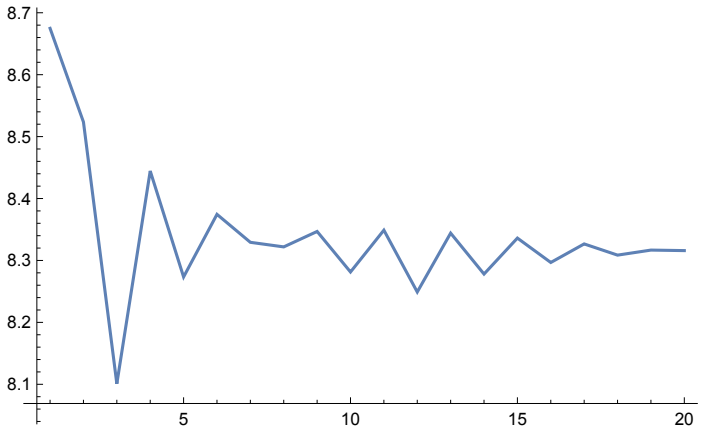
```
In[•]:= ListLinePlot[
         Table[
          {n, Last[xEuroPutOptGeoBin[125, 0.1, 120, 0.5, 0.04, n]][[-1, 1]]},
          {n, 1, 20}
         ],
         PlotRange → All
        ]
```

Out[•]=



In[•]:=
```
ListLinePlot[
 Table[
  {n, Last[xEuroCallOptGeoBin[125, 0.1, 120, 0.5, 0.04, n]][[-1, 1]]},
  {n, 1, 20}
 ],
 PlotRange → All
]
```

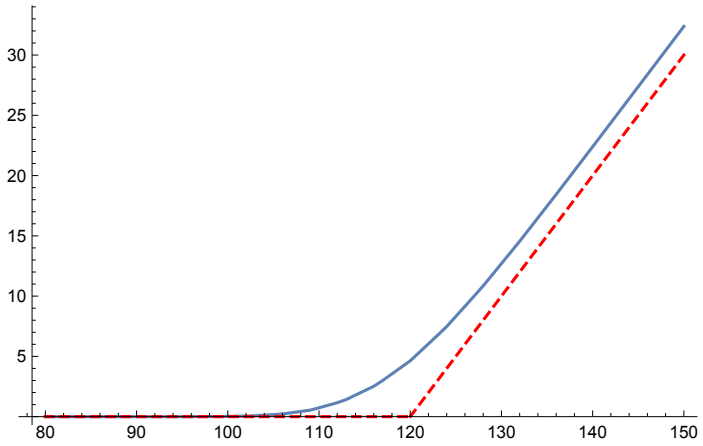
Out[•]=



Time to expiry $(T - t) = 0.5$:

In[•]:=
```
Show[
 ListLinePlot[
  Table[
   {p, Last[xEuroCallOptGeoBin[p, 0.1, 120, 0.5, 0.04, 20]][[-1, 1]]},
   {p, 80, 150}
  ],
  PlotRange → All
 ],
 Plot[Max[p - 120, 0], {p, 80, 150}, PlotStyle → {Red, Dashed}]
]
```

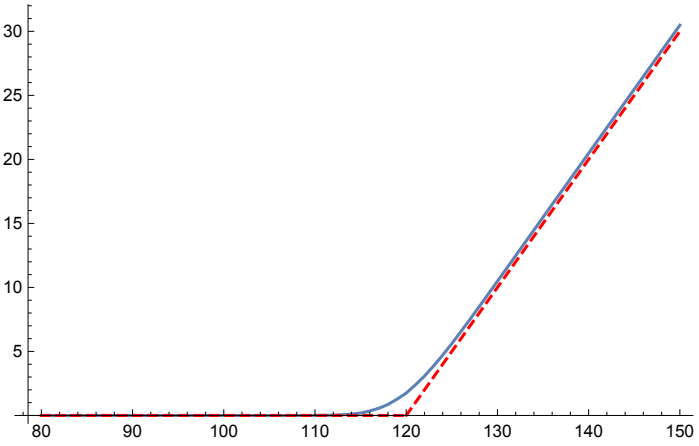
Out[•]=



Time to expiry $(T - t) = 0.1$:

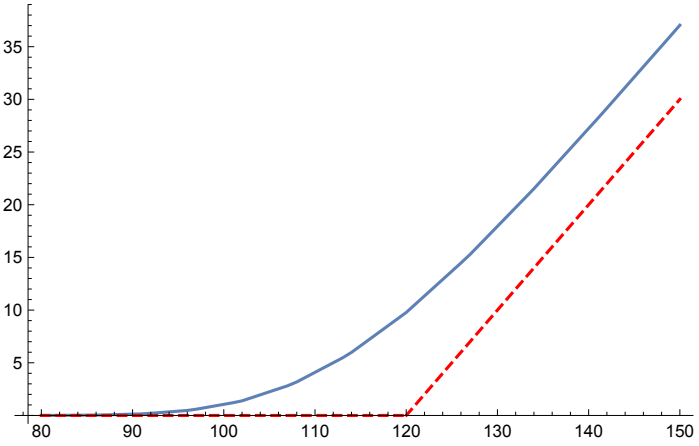
In[•]:=
```
Show[
 ListLinePlot[
  Table[
   {p, Last[xEuroCallOptGeoBin[p, 0.1, 120, 0.1, 0.04, 20]][[-1, 1]]},
   {p, 80, 150}
  ],
  PlotRange → All
 ],
 Plot[Max[p - 120, 0], {p, 80, 150}, PlotStyle → {Red, Dashed}]
]
```

Out[•]=



In[•]:=
```
Show[
 ListLinePlot[
  Table[
   {p, Last[xEuroCallOptGeoBin[p, 0.1, 120, 1.5, 0.04, 20]][[-1, 1]]},
   {p, 80, 150}
  ],
  PlotRange → All
 ],
 Plot[Max[p - 120, 0], {p, 80, 150}, PlotStyle → {Red, Dashed}]
]
```

Out[•]=



Stochastic Calculus

Further Development of the Binomial Model

We'll define a standard binomial process $t \in [0, 1]$ using n steps, each of length $\Delta = 1/n$

$$B(t) = B(t - \Delta) + X(t)\ \sqrt{\Delta}$$

where the $X(t)$ are i.i.d. random variables

$$X(t) = \begin{cases} +1, & \text{with prob. } 1/2 \\ -1, & \text{with prob. } 1/2 \end{cases}$$

$X(t)$ *is not observed* until the end of the interval at t. In other words, there is no lookahead to future random events. Let

$$\Delta B(t) = B(t) - B(t - \Delta) = X(t)\ \sqrt{\Delta}$$

At time $(t - \Delta)$ we have

$$\mathbb{E}[B(t)] = \mathbb{E}[\Delta B\,(t)] + B(t - \Delta)] = B(t - \Delta)$$

Thus, $B(t)$ is a martingale. The $\Delta B(t)$ are called *innovations* because they represent new information not contained in the information available prior to t.

Note that each ΔB is *i.i.d.* with mean zero and variance Δ. Thus, the sum of n such innovations has mean 0 and variance $n\,\Delta = n(1/n) = 1$. By appealing to the Central Limit Theorem we can assert that for sufficiently large n (or

small Δ) and assuming $B(0) = 0$

$$\sum_{i=1}^{n} \Delta B(i\,\Delta) = B(1) \sim N[0,\,1]$$

This process is standardized so that independent of n and Δ it has a mean of zero and standard deviation of one over a unit time interval. As $n \to \infty$, as a consequence of the Central Limit Theorem, it approaches standard Normal distribution over that unit time interval. Note more generally, we have for $B(0) = 0$ and an arbitrary $t > 0$

$$\Delta \to 0, \quad B(t) \sim N\left[0,\,\sqrt{t}\right]$$

Geometric Binomial Process

Consider a price process

$$\Delta\, S(t) = \mu_S\, S(t-\Delta)\,\Delta \;+\; \sigma_S\, S(t-\Delta)\,\Delta \mathrm{B}(t)$$

or alternately as a return process

$$\frac{\Delta\, S(t)}{S(t-\Delta)} = \mu_S\,\Delta \;+\; \sigma_S\,\Delta \mathrm{B}(t)$$

By specifying a mean μ_S and standard deviation σ_S we can describe a geometric diffusion process.

Stochastic Integral

Note that summing the ΔB over the unit interval the result is a random variable.

$$\sum_{i=1}^{n} \Delta B(i\,\Delta) = B(1) \sim N[0,\,1]$$

When $\Delta \to 0$ ($n \to \infty$), we can think of the limit of this process as *an integral whose result is a random variable*. Thus, making the substitution $\Delta B(t) \to dB(t)$ as $\Delta \to 0$

$$\lim_{\Delta \to 0\,(\text{or } n \to \infty)} \sum_{i=1}^{n} \Delta B(i\,\Delta) \;\equiv\; \int_0^1 dB(t) \sim N[0,\,1]$$

And this obviously generalizes to

$$\int_0^t dB(s) \sim N\left[0,\,\sqrt{t}\right]$$

We have made some sense of this integral as the limit of the summation of a discrete process, but it does not quite have the same meaning as does the corresponding deterministic case because the result of the integral is a random variable.

$(dB(t))^2 = dt$

In the deterministic case with a "smooth" function, as the step size $\Delta \to 0$ the higher powers of Δ are dominated by Δ itself. However, in the stochastic case, the limit of the binomial process we examined is not differentiable in the conventional sense: it never "smooths" out.

Now for n steps for t from 0 to 1 of size $\Delta = 1 / n$ we have $\Delta B(\mathrm{t}) = \pm\sqrt{\Delta}$ and, therefore, $(\Delta \mathrm{B}(t))^2 = \Delta$.

This means that as $\Delta \to 0$ we have

$$(\Delta \mathrm{B}(t))^2 = \Delta \quad \Rightarrow \quad (dB(t))^2 = dt$$

and

$$\int_0^1 (dB(t))^2 = \int_0^1 dt = 1$$

and, more generally,

$$\int_0^t (dB(s))^2 = \int_0^t ds = t$$

Standard calculus is based on the notion that as $\Delta \to 0$ the higher order effects become negligible. A stochastic calculus can not make that assumption. For the class of problems we'll be looking at, both $dB(t)$ and $(dB(t))^2$ must be considered in order to capture both the mean and variance effects.

The Wiener Process *W(t)*

Instead of a discrete binomial process, consider a continuous process $W(t)$ such that the differences are i.i.d. random variables such that for $\Delta > 0$

$$W(t) - W(t-\Delta) = \Delta W(t) \sim N\left[0, \sqrt{\Delta}\right]$$

As before, we can divide the interval $t \in [0, 1]$ into n equal steps of size $\Delta = 1 / n$. Clearly,

$$\sum_{i=1}^{n} \Delta W(i\,\Delta) \sim N[0, 1]$$

Unlike the binomial process $B(t)$ which *approaches* a Normally distributed random variable in the limit, the equation above is a simple consequence of the fact that the increments of $W(t)$ are *i.i.d.* Normal random variables whose standard deviations are the square root of the size of the increment. The random process $W(t)$ is a called a *Wiener process*.

As we did with $B(t)$ above, we can define:

$$\lim_{\Delta \to 0 \,(\text{or } n \to \infty)} \sum_{i=1}^{n} \Delta W(i\,\Delta) = \int_0^1 dW(t) \sim N[0, 1]$$

As before, we have for $W(0) = 0$ and an arbitrary $t > 0$

$$W(t) = \int_0^t dW(t) \sim N\left[0, \sqrt{t}\right]$$

$(dW(t))^2 \to dt$ in Mean Square

For the binomial process examined above we determined that $(dB(t))^2 = dt$. We could do this because, for a given step size Δ, the value of $(\Delta B(t))^2 = \Delta$ deterministically.

For a Weiner process the value of $(\Delta W(t))^2$ is itself a random variable; however, given that $\Delta W(t)$ is $N[0, \sqrt{\Delta}]$, we do know that $E[(\Delta W(t))^2]$ (*i.e.*, the variance of $\Delta W(t)$) is Δ. As $\Delta \to 0$ ($n \to \infty$) we gain more and more samples over any finite interval so $\text{Var}\left[E\left[(\Delta W(t))^2\right]\right] \to 0$ over that interval. We say that $(dW(t))^2 \to dt$ in *mean square* as $\Delta \to 0$. In other words, the limit is not deterministic, but we can make the variance arbitrarily small by choosing a sufficiently small interval.

Simulation Studies

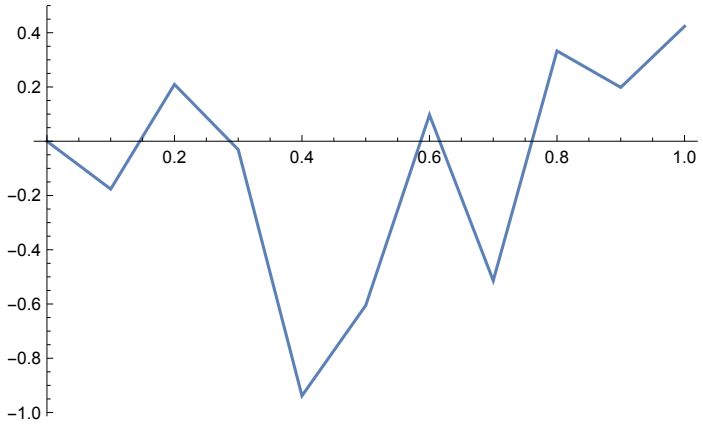
We'll illustrate these notions for the standard Wiener process $W(t)$.

First, consider the sum $\sum_{i=1}^{n} \Delta W(i\,\Delta)$ using $n = 1000$ steps. As expected, $W(1) \sim N[0, 1]$.

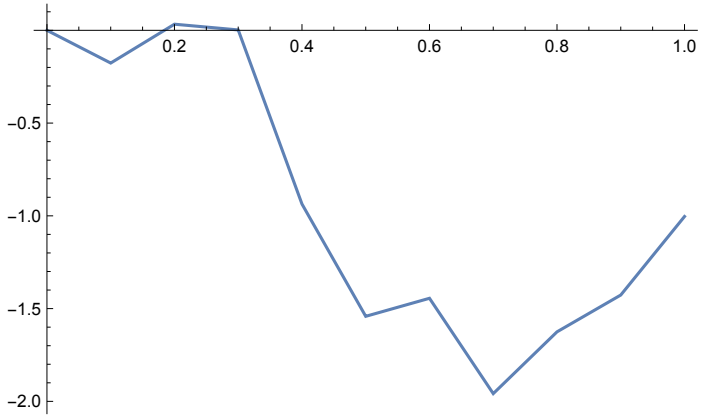
```
In[•]:= xDeltaW[n_, t_] := Join[
          {{0, 0}},
          Table[
           {i t / n, RandomReal[NormalDistribution[0, Sqrt[1. / n]]]},
           {i, 1, n}
          ]
         ];

In[•]:= xDeltaW[3, 1]

Out[•]= {{0, 0}, {1/3, 0.188971}, {2/3, 0.458512}, {1, 0.751913}}

In[•]:= xIntegrateTimeSeries = {First /@ #, Accumulate[Last /@ #]}^T &;

In[•]:= xSquareTimeSeries = {First /@ #, Power[Last /@ #, 2]}^T &;
```

In[•]:=
```
mnΔW = xDeltaW[10, 1];
ListPlot[mnΔW, Joined → True]
ListPlot[xIntegrateTimeSeries[mnΔW], Joined → True]
Show[
 ListPlot[xIntegrateTimeSeries[xSquareTimeSeries[mnΔW]], Joined → True],
 Plot[x, {x, 0, 1}, PlotStyle → Red]
]
```

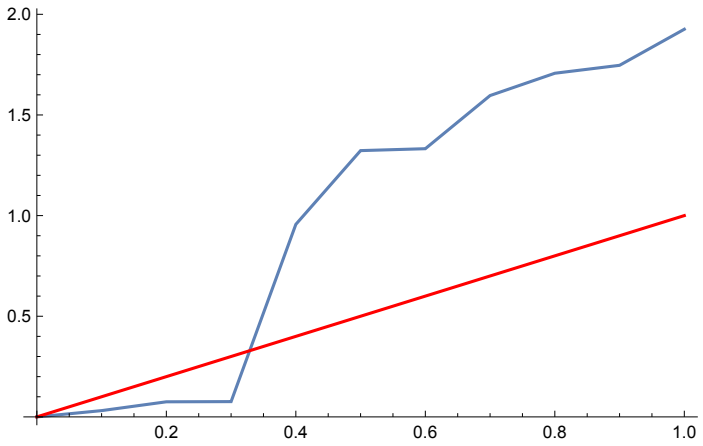
Out[•]=



Out[•]=



Out[•]=

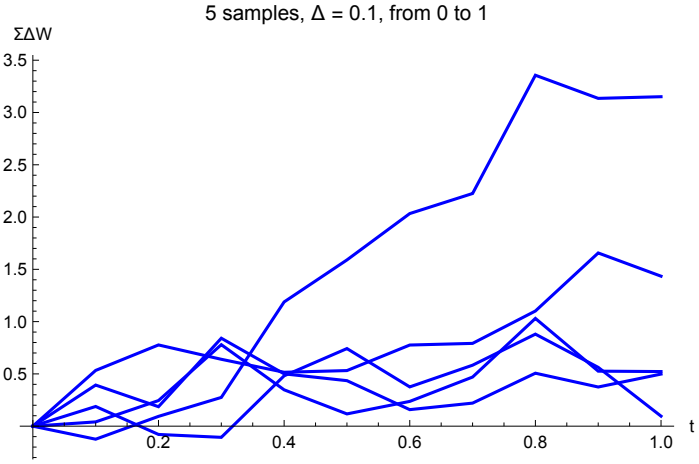


Next, consider the sum $\Delta W(i\,\Delta)$ with $\Delta = 1/n$ using 10, 100, 1000 and 10,000 steps. For each case we'll generate 5 samples of the sum $\sum_{i=1}^{n} \Delta W(i\,\Delta)^m$ for $m = 1$ and 2 over the period $t \in [0, 1]$ for n time steps of $\Delta = 1 / n$.

As the number of steps increases the samples converge more and more closely to *t*. This is what you would expect with mean square convergence: The spread on the variance decreases as the number of samples increases.

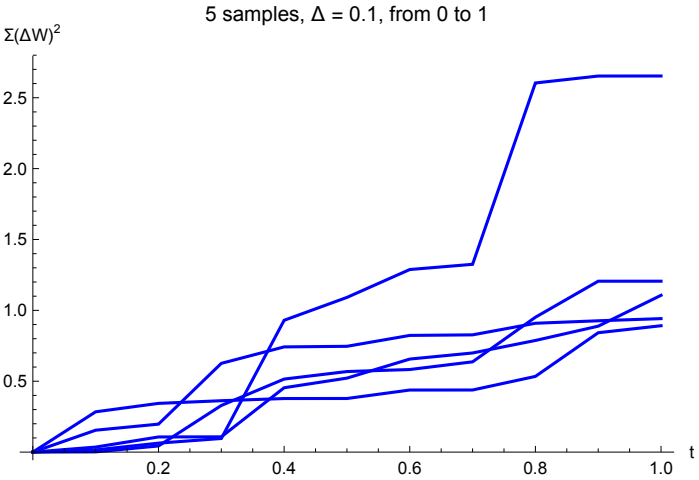
```
In[•]:= xSimWiener[k_, n_, t_] := Module[
          {vmnΔW},
          {
           vmnΔW = Table[xDeltaW[n, t], {k}],
           ListPlot[
            xIntegrateTimeSeries /@ vmnΔW,
            Joined → True,
            PlotStyle → Blue,
            AxesLabel → {"t", "ΣΔW"},
            PlotLabel → ToString[k] <>
              " samples, Δ = " <> ToString[1. / n] <> ", from 0 to " <> ToString[t]
           ],
           ListPlot[
            xIntegrateTimeSeries /@ (xSquareTimeSeries /@ vmnΔW),
            Joined → True,
            PlotStyle → Blue,
            PlotRange → All,
            AxesLabel → {"t", "Σ(ΔW)^2"},
            PlotLabel → ToString[k] <>
              " samples, Δ = " <> ToString[1. / n] <> ", from 0 to " <> ToString[t]
           ]
          }
         ]

In[•]:= r = xSimWiener[5, 10, 1];

In[•]:= r[[2]]
```

Out[•]=



In[•]:=
```
r[[3]]
```

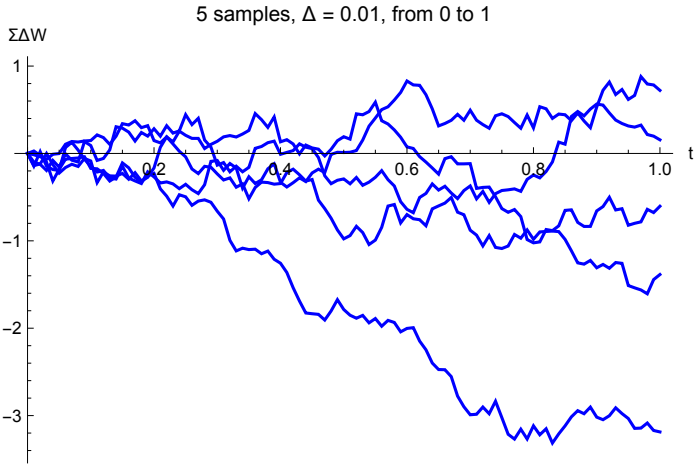
Out[•]=



In[•]:=
```
r = xSimWiener[5, 100, 1];
```

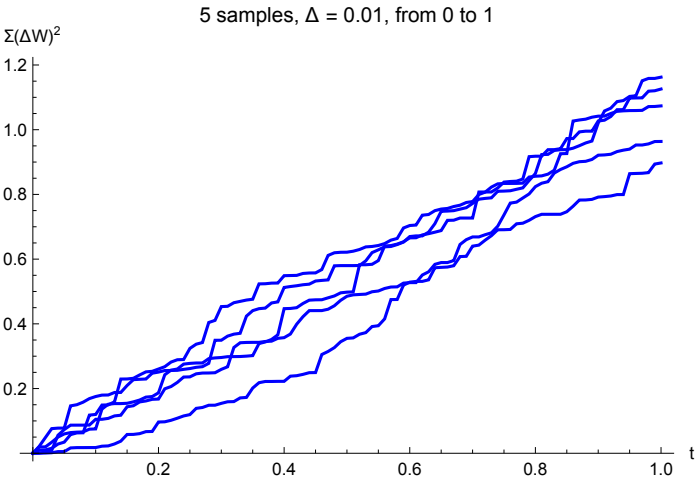
In[•]:=
```
r[[2]]
```

Out[•]=



In[•]:=
```
r[[3]]
```

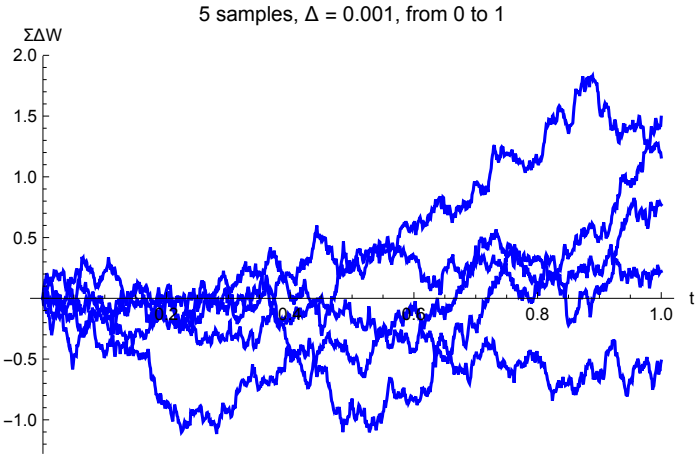
Out[•]=



In[•]:=
```
r = xSimWiener[5, 1000, 1];
```

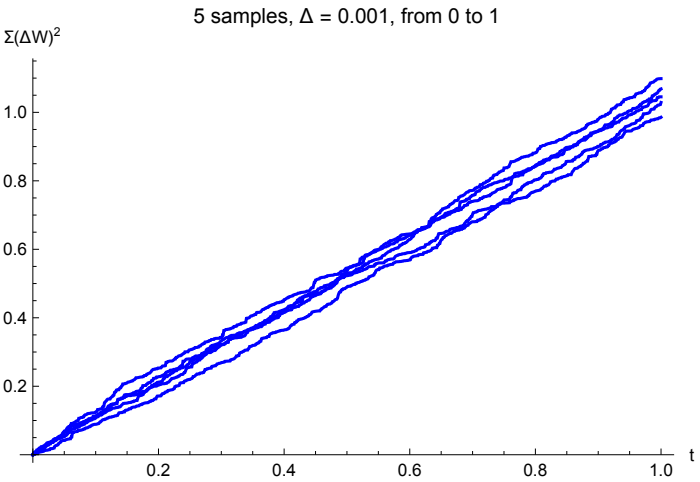
In[•]:=
```
r[[2]]
```

Out[•]=



In[•]:=
```
r[[3]]
```

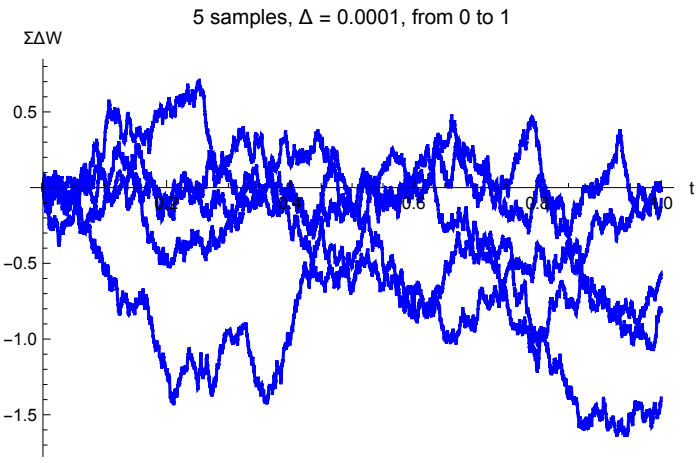
Out[•]=



In[•]:=
```
r = xSimWiener[5, 10 000, 1];
```

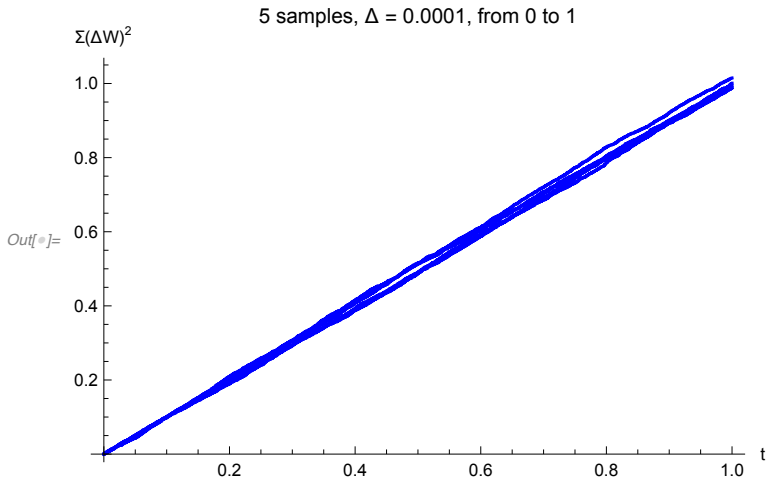
In[•]:=
```
r[[2]]
```

Out[•]=



In[•]:= r[[3]]

Out[•]=



Itô Processes and Itô's Lemma

Consider the following stochastic process, called an *Itô process*

$$d\,X(t) = a[X(t),\ t]\,d\,t + b[X(t),\ t]\,d\,W(t)$$

The $a[X(t),\ t]\,d\,t$ is often called the *drift term* and corresponds to the local mean of the process and the $b[X(t),\ t]\,d\,W(t)$ the *volatility term* and corresponds to the local standard deviation of the process.

Consider the case of simple Brownian motion:

$$d\,X(t) = \mu\,d\,t + \sigma\,d\,W(t)$$

Then, making the simplifying assumption that $X(0) = 0$, we have $X(t) \sim \mathrm{N}[\mu\,t, \sigma\sqrt{t}\ ]$

The Chain Rule

Consider the deterministic case

$$d\,x = a[x, t]\,d\,t$$

and a function $g[x, t]$ which is differentiable with respect to both x and t, then if $y = g[x, t]$ we can write dy as a Taylor series

$$d\,y = \begin{pmatrix} \frac{\partial g}{\partial t} \\ \frac{\partial g}{\partial x} \end{pmatrix}^T \begin{pmatrix} d\,t \\ d\,x \end{pmatrix} + \frac{1}{2}\begin{pmatrix} d\,t \\ d\,x \end{pmatrix}^T \begin{pmatrix} \frac{\partial^2 g}{(\partial t)^2} & \frac{\partial^2 g}{\partial t\,\partial x} \\ \frac{\partial^2 g}{\partial t\,\partial x} & \frac{\partial^2 g}{(\partial x)^2} \end{pmatrix} \begin{pmatrix} d\,t \\ d\,x \end{pmatrix} + \ldots$$

expanding this expression

$$d\,y = \frac{\partial g}{\partial t}\,d\,t + \frac{\partial g}{\partial x}\,d\,x + \frac{1}{2}\left(\frac{\partial^2 g}{(\partial t)^2}\,(d\,t)^2 + 2\,\frac{\partial^2 g}{\partial t\,\partial x}\,d\,t\,d\,x + \frac{\partial^2 g}{(\partial x)^2}\,(d\,x)^2\right) + \ldots$$

The higher than linear order terms (*i.e.*, those in red) in the series become negligible at sufficiently small time scales. We drop those terms and get

$$dy = \frac{\partial g}{\partial t} dt + \frac{\partial g}{\partial x} dx$$

Substituting $dx = a(x, t)\, dt$ into the above gives the familiar chain rule.

$$dy = \left(\frac{\partial g}{\partial t} + \frac{\partial g}{\partial x} a[x, t] \right) dt$$

Itô's Lemma

Consider the following Itô process:

$$dX(t) = a[X(t),\ t]\, dt + b[X(t),\ t]\, dW(t)$$

Let $g(x, t)$ be a function which is at least twice differentiable with respect to x and once differentiable with respect to t, then $Y(t) = g[X(t), t]$ follows an Itô process with the same Weiner process $W(t)$:

$$dY(t) = \left(\frac{\partial g}{\partial X} a + \frac{\partial g}{\partial t} + \frac{1}{2} \frac{\partial^2 g}{\partial X^2} b^2 \right) dt + \frac{\partial g}{\partial X} b\, dW(t)$$

in which the derivatives of g and the coefficients a and b depend on the arguments $(X(t), t)$.

Proof: First, write $dY(t)$ as a Taylor series:

$$dY(t) = \frac{\partial g}{\partial t} dt + \frac{\partial g}{\partial X} dX(t) + \frac{1}{2} \left(\frac{\partial^2 g}{(\partial t)^2} (dt)^2 + 2 \frac{\partial^2 g}{\partial t\, \partial X(t)} dt\, dX(t) + \frac{\partial^2 g}{(\partial X(t))^2} (dX(t))^2 \right) + \ldots$$

The terms in red are those which are negligible and can be dropped; the term in blue is the second order effect from the random process that, in contrast to the deterministic case, cannot be ignored. Thus, discarding the terms that are negligible:

$$dY(t) = \frac{\partial g}{\partial t} dt + \frac{\partial g}{\partial X} dX(t) + \frac{1}{2} \frac{\partial^2 g}{\partial X^2} (dX(t))^2$$

Substituting $dX(t)$ and $(dX(t))^2$ in the above

$$dY(t) = \frac{\partial g}{\partial t} dt + \frac{\partial g}{\partial X} (a\, dt + b\, dW(t)) + \frac{1}{2} \frac{\partial^2 g}{\partial X^2} (a\, dt + b\, dW(t))^2$$

Expanding

$$dY(t) = \frac{\partial g}{\partial t} dt + \frac{\partial g}{\partial X} (a\, dt + b\, dW(t)) + \frac{1}{2} \frac{\partial^2 g}{\partial X^2} \left(a^2 (dt)^2 + 2\, a\, b (dt)\, (dW(t)) + b^2 (dW(t))^2 \right)$$

And, again, discarding terms which are negligible and finally replacing $(dW(t))^2$ with dt yields

$$dY(t) = \frac{\partial g}{\partial t} dt + \frac{\partial g}{\partial X} a\, dt + \frac{\partial g}{\partial X} b\, dW(t) + \frac{1}{2} \frac{\partial^2 g}{\partial X^2} b^2\, dt$$

After some simple rearrangement we have the final form above.

$$dY(t) = \left(\frac{\partial g}{\partial X} a + \frac{\partial g}{\partial t} + \frac{1}{2} \frac{\partial^2 g}{\partial X^2} b^2 \right) dt + \frac{\partial g}{\partial X} b\, dW(t)$$

Strictly speaking, this is not a rigorous proof; however, the approach of expanding the differential as a Taylor series, discarding the terms other than dt, $dW(t)$ and $(dW(t))^2$, and then replacing $(dW(t))^2$with dt is one that works well for many practical problems.

Compare the deterministic and stochastic cases with additional terms in the stochastic case in blue.

$$dy = \left(\frac{\partial g}{\partial t} + \frac{\partial g}{\partial x}\, a\right) dt$$

$$dY(t) = \left(\frac{\partial g}{\partial X}\, a + \frac{\partial g}{\partial t} + \frac{1}{2}\,\frac{\partial^2 g}{\partial X^2}\, b^2\right) dt + \frac{\partial g}{\partial X}\, b\, dW(t)$$

Example

Here is one approach to implementing Itô's lemma in *Mathematica*. The functions must be directly coded to have the head Function. The result is the vector $\{dt, dW(t)\}$.

In[•]:=
```
xIto[a_Function, b_Function, g_Function, s_Symbol, t_Symbol] :=
 {a[s, t] × ∂s g[s, t] + ∂t g[s, t] + 1/2 b[s, t]^2 ∂s,s g[s, t], b[s, t] × ∂s g[s, t]}
```

For the case $dX(t) = \mu\, dt + \sigma\, dW(t)$, $g[x, t] = e^x$, and $Y(t) = g[X(t), t]$:

In[•]:=
```
xIto[Function[{s, t}, μ], Function[{s, t}, σ], Function[{s, t}, Exp[s]], x, t]
```

Out[•]=
$$\left\{e^x\, \mu + \frac{e^x\, \sigma^2}{2},\ e^x\, \sigma\right\}$$

Thus,

$$\left.\begin{array}{l} dX(t) = \mu\, dt + \sigma\, dW(t) \\ Y(t) = g[X(t), t] = e^x \end{array}\right\} \implies dY(t) = \left(\mu + \frac{\sigma^2}{2}\right) e^{X(t)}\, dt + \sigma\, e^{X(t)}\, dW(t)$$

and substituting $e^{X(t)} \to Y(t)$ we end up with the final form in terms of $Y(t)$

$$dY(t) = \left(\mu + \frac{\sigma^2}{2}\right) Y(t)\, dt + \sigma\, Y(t)\, dW(t)$$

A Simple Stock Model

We can apply the above to model *constant coefficient geometric Brownian motion* of a stock price $S(t)$

$$dS(t) = \mu\, S(t)\, dt + \sigma\, S(t)\, dW(t)$$

Sometimes this is expressed in terms of the instantaneous rate of return of the stock. Dividing both sides by $S(t)$

$$\frac{dS(t)}{S(t)} = \mu\, dt + \sigma\, dW(t)$$

Note that our earlier experiments with the geometric binomial lattice (which in the limit is the same at the price model above) it appeared that the log price was Normally distributed. Let S follow the constant-coefficient geometric process above with $S(0) = 1$. Take $g[S(t), t] = \log S(t)$ and apply Itô's lemma then we get

In[•]:=
```
xIto[Function[{s, t}, s μ], Function[{s, t}, s σ], Function[{s, t}, Log[s]], s, t]
```

Out[•]=
$$\left\{\mu - \frac{\sigma^2}{2},\ \sigma\right\}$$

You can also specify the function using "&" notation.

In[•]:=
```
xIto[μ # &, σ # &, Log[#] &, s, t]
```

Out[•]=
$$\left\{\mu - \frac{\sigma^2}{2}, \sigma\right\}$$

Thus, we have

$$d \log S(t) = \left(\mu - \frac{1}{2}\sigma^2\right) dt + \sigma\, dW(t)$$

Thus, $\log S(t)$ is Normally distributed with a mean of $\left(\mu - \frac{1}{2}\sigma^2\right) t$ and a standard deviation of $\sigma\sqrt{t}$. It is also clear that $S(t)$ is log Normally distributed and

$$\log S(t) = \left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma\, W(t) \quad \Longrightarrow \quad S(t) = S(0)\, e^{\left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma\, W(t)}$$