Computational Methods in Finance, Lecture 1, Duality and Dynamic Programming.

Last revised September 2000

Jonathan Goodman *
Courant Institute of Mathematical Sciences, NYU

September 12, 2000

1 Introduction

There are two main ideas in the arbitrage theory of pricing. One is that in complete markets, everyone should agree on a common price – any other price leads to an arbitrage opportunity. The other is that this price is the expected value of the cash flow with respect to some probability model – risk neutral pricing. In the simplest case, this probability model is a discrete Markov chain. This lecture describes how to compute probabilities and expected values for discrete Markov chain models. This is the main computational step in "risk neutral" option pricing.

The methods here compute the expected values by a time marching process that uses the transition matrix. Another evolution process allows us to compute probabilities. These evolution processes are related but not the same. The relation between the forward evolution for probabilities and the backward evolution for expected values is called *duality.* It is similar to the relation between a matrix and its transpose. The transpose of a matrix is sometimes called its dual.

*goodman@cims.nyu.edu, or http://www.math.nyu.edu/faculty/goodman,

The method of risk neutral arbitrage pricing extends to other more technical situations, but the main ideas are clear in the simple context of Markov chains. If the Markov chain model is replaced by a stochastic differential equation model, then the transition matrix is replaced by a partial differential operator – the "generator", and the matrix transpose is replaced by the "dual" of this generator. This is the subject of future lectures.

Many financial instruments allow the holder to make decisions along the way that effect the ultimate value of the instrument. American style options, loans that be repaid early, and convertible bonds are examples. To compute the value of such an instrument, we also seek the optimal decision strategy. *Dynamic programming* is a computational method that computes the value and decision strategy at the same time. It reduces the difficult "multiperiod decision problem" to a sequence of hopefully easier "single period" problems. It works backwards in time much as the expectation method does. The tree method commonly used to value American style stock options is an example of the general dynamic programming method.

2 Markov Chains

(This section assumes familiarity with basic probability theory using mathematicians' terminology. References on this include the probability books by G. C. Rota, W. Feller, Hoel and Stone, and B. V. Gnedenko.)

Many discrete time discrete state space stochastic models are stationary discrete Markov chains. Such a Markov chain is characterized by its state space, $\mathcal{S}$, and its transition matrix, P. We use the following notations:

- x, y, $\ldots$: possible states of the system, elements of $\mathcal{S}$.
- The possible times are $t = 0, 1, 2, \ldots$.
- $X(t)$: the (unknown) state of the system at time t. It is some element of $\mathcal{S}$.
- $u(x,t) = \mathbf{Pr}(X(t) = x)$. These probabilities satisfy an evolution equation moving forward in time. We use similar notation for conditional probabilities, for example, $u(x,t|X(0) = x_0) = \mathbf{Pr}(X(t) = x|X(0) = x_0)$.

- $p(x,y) = \mathbf{Pr}(x \to y) = \mathbf{Pr}(X(t+1) = y | X(t) = x)$. These "transition probabilities" are the elements of the transition matrix, P.

The transition probabilities have the properties:

$$0 \le p(x,y) \le 1 \quad \text{for all } x \in \mathcal{S} \text{ and } y \in \mathcal{S}. \tag{1}$$

and

$$\sum_{y \in \mathcal{S}} p(x,y) = 1 \quad \text{for all } x \in \mathcal{S}. \tag{2}$$

The first is because the $p(x,y)$ are probabilities, the second because the state x must go somewhere, possibly back to x. It is not true that

$$\text{(NOT ALWAYS TRUE)} \qquad \sum_{x \in \mathcal{S}} p(x,y) = 1 \ . \qquad \text{(NOT ALWAYS TRUE)}$$

The Markov property is that knowledge of the state at time t is all the information about the present and past relevant to predicting the future. That is:

$$\begin{aligned} \mathbf{Pr}(X(t+1) = y | X(t) = x_0, X(t-1) = x_1, \ldots) \\ = \mathbf{Pr}(X(t+1) = y | X(t) = x_0) \end{aligned} \tag{3}$$

no matter what extra history information $(X(t-1) = x_1, \ldots)$ we have. This may be thought of as a lack of long term memory. It may also be thought of as a completeness property of the model: the state space is rich enough to characterize the state of the system at time t completely.

To illustrate this point, consider the model

$$Z(t+1) = aZ(t) + bZ(t-1) + \xi(t) \ , \tag{4}$$

where the $\xi(t)$ are independent random variables. Models like this are used in "time series analysis". Here Z is a continuous variable instead a discrete variable to make the example simpler. If we say that the state at time t is $Z(t)$ then (4) is not a Markov chain. Clearly we do better at predicting $Z(t+1)$ if we know both $Z(t)$ and $Z(t-1)$ than if we know just $Z(t)$. If we say that the state at time t is the two dimensional vector

$$X(t) = \begin{pmatrix} Z(t) \\ Z(t-1) \end{pmatrix} \ ,$$

then

$$\begin{pmatrix} Z(t) \\ Z(t-1) \end{pmatrix} = \begin{pmatrix} a & b \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \xi(t) \\ 0 \end{pmatrix}$$

may be rewriten

$$X(t+1) = AX(t) + \begin{pmatrix} \xi(t) \\ 0 \end{pmatrix} .$$

Thus, $X(t)$ is a Markov chain. This trick of expressing lag models with multidimensional states is common in time series analysis.

The simpler of the evolutions, and the one less used in practice, is the forward evolution for the probabilities $u(x,t)$. Once we know the numbers $u(x,t)$ for all $x \in \mathcal{S}$ and a particular t, we can compute them for $t+1$. Proceding in this way, starting from the numbers $u(x,0)$ for all $x \in \mathcal{S}$, we can compute up to whatever T is desired. The evolution equation for the probabilities $u(x,t)$ is found using conditional probability:

$$\begin{aligned} u(x,t+1) &= \mathbf{Pr}(X(t+1)=x) \\ &= \sum_{y\in\mathcal{S}} \mathbf{Pr}(X(t+1)=x|X(t)=y)\cdot\mathbf{Pr}(X(t)=y) \\ u(x,t+1) &= \sum_{y\in\mathcal{S}} p(y,x)u(y,t) \ . \end{aligned} \tag{5}$$

To express this in matrix form, we suppose that the state space, $\mathcal{S}$, is finite, and that the states have been numbered x_1, ..., x_n. The transition matrix, P, is $n \times n$ and has (i,j) entry $p_{ij} = p(x_i,x_j)$. We sometimes conflate i with x_i and write $p_{xy} = p(x,y)$; until you start programming the computer, there is no need to order the states. With this convention, (5) can be interpreted as vector–matrix multiplication if we define a *row* vector $\underline{u}(t)$ with components $(u_1(t),\ldots,u_n(t))$, where we have written $u_i(t)$ for $u(x_i,t)$. As long as ordering is unimportant, we could also write $u_x(t) = u(x,t)$. Now, (5) can be rewritten

$$\underline{u}(t+1) = \underline{u}(t)P \ . \tag{6}$$

Since $\underline{u}$ is a row vector, the expression $P\underline{u}$ does not make sense because the dimensions of the matrices are incompatible for matrix multiplication. The convention of using a row vector for the probabilities and therefore putting the vector in the left of the matrix is common in applied probability. The relation (6) can be used repeatedly[1]

$$\underline{u}(1) = \underline{u}(0)P \text{ and } \underline{u}(2) = \underline{u}(1)P$$
$$\longrightarrow$$
$$\underline{u}(2) = \left(\underline{u}(0)P\right)P = \underline{u}(0)\left(PP\right) = \underline{u}(0)P^2$$

to yield

$$\underline{u}(t) = \underline{u}(0)P^t \ , \tag{7}$$

where P^t means P to the power t, not the transpose of P.

Actually, the Markov property is a bit stronger than (3). It applies not only to events determined by time $t+1$, but to any events determined in the future of t. For example, if A is the event $X(t+3) = x$ or y and $X(t+1) \neq X(t+4)$, then

$$\mathbf{Pr}(A \mid X(t) = z \text{ and } X(t-1) = w) = \mathbf{Pr}(A \mid X(t) = z) \ .$$

3 Expected Values

The more general and useful evolution equation is the backward evolution for expected values. In the simplest situation, suppose that $X(t)$ is a Markov chain, that the probability distribution $u(x,0) = \mathbf{Pr}(X(0) = x)$ is known, and that we want to evaluate $\mathbf{E}(V(X(T))$. We will call time $t = 0$ the present, time $t = T$ the payout time, and times $t = 1, \cdots, T-1$ intermediate times.

The backward evolution computed the desired expected value in terms of a collection of other conditional expected values, $f(x,t)$, where $x \in \mathcal{S}$ and t is an intermediate time. We start with the final time values $f(x,T) = V(x)$ for all $x \in \mathcal{S}$. We then compute the numbers $f(x,T-1)$ using the $f(x,t)$ and P. We continue in this way back to time $t = 0$.

The $f(x,t)$ are expected values of the payout, given knowledge of the state at a future intermediate time:

$$f(x,t) = \mathbf{E}\left[V(X(T)) | X(t) = x\right] \ . \tag{8}$$

Recall our convention that time 0 is the present time, time $t > 0$ is in the future, but not as far in the future as the time, T, at which the payout is

[1] The most important fact in linear algebra is that matrix multiplication is associative: $(AB)C = A(BC)$ for any three matrices of any size, including row or column vectors, as long as the multiplication is compatible.

made. We may think of the $f(x,t)$ as possible expected values at the future intermediate time t. At time t we would know the value of $X(t)$. If that value were x, then the expected value of $V(X(T))$ would be $f(x,t)$.

Instead of computing $f(x,t)$ directly from the definition (8), we can compute it in terms of the $f(x,t+1)$ using the transition matrix. If the system is in state x at time t, then the probability for it to be at state y at the next time is $p(x \to y) = p(x,y)$. For expectation values, this implies

$$
\begin{aligned}
f(x,t) &= \mathbf{E}\left[f_T(X(T)) | X(t) = x\right] \\
&= \sum_{y \in \mathcal{S}} \mathbf{E}\left[f_T(X(T)) | X(t+1) = y\right] \cdot \mathbf{Pr}\left(X(t+1) = y \mid X(t) = x\right) \\
f(x,t) &= \sum_{y \in \mathcal{S}} f(y,t+1) p(x,y) \quad . \qquad (9)
\end{aligned}
$$

It is clear from (8) that $f(x,T) = V(x)$; if we know the state at time T then we know the payout exactly. From these, we compute all the numbers $f(x,T-1)$ using (9) with $t = T-1$. Continuing like this, we eventually get to $t = 0$. We may know $X(0)$, the state of the system at the current time. For example, if $X(t)$ is the price of a stock at time t, then $X(0) = x_0$ is the current spot price. Then the desired expected value would be $f(x_0, 0)$. Otherwise we can use

$$
\begin{aligned}
\mathbf{E}\left[V(X(T))\right] &= \sum_{x \in \mathcal{S}} \mathbf{E}\left[V(X(T)) | X(0) = x\right] \cdot \mathbf{Pr}\left(X(0) = x\right) \\
&= \sum_{x \in \mathcal{S}} f(x,0) u(x,0) \quad .
\end{aligned}
$$

All the values on the bottom line should be known.

Another remark on the interpretation of (9) will be helpful. Suppose we are at state x at time t and wish to know the expected value of $V(X(T))$. In one time step, starting from state x, we could go to state y at time $t+1$ with probability[2] $p(x,y)$. The right side of (9) is the average over the possible y values, using probability $p(x,y)$. The quantities being averaged, $f(y,t+1)$ are themselves expected values of $V(X(T))$. Thus, we can read (9) as saying that the expected value is the expected value of the expected values at the next time. A simple model for this situation is that we toss a coin. With probability p we get payout U and with probability $1-p$ we get payout V. Let us suppose that both U and V are random with expected values $f_U = \mathbf{E}(U)$

[2]Here we should think of y as the variable and x as a parameter.

and $f_V = \mathbf{E}(V)$. The overall expected payout is $p \cdot f_u + (1-p) \cdot f_V$. The Markov chain situation is like this. We are at a state x at time t. We first choose state $y \in \mathcal{S}$ with probability $p(x,y)$. For each y at time $t+1$ there is a payout probability, U_y, whose probability distribution depends on y, $t+1$, V, and the Markov chain. The overall expected payout is the average of the expected values of the U_y, which is what (9) says.

As with the probability evolution equation (5), the equation for the evolution of the expectation values (9) can be written in matrix form. The difference from the probability evolution equation is that here we arrange the numbers $f_j = f(x_j, t)$ into a *column* vector, $\underline{f(t)}$. The evolution equation for the expectation values is then written in matrix form as

$$\underline{f}(t) = P\underline{f}(t+1) \quad . \tag{10}$$

This time, the vector goes on the right. If apply (10) repeatedly, we get, in place of (7),

$$\underline{f}(t) = P^{T-t}\underline{f}(T) \quad . \tag{11}$$

There are several useful variations on this theme. For example, suppose that we have a running payout rather than a final time payout. Call this payout $g(x,t)$. If $X(t) = x$ then $g(x,t)$ is added to the total payout that accumulates over time from $t=0$ to $t=T$. We want to compute

$$\mathbf{E}\left[\sum_{t=0}^{T} g(X(t),t)\right] \quad .$$

As before, we find this by computing more specific expected values:

$$f(x,t) = \mathbf{E}\left[\sum_{t'=t}^{T} g(X(t'),t') | X(t) = x\right] \quad .$$

These numbers are related through a generalization of (9) that takes into account the known contribution to the sum from the state at time t:

$$f(x,t) = \sum_{y \in \mathcal{S}} f(y,t+1)p(x,y) + g(x,t) \quad .$$

The "initial condition", given at the final time, is

$$f(x,T) = g(x,T) \quad .$$

This includes the previous case, we take $g(x,T) = f_T(x)$ and $g(x,t) = 0$ for $t < T$.

As a final example, consider a path dependent discounting. Suppose for a state x at time t there is a discount factor $r(x,t)$ in the range $0 \leq r(x,t) \leq 1$. A cash flow worth f at time $t+1$ will be worth $r(x,t)f$ at time t if $X(t) = x$. We want the discounted value at time $t = 0$ at state $X(0) = x$ of a final time payout worth $f_T(X(T))$ at time T. Define $f(x,t)$ to be the value at time t of this payout, given that $X(t) = x$. If $X(t) = x$ then the time $t+1$ expected discounted (to time $t+1$) value is

$$\sum_{y \in \mathcal{S}} f(y, t+1) p(x,y) \quad .$$

This must be discounted to get the time t value, the result being

$$f(x,t) = r(x,t) \sum_{y \in \mathcal{S}} f(y, t+1) p(x,y) \quad .$$

4 Duality and Qualitative Properties

The forward evolution equation (5) and the backward equation (9) are connected through a duality relation. For any time t, we compute (8) as

$$\begin{aligned} \mathbf{E}\left[V(X(T))\right] &= \sum_{x \in \mathcal{S}} \mathbf{E}\left[V(X(T)) | X(t) = x\right] \cdot \mathbf{Pr}(X(t) = x) \\ &= \sum_{x \in \mathcal{S}} f(x,t) u(x,t) \quad . \end{aligned} \tag{12}$$

For now, the main point is that the sum on the bottom line does not depend on t. Given the constancy of this sum and the u evolution equation (5), we can give another derivation of the f evolution equation (9). Start with

$$\sum_{x \in \mathcal{S}} f(x, t+1) u(x, t+1) = \sum_{y \in \mathcal{S}} f(y,t) u(y,t) \quad .$$

Then use (5) on the left side and rearrange the sum:

$$\sum_{y \in \mathcal{S}} \left(\sum_{x \in \mathcal{S}} f(x, t+1) p(y,x) \right) u(y,t) = \sum_{y \in \mathcal{S}} f(y,t) u(y,t) \quad .$$

Now, if this is going to be true for any $u(y,t)$, the coefficients of $u(y,t)$ on the left and right sides must be equal for each y. This gives (9). Similarly, it is possible to derive (5) from (9) and the constancy of the expected value.

The evolution equations (5) and (9) have some qualitative properties in common. The main one being that they preserve positivity. If $u(x,t) \geq 0$ for all $x \in \mathcal{S}$, then $u(x,t+1) \geq 0$ for all $x \in \mathcal{S}$ also. Likewise, if $f(x,t+1) \geq 0$ for all x, then $f(x,t) \geq 0$ for all x. These properties are simple consequences of (5) and (9) and the positivity of the $p(x,y)$. Positivity preservation does not work in reverse. It is possible, for example, that $f(x,t+1) < 0$ for some x even though $f(x,t) \geq 0$ for all x.

The probability evolution equation (5) has a conservation law not shared by (9). It is

$$\sum_{x \in \mathcal{S}} u(x,t) = const\,. \tag{13}$$

independent of t. This is natural if u is a probability distribution, so that the constant is 1. The expected value evolution equation (9) has a *maximum principle*

$$\max_{x \in \mathcal{S}} f(x,t) \leq \max_{x \in \mathcal{S}} f(x,t+1) \quad . \tag{14}$$

This is a natural consequence of the interpretation of f as an expectation value. The probabilities, $u(x,t)$ need not satisfy a maximum principle either forward of backward in time.

This duality relation has is particularly transparent in matrix terms. The formula (8) is expressed explicitly in terms of the probabilities at time t as

$$\sum_{x \in \mathcal{S}} f(x,T)u(x,T) \quad ,$$

which has the matrix form

$$\underline{u}(t)\underline{f}(T) \quad .$$

Written in this order, the matrix multiplication is compatible; the other order, $\underline{f}(T)\underline{u}(T)$, would represent an $n \times n$ matrix instead of a single number. In view of (7), we may rewrite this as

$$\underline{u}(0)P^t\underline{f}(T) \quad .$$

Because matrix multiplication is associative, this may be rewritten

$$\left[\underline{u}(0)P^t\right] \cdot \left[P^{T-t}\underline{f}(T)\right] \tag{15}$$

for any t. This is the same as saying that $\underline{u}(t)\underline{f}(T-t)$ is independent of t, as we already saw.

In linear algebra and functional analysis, "adjoint" or "dual" is a fancy generalization of the transpose operation of matrices. People who don't like to think of putting the vector to the left of the matrix think of $\underline{u}P$ as multiplication of (the transpose of) $\underline{u}$, on the right, by the transpose (or adjoint or dual) of P. In other words, we can do enough evolution to compute an expected value either using P its dual (or adjoint or transpose). This is the origin of the term "duality" in this context.

5 Dynamic Programming

Dynamic programming is a method for valuing American style options and other financial instruments that allow the holder to make decisions that effect the ultimate payout. The idea is to define the appropriate value function, $f(x, t)$, that satisfies a nonlinear version of the backwards evolution equation (9). In the real world, dynamic programming is used to determine "optimal" trading strategies for traders trying to take or unload a big position without moving the market, to find cost efficient hedging strategies when trading costs or other market frictions are significant, and for many other purposes. Its main drawback stems from the necessity of computing the cost to go function (see below) for every state $x \in \mathcal{S}$. For complex models, the state space may be too large for this to be practical. That's when things really get interesting.

I will explain the idea in a simple but somewhat abstract situation. As in the previous section, it is possible to use these ideas to treat other related problems. We have a Markov chain as before, but now the transition probabilities depend on a "control parameter", ξ. That is

$$p(x, y, \xi) = \mathbf{Pr}\left(X(t+1) = y | X(t) = x, \xi\right) \quad .$$

In the "stochastic control problem", we are allowed to choose the control parameter at time t, $\xi(t)$, knowing the value of $X(t)$ but not any more about the future than the transition probabilities. Because the system is a Markov chain, knowledge of earlier values, $X(t-1), \ldots$, will not help predict or control the future. Choosing ξ as a function of $X(t)$ and t is called "feedback control" or a "decision strategy". The point here is that the optimal control policy is a feedback control. That is, instead of trying to choose a whole control trajectory, $\xi(t)$ for $t = 0, 1, \ldots, T$, we instead try to choose the feedback functions $\xi(X(t), t)$. We will write $\xi(X, t)$ for such a decision strategy.

Any given strategy has an expected payout, which we write

$$\mathbf{E}_\xi \left[V(X(T)) \right] \quad .$$

Our object is to compute the value of the financial instrument under the optimal decision strategy:

$$\max_\xi \mathbf{E}_\xi \left[V(X(T)) \right] \quad , \tag{16}$$

and the optimal strategy that achieves this.

The appropriate collection of values for this is the "cost to go" function

$$f(x,t) = \max_\xi \mathbf{E}_\xi \left[V(X(T)) | X(t) = x \right] \quad . \tag{17}$$

As before, we have "initial data" $f(x,T) = V(x)$. We need to compute the values $f(x,t)$ in terms of already computed values $f(x,t+1)$. For this, we suppose that the optimal decision strategy at time t is not yet known but those at later times are already computed. If we use control variable $\xi(t)$ at time t, and the optimal control thereafter, we get payout depending on the state at time $t+1$:

$$\mathbf{E} \left[f(X(t+1), t+1) | X(t) = x, \xi(t) \right] = \sum_{y \in \mathcal{S}} f(y, t+1) p(x, y, \xi(t)) \quad .$$

Maximizing this expected payout over $\xi(t)$ gives the optimal expected payout at time t:

$$f(x,t) = \max_{\xi(t)} \sum_{y \in \mathcal{S}} f(y, t+1) p(x, y, \xi(t)) \quad . \tag{18}$$

This is the principle of dynamic programming. We replace the "multiperiod optimization problem" (17) with a sequence of hopefully simpler "single period" optimization problems (18) for the cost to go function.

6 Examples and exercises

1. A stationary Markov chain has three states, called A, B, and C. The probability of going from A to B in one step is .6. The probability of staying at A is .4. The probability of going from B to A is .3. The probability of staying at B is .2, and the probability of going to C is .5. From state C, the probability of going to B is .8 and the probability of going to A is zero. The payout for state A is 1, for state B is 4, and for state C is 9.

a. Compute the probabilities that the system will be in state A, B, or C after two steps, starting from state A. Use these three numbers to compute the expected payout after two steps starting from state A.

b. Compute the expected payouts in one step starting from state A and from state B. These are $f(A, 1)$ and $f(B, 1)$ respectively.

c. See that the appropriate average of $f(A, 1)$ and $f(B, 1)$ agrees with the answer from part a.

2. Suppose a stock price is a stationary Markov chain with the following transition probabilities. In one step, the stock goes from S to uS with probability p and from S to dS with probability $q = 1 - p$. We generally suppose that u (the uptick) is slightly bigger than one while d (the downtick) as a bit smaller. Show that the method for computing the expected payout is exactly the binomial tree method for valuing European style options.

3. Formulate the American style option valuation problem as an optimal decision problem. Choosing the early exercise time is the same as deciding on each day whether to exercise or not. Show that the dynamic probramming algorithm discussed above is the binomial tree method for Amercian style options. The optimization problem (18) reduces to taking the max between the computed f and the intrinsic value.

4. This is the simplest example of the "linear quadratic gaussian" (LQG) paradigm in optimal control that has become the backbone of traditional control engineering. Here $X(t)$ is a real number. The transitions are given by

$$X(t+1) = aX(t) + \sigma G(t) + \xi(t) \; , \tag{19}$$

where $G(t)$ is a standard normal random variable and the $G(t)$ for different t values are independent. We want to minimize the quantity

$$C = \sum_{t=1}^{T} X(t)^2 + \mu \sum_{t=0}^{T-1} \xi(t)^2 \tag{20}$$

We want to find a chioce of the control, ξ, that minimizes $\mathbf{E}(C)$. Note that the dynamics (19) are linear, the noise is gaussian, and the cost function (20) is quadratic. Define the cost to go function $f(x, t)$ to be the cost incurred starting at x at time t ignoring the costs that are incurred at earlier times. Start by computing $f(x, T-1)$ explicitly by minimizing over the single variable $\xi(T-1)$. Note that the optimal $\xi(T-1)$ is a linear function of $X(T-1)$. Using this information, compute $f(x, T-2)$ by optimizing over $\xi(T-2)$, and so on. The LQG model in control engineering justifies linear feedback control in much the same way the gaussian error model and maximum likelihood justifies least squares estimation in statistics.