COMPUTATIONAL FINANCE

Lecture 2: Pricing Futures and Futures Options
A Simple Futures Option Pricing Program

Philip H. Dybvig
Washington University
Saint Louis, Missouri

Copyright © Philip H. Dybvig 1996

Commodity Spot Market (Cash) Prices

- One-way arb
  - Can go long physical commodity
  - cannot go short
- Adjust return by:
  - storage cost
  - use value

E*[Return] $\leq$ r - use value + storage cost

Commodity Futures: Intuition

- Futures price = risk-adjusted expected spot rate
- Can jump down as maturity increases
  - Agriculturals: at expected harvest
  - Oil: at anticipated easing of supply
  - Silver and gold: rare, since stock is large
  - Natural gas: storage expensive, can jump either way
  - Paper: strange; futures but no spot

Forwards versus Futures

A Forward Contract is a commitment to buy a fixed amount at a fixed price at the end.

A Futures Contract gives the holder a daily claim to the variation in the market-determined Futures Price day by day. At maturity, the futures price is given by the underlying commodity price (with cash or physical settlement).

If the interest rate is nonrandom and there is no arbitrage, then the futures price equals the forward price. This is usually a good approximation even when interest rates are random.

One futures contract gives "more bang for the buck" than one forward contract, since the change in value is received immediately rather than at the end. Holding one futures contract is equivalent to holding Q forward contracts, where Q is one over the discount factor to maturity of the contract.

Economic Advantages of Futures over Forwards

- Less margin or credit verification is required, since only one day's worth of information arrival needs to be covered.
- The contract is the same every day.

Valuing Commodity Futures

If the futures price moves as

$$F \quad \begin{array}{l} \nearrow \; F(1+\delta) \\ \searrow \; F(1-\delta) \end{array}$$

this means that an investment of 0 paying the change (called variation) is a fair trade:

$$0 \quad \begin{array}{l} \nearrow \; +F\delta \\ \searrow \; -F\delta \end{array}$$

which is to say that the expected change is zero in the risk-neutral probabilities. For many commodities, this is a good model with $\delta = \sigma\sqrt{\Delta t}$ constant and risk-neutral probabilities of 1/2 and 1/2. (We could take the spacing to be unequal to have volatility fall as the price falls, with the necessary adjustment in the probabilities.)

Valuing and Hedging Futures Options

Valuing commodity futures options is performed exactly like valuing equity options or interest options, once we have established the risk-neutral probabilities.

The simple assumption about risk-neutral probabilities is the one described in the previous slide, although it may be more natural to usual approach for the underlying asset for bond futures options or stock index futures options.

The number of futures to hold is the number needed to give the same difference in value across the two states tomorrow as holding the futures would.

In-class Exercise: Futures Options

Consider a two-period binomial model. The short riskless interest rate is fixed at 10%. A two-period corn option is valued at \$200 today and will go up or down by \$50 each day, with risk-neutral probabilities 1/2 and 1/2. What is the price today of a futures call option with an exercise price of \$50 and maturity one period from now? (To be perfectly clear, it is the call option that matures one period from now; the underlying futures matures two periods from now.)

The Header File fut.h

```
// fut.h
//
// Binomial futures option pricing model: declarations
//
#define MAXTERNODES 40

struct valhedge {
  double value;
  double delta;};
class fut {
public:
  fut(double ttm=.25,int npers=4,double r=.05,double sigma=.3);
  valhedge eurcall(double f0,double X);

private:
  int nper;
  double tinc,r1per,disc,disctm,up,down,pratio,prcup,prcdn;
```

```
double val[MAXTERNODES];};
```

The Test File futtest.cc

```
// futtest.cc
//
// Binomial futures option pricing model: test
//
#include <iostream.h>
#include "fut.h"
main() {
  fut c1;
  valhedge x;
  double futprice,strikeP;
  cout << "\nType the futures price, a space, the strike"
    << " price, and then ENTER.\n"
    << "Make the futures price negative to terminate." << "\n\n";
  while(1) {
    cout << "futures-price strike-price: ";
    cin >> futprice >> strikeP;
    if(futprice < 0.0) {
```

```
  cout << endl;
  return(0);}
if(!cin) {
  cout << "invalid input" << endl;
  return(1);}
x = c1.eurcall(futprice,strikeP);
cout << "call option value = " << x.value << "\n"
  << "hedge (number of contracts) = " << x.delta
  << "\n\n";}}
```

The Implementation File fut.cc: Part 1

```
// fut.cc
//
// Binomial futures option pricing model: implementation
//
#include <math.h>
#include "fut.h"
#define MAX(a,b) (((a) > (b)) ? (a) : (b))
```

The Implementation File fut.cc: Part 2

```
fut::fut(double ttm,int npers,double r,double sigma) {
  nper = npers;
  tinc = ttm/(double) nper;
  r1per = exp(r * tinc);
  disc = 1.0/r1per;
  disctm = exp(-r * ttm);
  up = 1.0 + sigma * sqrt(tinc);
  down = 1.0 - sigma * sqrt(tinc);
  prcup = 0.5*disc;
  prcdn = 0.5*disc;}
```

The Implementation File fut.cc: Part 3

```
valhedge fut::eurcall(double f0,double X) {
  int i,j;
  double futprice;
  valhedge x1;
// initialize terminal payoffs
// i is the number of up moves over the whole life
  for(i=0;i<=nper;i++) {
    futprice = f0 * pow(up,(double) i) * pow(down,(double) (nper-i));
    val[i] = MAX(futprice - X,0);}
// compute prices back through the tree
// j+1 is the number of periods from the end
// i is the number of up moves from the start
  for(j=0;j<nper-1;j++) {
    for(i=0;i<nper-j;i++) {
      val[i] = prcdn * val[i] + prcup * val[i+1];}}
  x1.value = prcdn * val[0] + prcup * val[1];
  x1.delta = (val[1]-val[0]) / (f0*(up-down));
```

```
return(x1);}
```