UNIVERSITY OF EXETER

MTHM005 - Mathematical Sciences Project

Pricing Asian Options

Candidate Number: 145670

Supervised by Prof. Byott

July 12, 2022

Abstract

Insert some cool abstract that makes our examiner go, "hmm that's a pretty good abstract".

All files available here:
https://github.com/leele2/Mathematics-in-Business-Project

Contents

List of Figures and Tables

Chapter 1: Introduction

This report studies a vital financial derivative in today's markets, namely options. The option market is widely considered a venue for informed trading [Li21, Hu14, CGM04], that is investors trading with superior knowledge of the probability distribution of share prices, through either access to private information or skillful processing of public information [Gro75].

The importance of the option market has been shown by empirical studies which speculate that option trading improves information efficiency in the broader stock market [PP06, Li21]. It has also been shown that firms with listed options experience lower implied cost of equity capital [NNT13], indicating that options trading reduces the cost of capital [Li21]. The popularity of the options market can easily be seen by the exponential growth in their trading volume from when standardized, exchange-traded stock options were first listed in The Chicago Board Options Exchange in 1973 [Mar02], shown in Figure 1.1. In 2020 option trading volume became higher than the underlying stock volume for the first time ever [Wayne].

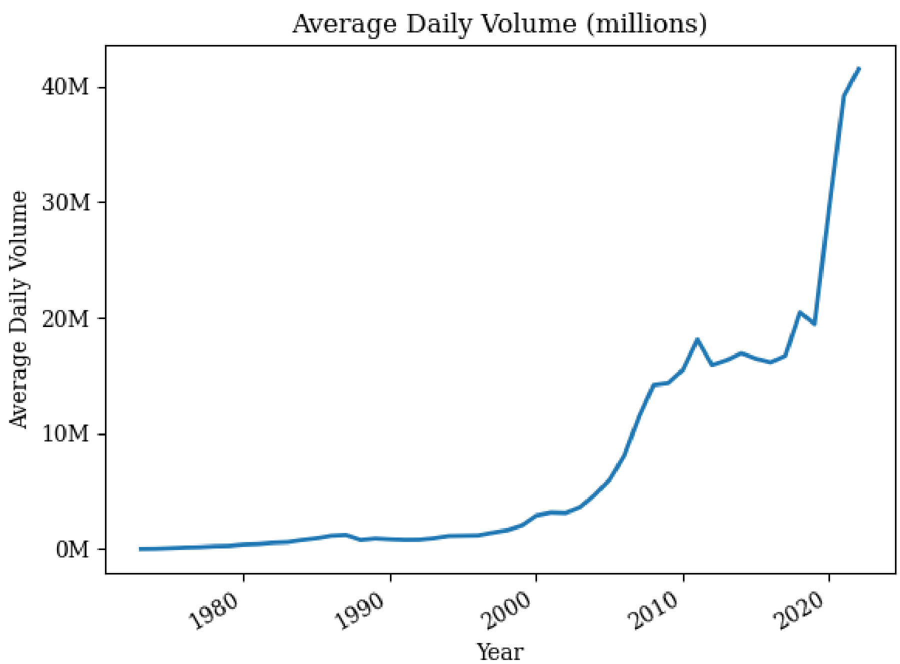

Figure 1.1: Time series plot of the average daily option and future contracts trading volume per annum. Data provided by the Options Clearing Corporation (OCC) [Optne]. Source code Listing B.1

1.1 A brief history of options

1.2 Standard options

A standard option is a contract between two parties which gives the holder the right to buy (or sell) an asset for an agreed upon (exercise) price prior to, or on a determined (expiry) time in the future; regardless of the current (spot) price. Since the holder of the contract is not obliged to exercise the contract at the expiry time, they would not hold any liability in the absence of a price to purchase the option. The problem then becomes what is the correct price to charge the holder of the option to balance this inequality of liability.

1.3 Asian options

Whilst standard options, namely European and American style involve using the spot price as the underlying value of the asset; this is not always the case with so-called exotic options. Exotic options differ in their payment structures, expiration dates, and/or strike prices. In the case of exotic fixed-strike price Asian options, the averaging price of the asset is used in place of the underlying asset value. This differs from fixed-price Asian options which instead use the averaging price of the asset to take place of the strike price. These are the two main variations of Asian style options but both of these can be varied further in how the averaging is calculated, for example: geometrically, arithmetically, average taken every day or average taken at the start of each month and so on. They can be varied further by having an expiry structure matching a European or American style option.

Bibliography

[CGM04] Sugato Chakravarty, Huseyin Gulen, and Stewart Mayhew. Informed trading in stock and option markets. *The Journal of Finance*, 59(3):1235–1257, 2004.

[Gro75] Steven D Grossman. An application of communication theory to financial statement analysis. *Financial Review*, 10(1):12–20, 1975.

[Hu14] Jianfeng Hu. Does option trading convey stock price information? *Journal of Financial Economics*, 111(3):625–645, 2014.

[Li21] Keming Li. The effect of option trading. *Financial Innovation*, 7(1):1–32, 2021.

[Mar02] J.W. Markham. *A Financial History of the United States Volume III (1970-2001) From the Age of Derivatives to the New Millennium*. A Financial History of the United States. M.E. Sharpe, 2002.

[NNT13] Vic Naiker, Farshid Navissi, and Cameron Truong. Options trading and the cost of equity capital. *The Accounting Review*, 88(1):261–295, 2013.

[Optne] Options Clearing Corporation. Annual Volume and Open Interest Statistics. https://www.theocc.com/Market-Data/Market-Data-Reports/Volume-and-Open-Interest/Historical-Volume-Statistics, 24th June 2022 [Online]. Accessed: 08-06-2022.

[PP06] Jun Pan and Allen M. Poteshman. The Information in Option Volume for Future Stock Prices. *The Review of Financial Studies*, 19(3):871–908, 02 2006.

[Wayne] Wayne Duggan. Option Trading Volume Higher Than Underlying Stock Volume For First Time Ever. https://finance.yahoo.com/news/option-trading-volume-higher-underlying-211006236.html?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAGAPwh_iWDvxoDcVe3TMvrMR9gvv7XIYDY0XVc15LPN5ZLNnVmaq149OIXGIFPOPbL2J31GH2JpvFN_eb7bzK4n042h7DWpHfmW5Y-EgRt9l_3vWsKIIB-jMSN7bCqAaIl4Lkx3Jfv_vXhajuXKmbUnP3fZnEA5h54dj1GmRNw1G, 8th July 2020 [Online]. Accessed: 24-06-2022.

Appendix A: Matlab Files

All files can be found: https://github.com/leele2/Mathematics-in-Business-Project/tree/master/MATLAB%20Files

Listing A.1: ../../MATLAB Files/BinoAsian.m

```
%function BinoAsian(S0,E,T,r,sigma,N,F)
%% Test
clear; tic; S0=50;  E=50; T=1.0; r=0.1; sigma=0.3; N=50; F=@(S,A)max(A-E,0);
%% Function to evaluate European Call option by Binomial Method
%   Parameters:
%   S0 = initial share price
%   E = exercise price`
%   T = time to expiry
%   r = riskfree interest rate
%   sigma = volatility
%   N = Number of steps
%   F = Option Payoff (European Call in given)
%% Calculated parameters
dt   = T/N;                   %Timestep
u    = exp(sigma*sqrt(dt));   %Up price movement
d    = 1/u;                   %Down price movement
disf = exp(-r*dt);            %Discount factor over each timestep
p    = (1/disf - d)/(u-d);    %Risk-neutral probability
%% Initalizing Arrays and Functions
S      = zeros(N+1,N+1);   %Underlying Asset Price
S_k    = cell(N+1,N+1);    %S_max %% Cells are used so different sized vectors can be stored at each element in cell array
A      = zeros(N+1,N+1);   %Average of Underlying Asset Price
A_k    = cell(N+1,N+1);    %Representitive averages
C      = zeros(N+1,N+1);   %Price of Option
C_k    = cell(N+1,N+1);    %Option price for given representitive avg
%F      = @(S,A)max(A-E,0); %Option Payoff (European Call)
%% Functions to Calculate Maximum and Minimum Representative Averages
A_min = @(i,j) (1/(i+1)) * (sum(S0*d.^(0:i-j)) + sum(S0*d.^(0:(j-1)+i-(2*j))));
A_max = @(i,j) (1/(i+1)) * (sum(S0*u.^(0:j)) + sum(S0*u.^((0:(i-j-1))+(2*j)-i)));
%% Calculate Underlying asset price
for i = 0:N
    for j = 0:i
        S(i+1,j+1) = S0*u^(j)*d^(i-j);
    end
end
%% Calculating All S_max and Representitive Averages
for i = 0:N
    for j = 0:i %j indexes at j+1 due to matlab not allowing C_k{:,0)
        A_k{i+1,j+1} = zeros(j*(i-j)+1,1); %Create Vector to hold rep avgs
        S_k{i+1,j+1} = NaN(j*(i-j),1);     %Create Vector to hold S_max
        A_k{i+1,j+1}(1) = A_max(i,j);      %Assign A_max to first element in vector
        % Paths with only up (i = j) or down movements (j = 0) or i = 1 will only have one representative average
        if i < 1 || i == j || j == 0
            S_k{i+1,j+1}(1) = S0*(u^j)*d^(i-j);
            continue
        end
        %% Filling S_max Vector
        % Setting Paramaters
        n = j*(i-j);         %Size of vector
        rep = min([j;i-j]); %Maximum element repetition in vector
        % Calculating unique elements in vector
        unique = 2*numel(1:rep-1) + (n - (2*sum(1:rep-1)))/rep;
        % First and last values
        S_k{i+1,j+1}(1)   = S0 * (u ^ j);
        S_k{i+1,j+1}(end) = S0 * (u ^ (j - unique + 1));
        %%The following if statement changes the final output%%
        if n ~= 2
            % Ascending/Descending repeated values
            k     = 2; %iterator
            count = 2; %Number of elements filled
            while k < rep
                % Filling from top
                S_k{i+1,j+1}(sum(1:k-1)+1:sum(1:k-1)+k) = repmat(S0*(u^(j-k+1)), k, 1);
                % Filling from bottom
                S_k{i+1,j+1}(n-sum(1:k-1)+1-k:n-sum(1:k-1)) = repmat(S0*(u^(j-unique+k)), k, 1);
                count = count + 2*k;
                k = k + 1;
            end
            % "Middle" repeated values
            for l = 0 : (n-count)/rep - 1
                S_k{i+1,j+1}(count/2+1+l*rep:end) = [repmat(S0*u^(j-k+1), rep, 1);
                    S_k{i+1,j+1}(count/2+1+l*rep+rep:end)];
                k = k + 1;
            end
        end
        %% Calculating Representitive Averages Using S_max
```

```
        A_k{i+1,j+1}(j*(i-j) + 1) = A_min(i,j); %Assign A_min to last element in vector
        for k = 2:j*(i-j)
            A_k{i+1,j+1}(k) = A_k{i+1,j+1}(k-1) - (1/(i+1))* ...
                (S_k{i+1,j+1}(k-1) - S_k{i+1,j+1}(k-1)*(d^2));
        end
    end
end
%% Pricing Option Value at Final Time (N)
for j = 0:N
    C_k{N+1,j+1} = F(S(N+1,j+1),A_k{N+1,j+1});
end
%% Pricing Option
err = 1e-3;
for i = N-1:-1:0
    for j = 0:i
        C_k{i+1,j+1} = zeros(j*(i-j)+1,1);
        for k = 1:j*(i-j)+1
            %% Find K_u
            Ku  = ( (i+1)*A_k{i+1,j+1}(k) + u*S(i+1,j+1) )/(i+2);
            [loc, ubound, lbound] = findInSorted(Ku,A_k{i+2,j+2},err);
            % If found set accordingly
            if loc > 0
                Cu = C_k{i+2,j+2}(loc);
            else
                % If not found, interpolate between closest values
                Cu = C_k{i+2,j+2}(lbound)+(Ku-A_k{i+2,j+2}(lbound))*(...
                    (C_k{i+2,j+2}(ubound)-C_k{i+2,j+2}(lbound))/...
                    (A_k{i+2,j+2}(ubound)-A_k{i+2,j+2}(lbound)));
            end
            %% Find K_d
            Kd = ( (i+1)*A_k{i+1,j+1}(k) + d*S(i+1,j+1) )/(i+2);
            [loc, ubound, lbound] = findInSorted(Kd,A_k{i+2,j+1},err);
            %If found set accordingly
            if loc > 0
                Cd = C_k{i+2,j+1}(loc);
            else
                %If not, interpolate between closest values
                Cd = C_k{i+2,j+1}(lbound)+(Kd-A_k{i+2,j+1}(lbound))*(...
                    (C_k{i+2,j+1}(ubound)-C_k{i+2,j+1}(lbound))/...
                    (A_k{i+2,j+1}(ubound)-A_k{i+2,j+1}(lbound)));
            end
            %% Calculate option value at previous node (i,j decreasing)
            C_k{i+1,j+1}(k) = disf*(p*Cu + (1-p)*Cd);
        end
    end
end
disp(C_k{1,1})
toc;
```

Listing A.2: ../../MATLAB Files/Functions/findInSorted.m

```
function [loc, A, B] = findinSorted(x,range,err)
%%%%%%%%%%%%%% inputs %%%%%%%%%%%%%%
% x = value to find                %
% range = (sorted) vector to search  %
% err = tolerence on search          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Outputs
A   = 1;              %left boundary
B   = numel(range); %right boundary
loc = 0;              %initially set to 0 implying not found
%% Binary Search Algorithm
while B-A > 1
    mid = floor((A+B)/2);
    if x > range(mid)
        B = mid;
    else
        A = mid;
    end
end
%% Returning returned value if found (within tolerance)
if abs(range(A) - x) < err
    loc = A;
    return
elseif abs(range(B) - x) < err
    loc = B;
    return
end
```

Appendix B: Python Files

All files can be found: https://github.com/leele2/Mathematics-in-Business-Project/tree/master/Python%20Files

Listing B.1: ../../Python Files/OptionVolume.py

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Jun  8 18:05:55 2022
Scraping and plotting data from theocc.com
@author: leele2
"""
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from datetime import datetime
from bs4 import BeautifulSoup
from os import listdir
from pathlib import Path

def string_to_int(string):
    if not string:
        return 0
    if string[0] == "$":
        string = string[1:]
    return int(string.replace(',', ''))

# Ensuring html file is on path
"""
html file taken from:
https://www.theocc.com/Market-Data/Market-Data-Reports/Volume-and-Open-Interest/Historical-Volume-Statistics
"""
html_file = [f for f in listdir('.') if f.endswith('.html')]
if len(html_file) != 1:
    raise ValueError('should be exactly one html file in the current directory')
# Importing html file
with open(html_file[0], 'r', encoding='utf-8') as file:
    soup = BeautifulSoup(file, 'lxml')
# Finding Table
table = soup.find_all("table")
# Selecting Data
table_data = table[0].find_all("tr")
# Finding Table Names
table_names = []
for names in table_data[0]:
    table_names.append(str(names.find("span").string))
del table_names[0]
del table_names[-1]
# Finding Headers
headers = []
for header in table_data[1]:
    headers.append(str(header.find("span").string))
# Populating Data
Data = {}
for rows in table_data[2:]:
    temp = {}
    for i, data in enumerate(rows):
        if i > 6 and i < 10 or i == 0:
            temp[headers[i]] = data.get_text()
    Date = datetime.strptime(temp["Date"], "%Y")
    del temp["Date"]
    Data[Date] = {k: string_to_int(v) for k, v in temp.items()}
# Converting data into pandas dataframes
Data = pd.DataFrame.from_dict(Data, orient="index")
# Plotting time-series
plt.rc('font', family='serif')
fig, ax1 = plt.subplots()
plt.gcf().set_size_inches(6, 4.5)
ax1.plot(Data['Options']/1e6, linewidth=2, zorder=1, label="Options")
# ax1.plot(Data[Data['Futures'] != 0]['Futures']/1e6, linewidth=2, zorder=1, label="Futures")
# ax1.legend()
ax1.set_title('Average Daily Volume (millions)')
ax1.set_xlabel('Year')
ax1.xaxis.set_minor_formatter(mdates.DateFormatter("%Y"))
ax1.set_ylabel('Average Daily Volume')
ax1.yaxis.set_major_formatter(mpl.ticker.StrMethodFormatter('{x:,.0f}M'))
fig.autofmt_xdate()
plt.tight_layout()
plt.show()
# Saving plot
cwd = str(Path(__file__).parent.parent.absolute())
fig.savefig(cwd + "\Latex_Files\Main\Chapters\C1\plots\OptionVolume.png", bbox_inches='tight')
```