|  |  |  |
| --- | --- | --- |
|  | | |
|  | Foreign Exchange Option | [All Projects (P)](https://seattleu.instructure.com/courses/1589251/assignments/6766679#dialog_for_Project) |

Write a foreign exchange option pricing function.

Foreign Exchange Options

Many companies hedge foreign currency exchange risk versus their domestic currency (say US Dollars for American companies). If future foreign cash flows are certain, the best hedge is to enter into a forward contract to exchange foreign currency for domestic currency on some future date at a specific exchange rate. These forward FX markets are quite large, liquid, and efficient. But if the future foreign cash flows are uncertain, many companies still want to hedge those risks and to do so they choose ***options***.

Forex options are contracts that allow one of the parties to force an exchange of two given currencies at a given rate on a date in the future. The currency pair is called the option *underlier,*the rate of future exchange is called the option *strike price*, and the future date is called the option *exercise date*. The choice to do the exchange or not on the exercise date is at the option of the option holder on that future date. So if the strike price is better than the exchange rate on the exercise date, the option is deemed *in the money* and the holder will choose to exercise the option and collect a profit. If the option is *out of the money* on exercise date, then the holder will choose not to exercise and the option will expire worthless.

Sometimes we are the holder of these options and sometimes it is the other party that holds the option. If we hold the option and the exchange is for us to pay with dollars for receiving the foreign currency, the option is described as a *call* option and its quantity is the number of foreign currency that will be received. For example, holding 100 call options on Great Britain Pounds (GBP) with strike 1.5 would mean that three months from now we could choose to buy 100 GBP for 150 United States Dollars (USD). If we hold the option but we are selling foreign currency for dollars coming in, this is described as a *put* option, e.g., 100 put options on GBP with strike 1.5 means we could choose to sell 100 GBP for 150 USD. If the other party holds the option, we still describe it as a put or call from their point of view, but the quantity of our position is negative. For example, if Deutschebank holds an option to buy 100 GBP from us for 150 USD, we have a position of -100 call options on GBP struck at 1.5. Note that there are **four** possible types of option positions to have for a given currency pair at a particular strike:

* long a call option (we have the option to buy currency from them for dollars),
* short a call option (they have the option to buy currency from us for dollars),
* long a put option (we have the option to sell currency to them for dollars), and
* short a put option (they have the option to sell currency to us for dollars).

The options are interesting in that they are non-linear. When they are in the money, they behave linearly like holding a foreign currency position, but when they are out of the money they are worthless. You can think of them as bets--the holder of the option is like the player in the casino and the writer of the option is like the casino. And analogously, just as a casino requires a wager from the player, the writer of the option requires an upfront payment (called the *premium*) in order to give the holder the opportunity for potential future winnings.

We will use FX options later in the course, too, in the [final project](https://seattleu.instructure.com/courses/1589251/assignments/6766685).

Option Valuation

There has been a lot of research on valuation of these options, including that done by Fischer Black, Myron Scholes,  and Robert Merton resulting in the Nobel-winning [Black-Scholes (Links to an external site.)](https://en.wikipedia.org/wiki/Black%E2%80%93Scholes_model) pricing model. This was later applied to currency markets by Mark Garman and Steven Kohlhagen. We will use the [Garman-Kohlhagan model (Links to an external site.)](https://en.wikipedia.org/wiki/Foreign_exchange_option#Garman%E2%80%93Kohlhagen_model) to price our options in this assignment. We will write a function that takes the option contract parameters (type--call or put, strike price, and expiration date) as well as market observations (spot exchange rate, volatility, and interest rates in both the domestic and foreign currencies) and returns the fair market value of the option contract.

The work will be decomposed into several functions and the pricing function will use these lower-level functions to calculate its result. The functions you will write are:

1. fx\_option\_price(call, strike, expiration, spot\_date, spot, volatility, domestic\_rate, foreign\_rate)
2. fx\_option\_d2(term, volatility, d1)
3. fx\_option\_d1(strike, term, spot, volatility, domestic\_rate, foreign\_rate)
4. discount(rate, term)
5. years\_apart(date1, date2)

The signatures, specifications, and a few tests are shown below for each function. The  tests inside the function docstrings are called doctests and should run as shown with the exact same output. We'll talk more about doctests in [Worksheet 13 - Doctests](https://seattleu.instructure.com/courses/1589251/pages/worksheet-13-doctests).

def years\_apart(date1, date2):  
 """Returns the fractional difference in years between the given dates.  
 Assumes a 365-day year for the fractional part.  
 >>> years\_apart(date(1959, 5, 3), date(1960, 5, 3))  
 1.0  
 >>> years\_apart(date(2004, 1, 1), date(2005, 1, 2)) # 365 days even if a leap year  
 1.0027397260273974  
 >>> years\_apart(date(1959, 5, 1), date(2019, 6, 2))  
 60.087671232876716  
 >>> years\_apart(date(2019, 7, 1), date(2019, 4, 1)) # reversed is ok  
 0.2493150684931507  
 """  
 # YOUR IMPLEMENTATION HERE - USE A LOOP THROUGH THE YEARS  
  
def discount(rate, term):  
 """Calculate the discount factor for given simple interest rate and term.  
 present\_value = future\_value \* discount(rate, term)  
 >>> discount(0.123, 0.0)  
 1.0  
 >>> discount(0.03, 2.1)  
 0.9389434736891332  
 """  
 # YOUR IMPLEMENTATION HERE  
  
  
def fx\_option\_d1(strike, term, spot, volatility, domestic\_rate, foreign\_rate):  
 """Calculate the d1 statistic for Garman Kohlhagen formula for fx option  
 >>> '%.10f' % fx\_option\_d1(152, 91/365, 150, 0.13, 0.03, 0.04)  
 '-0.2100058012'  
 """  
 # YOUR IMPLEMENTATION HERE  
  
  
def fx\_option\_d2(term, volatility, d1):  
 """Calculate the d2 statistic for Garman Kolhagen formula for fx option  
 >>> '%.10f' % fx\_option\_d2(91/365, 0.13, -0.21000580120118273)  
 '-0.2749166990'  
 """  
 # YOUR IMPLEMENTATION HERE  
  
def fx\_option\_price(call, strike, expiration, spot\_date,  
 spot, volatility, domestic\_rate, foreign\_rate):  
 """  
 Calculates the fair price of a currency option.  
 :param call: True if this is a call option, False if this is a put option  
 :param strike: units of domestic currency per unit of foreign currency to be exchanged  
 :param expiration: date on which the exchange would take place if exercised  
 :param spot\_date: date of valuation  
 :param spot: market exchange rate for fx exchanged on spot\_date (same units as strike)  
 :param volatility: standard deviation of the logarithmic returns of holding this foreign currency (annualized)  
 :param domestic\_rate: simple risk-free interest rate from spot\_date to expiration\_date (annualized)  
 :param foreign\_rate: simple risk-free interest rate from spot\_date to expiration\_date (annualized)  
 :return: option value in domestic currency for one unit of foreign currency  
  
 >>> '%.10f' % fx\_option\_price(True, 152, date(2019,7,1), date(2019,4,1), 150, 0.13, 0.03, 0.04)  
 '2.8110445343'  
 >>> '%.10f' % fx\_option\_price(False, 152, date(2019,7,1), date(2019,4,1), 150, 0.13, 0.03, 0.04)  
 '5.1668650332'  
 """  
 # YOUR IMPLEMENTATION HERE - MUST USE THE OTHER FUNCTIONS ABOVE WHENEVER POSSIBLE

Guidance

Cumulative Normal Distribution Function

We can get the CDF calculation for the normal distribution from the [scipy (Links to an external site.)](https://scipy.org/" \t "_blank) library. Install the [matplotlib (Links to an external site.)](https://matplotlib.org/) package on your machine (as shown at the beginning of the course). This package includes the scientific Python library, scipy. Within scipy is a [statistics library (Links to an external site.)](https://docs.scipy.org/doc/scipy-0.16.1/reference/generated/scipy.stats.norm.html) with a normal CDF function.

At the top of your file, import scipy's stat library as shown:

from scipy.stats import norm # normal distribution statistics library

And then to call the CDF function, it's just:

cdf\_d1 = norm.cdf(d1)

E and Natural Logarithms

These are implemented in the [math (Links to an external site.)](https://docs.python.org/3/library/math.html) library. The discount factor is calculated as:

e − r T

The math library has math.exp(x) which computes e xand math.log(x) which computes ln ⁡ ( x ). Finally, math.e is e itself.

Dates

Dates and date arithmetic are implemented in the [datetime (Links to an external site.)](https://docs.python.org/3/library/datetime.html) library. I recommend just using the date class for this assignment. Import it like so:

from datetime import date

I want you to keep your implementation of years\_apart real simple without resorting to use of dateutils or anything else. The algorithm is to count years until you surpass the second date, then back up and count the number of days in the final year. To get the date a year from another date, you can do this:

next\_year = date(date1.year + 1, date1.month, date1.day)

You will need to compute the number of days in the difference of two dates. Something like this:

final\_year = date2 - date1  
final\_year\_fraction = final\_year.days / 365

[Previous](https://seattleu.instructure.com/courses/1589251/modules/items/16887848)[Next](https://seattleu.instructure.com/courses/1589251/modules/items/16887850)

**Submission**

**Submitted!**

Feb 16 at 7:58pm

[Submission Details](https://seattleu.instructure.com/courses/1589251/assignments/6766679/submissions/6208827)

[Download p5-1.py](https://seattleu.instructure.com/courses/1589251/assignments/6766679/submissions/6208827?download=65263831)

Grade: 96 (100 pts possible)

Graded Anonymously: no

**Comments:**