# CHEME 5660 Actual Prelim 2 Options Question - Lukas Wenzl

A trader at Olin Financial, an up-and-coming hedge fund, sold a short strangle on firm XYZ with a `01/20/2023` expiration.

**Assumptions**: (i) the short put (contract 1) has a strike price of $K_1$ = 230.0 USD/share and an implied volatility of 58.97\%; (ii) the short call (contract 2) has a strike price of $K_2$ = 300.0 USD/share and an implied volatility of 52.59\%; (iii) there are 78 days to 01/20/2023 (from today); (iv) the current share price of XYZ is 270.89 USD/share; (v) the risk-free rate is 4.10\%.

Use the Jupyter notebook `CHEME-5660-PP2-Options.ipynb`, and any associated data sets or other course materials to answer the following questions:

- a) Compute the premiums for the put $\mathcal{P}_1$ and call $\mathcal{P}_2$ contracts for the `01/20/2023` short strangle on firm XYZ using the Cox, Ross, and Rubinstein (CRR) binomial lattice model.
- b) Compute the maximum profit and break-even points for the Olin Financial short strangle position on XYZ.
- c) Compute the probability of the profit at expiration for the short strangle position by sampling the share price distribution from the `Equity` notebook.

## Solution

In [12]:
```
import Pkg; Pkg.activate("."); Pkg.resolve(); Pkg.instantiate();
```
```
  Activating project at `~/OtherCodes/CHEME-5660-Markets-Mayhem-Example-Notebooks/prelims/P2/actual`
  No Changes to `~/OtherCodes/CHEME-5660-Markets-Mayhem-Example-Notebooks/prelims/P2/actual/Project.toml`
  No Changes to `~/OtherCodes/CHEME-5660-Markets-Mayhem-Example-Notebooks/prelims/P2/actual/Manifest.toml`
```

In [13]:
```
# load external packages that are required for the calculations -
using DataFrames
using CSV
using Dates
using Statistics
using LinearAlgebra
using Plots
using Colors
using Distributions
using StatsPlots
using PQEcolaPoint

# setup paths to load XYZ OHLC data set -
const _NOTEBOOK_ROOT = pwd();
const _PATH_TO_DATA = joinpath(_NOTEBOOK_ROOT, "data");
```

In [14]:
```
include("CHEME-5660-AP2-CodeLib.jl"); # Look inside me to find out what I have!
```

In [15]:
```
# Date -
#D = Date(2022, 12, 16); # date when contracts expire
D = Date(2023, 1, 20); # date when contracts expire

# contract 1 parameters -
IV₁ = 58.97; # implied volatility for contract 1 (short put)
K₁ = 230.0;   # strike price short put
T₁ = ticker("P", "XYZ", D, K₁);

# contract 2 parameters -
IV₂ = 52.59; # implied volatility for contract 2 (short call)
K₂ = 300.00;  # strike price short call #corrected K2. Template had 350.5
T₂ = ticker("C", "XYZ", D, K₂);

# setup some shared constants
B = 365.0;   # number of days in a year
μ = 0.0410;  # risk-free rate
DTE = 78.0;  # days to expiration

# What is the current share price?
Sₒ = 270.89;

# How many levels on the tree?
L = 100;

# How many sample do we have?
number_of_samples = 10000;
```

### a) Estimate the price of the call and put contracts

In [16]:
```
# build the contracts for the trades -
put_contract_model = build(PutContractModel, (
    ticker = T₁,
    expiration_date = D,
    strike_price = K₁,
    premium = 0.0,
    number_of_contracts = 1,
    direction = 1, # Bug or feature? => always use 1 here (even if we are selling)
    current_price = 0.0
));

call_contract_model = build(CallContractModel, (
    ticker = T₂,
    expiration_date = D,
    strike_price = K₂,
    premium = 0.0,
    number_of_contracts = 1,
    direction = 1, # Bug or feature? => always use 1 here (even if we are selling)
    current_price = 0.0
));
```

In [17]:
```
# build the lattice models -
#note: different since IV are different
lattice_model_put = build(CRRLatticeModel; number_of_levels=(L+1), Sₒ = Sₒ, σ = (IV₁/100.0), μ = μ, T = (DTE/B));
lattice_model_call = build(CRRLatticeModel; number_of_levels=(L+1), Sₒ = Sₒ, σ = (IV₂/100.0), μ = μ, T = (DTE/B));
```

In [18]:
```
# compute the price of the put contract using the premium method
price_put_contract = premium(put_contract_model, lattice_model_put);
println("Put contract price = $(price_put_contract) (USD/share)")
```
```
Put contract price = 10.679189410786357 (USD/share)
```

In [19]:
```
# compute the price of the call contract using the premium method
price_call_contract = premium(call_contract_model, lattice_model_call);
println("Call contract price = $(price_call_contract) (USD/share)")
```
```
Call contract price = 16.078618669611114 (USD/share)
```

### b) Compute the maximum profit and breakeven points for the short strangle at expiration

In [9]:
```
# Compute max profit and break even points for the short strange ...
```

In [20]:
```
# max profit occurs when trade is opened -
max_profit = (price_put_contract + price_call_contract)
println("Max profit = $(max_profit) (USD/share)")
```
```
Max profit = 26.75780808039747 (USD/share)
```

In [21]:
```
# Break even 1 (low) -
B₁ = K₁ - (price_call_contract + price_put_contract)
println("Break-even share price (low) B₁ = $(B₁) (USD/share)")
```
```
Break-even share price (low) B₁ = 203.24219191960253 (USD/share)
```

In [22]:
```
# Break even 2 (high) -
B₂ = K₂ + (price_call_contract + price_put_contract)
println("Break-even share price (high) B₂= $(B₂) (USD/share)")
```
```
Break-even share price (high) B₂= 326.75780808039747 (USD/share)
```

### c) Compute the probability of profit at expiration

The probability of profit for a short strangle is the probability that we close between the low and high break-even points. Thus, we are looking for the value:

$$P(B_1 < X \leq B_2) = F_X(B_2) - F_X(B_1)$$

where $B_\star$ denotes the respective break-even points, and $F_X(x)$ is:

$$F_X(x) = P(X \leq x)$$

that is, the probability that a random-variable $X$ is less than (or equal) to a specified value $x$. Check out the `P` function in the CodeLib to estimate $P(X \leq x)$ or to fit a cumulative distribution, see [the Distributions.jl documentation](https://juliastats.org/Distributions.jl/stable/).

In [23]:
```
# build a log normal distribution from the previous GBM simulations
#these values are copied over from the other notebook
μ̂ = 5.91219
σ̂ = 0.24192
d = LogNormal(μ̂, σ̂);
```

In [25]:
```
# sample the future price distribution -
N = 10000;
S_expiration = rand(d,N);
```

In [26]:
```
# compute the probability of profit -

# we are profitable if we close between the break even prices -
prob_less_than_B1 = P(S_expiration, B₁); #this function gets probability less than second argument to function
prob_less_than_B2 = P(S_expiration, B₂);
```

In [27]:
```
#printing out the probability of profit at expiration
POP = prob_less_than_B2 - prob_less_than_B1 #difference since both are prob less than
```

Out[27]:
```
0.2997
```

In [ ]:
```
```