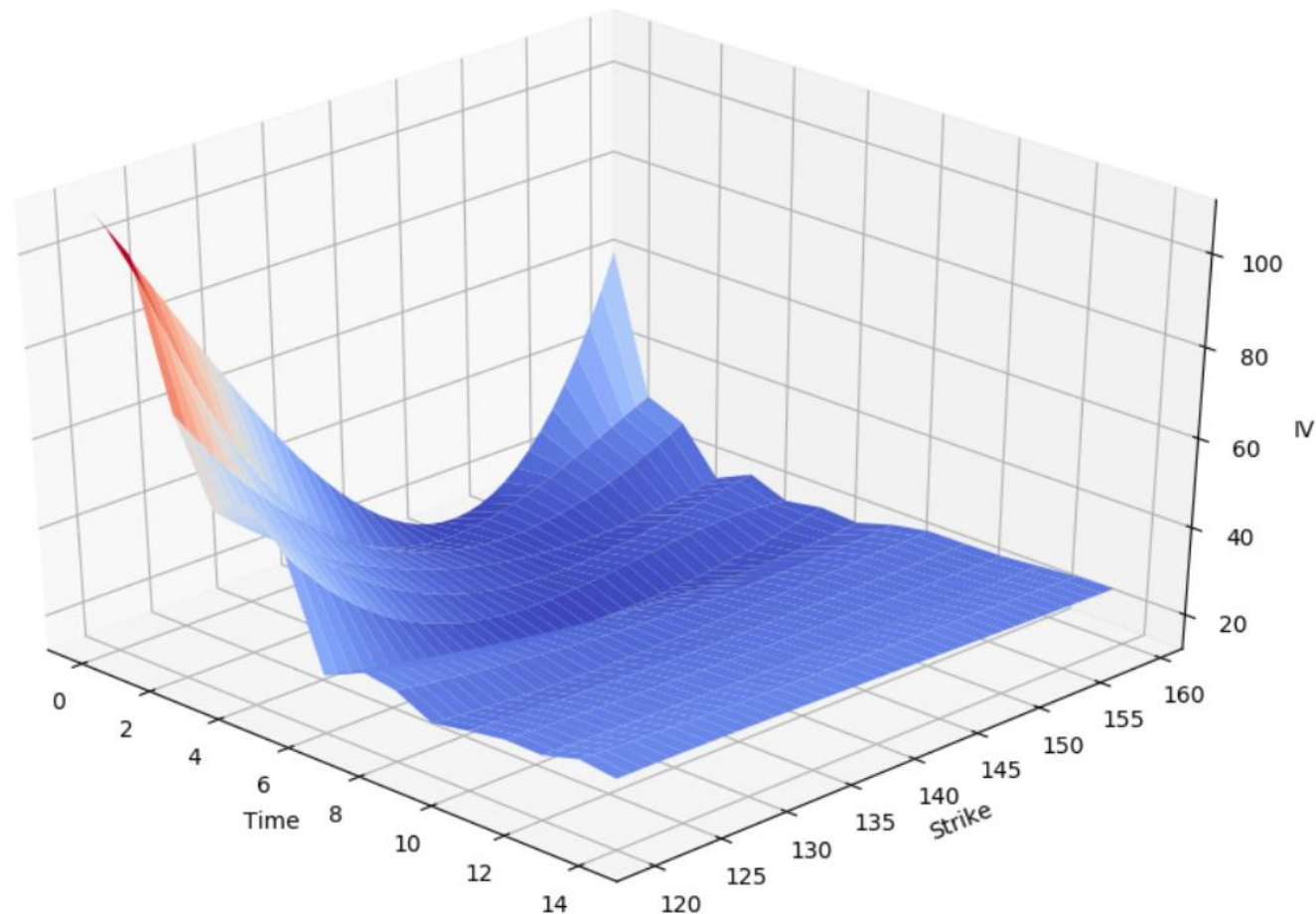


Working on this lecture while
at Aqua Caliente. The view
from outside the window.



Implied volatility ... and the Greeks

IV surface plotted by u/Nathan-T1



[-] **Nathan-T1** 1 point an hour ago

If you are interested in modeling vol surfaces then I would strongly recommend learning the basics of python. <https://imgur.com/a/NXqi2NM> This is a volatility surface for SPY on a specific trading day. It wasn't too hard to whip together a python script that pulls the entire option chain data for SPY for free and then solve for IV using the method of your choice. I used a simple Newton Raphson algorithm.

You can then use quadratic interpolation to make a 3d surface. If you end up going with this route feel free to steal some of the code from GitHub: <https://github.com/Nathan-T1/OptionDB>. I used Tradier's free API for my data.

[permalink](#) [embed](#) [save](#) [report](#) [give award](#) [reply](#)

Implied Volatility

Our theory suggests that the price of a call will be a function of the price of the stock, the strike price of the call, the volatility of the stock, and the time to expiration:


$$P_c = f(P_s, SP, \sigma_s, t)$$

Therefore we can conclude that if certain mathematical conditions are met, we can restructure this so that volatility is the dependent variable:

$$\sigma_s = g(P_c, P_s, SP, t)$$

When these functions take explicit form, as they do in the Black-Scholes - Merton and other options pricing models, then the solution value for volatility is called *implied volatility*.

Calculating implied volatility with B/S:

$$d_1 = \frac{\ln\left(\frac{SP}{STR}\right) + \sigma^2/2}{\sigma}$$


Easy to do:

Once Black-Scholes is structured, you can use an iterative technique to solve for σ .

```
34 # Below we calculate implied daily volatility
35 #
36 while tempcp < callpr:
37     cipd = cipd + 0.00001
38     d1 = math.log(stopr/strike)+((rfir/365)+(cipd**2)/2)*days
39     durvol = cipd*math.sqrt(days)
40     cumd1 = csnd(d1/durvol)
41     cumd2 = csnd((d1/durvol) - durvol)
42     discount = math.exp(-rfir*days/365)
43     tempcp = (stopr*cumd1)-(strike*discount*cumd2)
44 #
```

callidv.py

Go to our Jupyter Notebooks segment and find callidv.html, which is an elementary iterative program for using BSM to solve for implied volatility. Be prepared to create you own non-Jupyter version of it AND another for puts. The iterative converter is intentionally crude ... you may want to improve it.

```

34 # Below we calculate implied daily volatility
35 #
36 while tempcp < callpr:
37     cipd = cipd + 0.00001
38     d1 = math.log(stopr/strike)+((rfir/365)+(cipd**2)/2)*days
39     durvol = cipd*math.sqrt(days)
40     cumd1 = csnd(d1/durvol)
41     cumd2 = csnd((d1/durvol) - durvol)
42     discount = math.exp(-rfir*days/365)
43     tempcp = (stopr*cumd1)-(strike*discount*cumd2)
44 #

```

```

35 #
36 while temppp < putpr:
37     pidv = pidv + 0.00001
38     d1 = math.log(stopr/strike)+((rfir/365)+(pidv**2)/2)*days
39     durvol = pidv*math.sqrt(days)
40     cumd1 = csnd(-d1/durvol)
41     cumd2 = csnd(-(d1/durvol - durvol))
42     discount = math.exp(-rfir*days/365)
43     temppp = -(stopr*cumd1)+(strike*discount*cumd2)
44 #

```




what the VIX tells us about volatility ...

The VIX index measures short-term volatility on SPX, the S&P 500. It is not measured the same as we measure implied volatility of course. Across multiple strikes and near-term expiries for SPX options, it measures time-adjusted spreads for symmetric calls and puts .. the narrower the spread the lower the VIX. But it will be correlated with implied volatility of SPY, and all of this implies that short-run volatility is highly variable!

The BS/DE and what it says about the importance of the Greeks in determining an options price (when $r = 0$): **DWBH ...**

Theta

$$\Theta + P_0 \Delta + \frac{1}{2} \sigma^2 P_0^2 \Gamma = 0$$

The role played by:

Time (points to Θ)

Delta (points to $P_0 \Delta$)

Stock Price (points to P_0)

Volatility (points to σ^2)

Gamma (points to Γ)

When r is positive: see Hull 17.4.

$$\Theta + rP_0\Delta + \frac{1}{2}\sigma^2 P_0^2\Gamma = r\Pi$$

The Delta and Gamma

$$\Delta = \frac{\partial P_c}{\partial P_s}$$

$$\Gamma = \frac{\partial^2 P_c}{\partial P_s^2}$$

The delta (**important**) is defined to be the spot change in the price of a call (or put) relative to the change in the price of the underlying stock.

The delta is not a constant. For a call option, its mapping will loosely mapping on the next slide. For that reason we may also want to define the gamma, which measures the rate of change of the delta. Generally, the delta is close to zero for a very distant OTM strike, approximately 0.5 when ATM, then converges to 1.0 when deep ITM.

This implies that gamma is positive and falling when the option is OTM and drawing close to ATM.

Note: The mapping to the left of SP might be quasi-concave, with a rising then falling gamma.

Calculation of the Delta and Gamma

$$\Delta_c = N(d_1)$$

$$\Delta_p = N(-d_1)$$

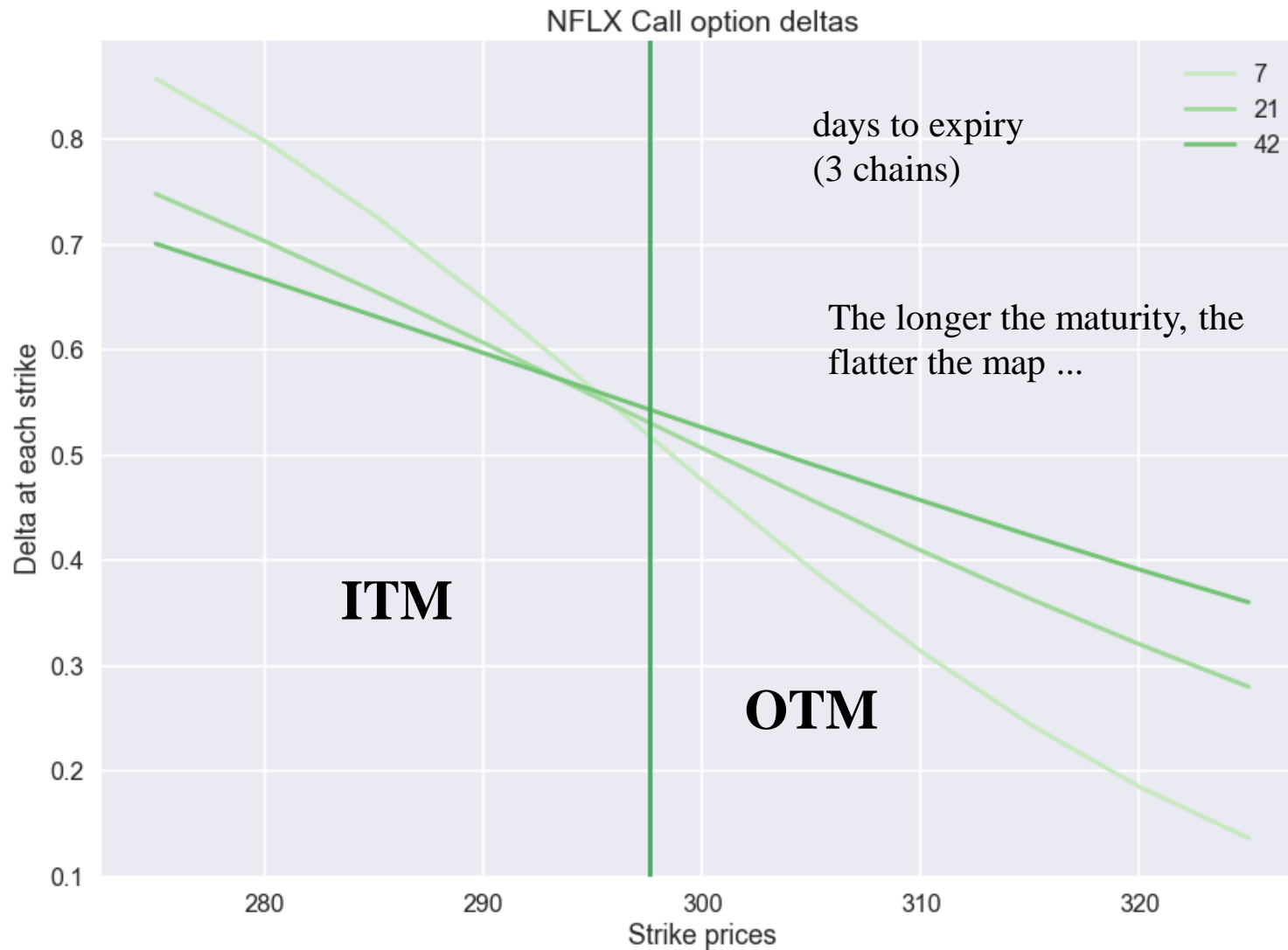
DWBH: We don't care about the formula for the gamma – just program it.

$$\Gamma = \frac{\left(\frac{1}{\sqrt{2\pi}} e^{-d_1^2/2} \right)}{P_0 \sigma \sqrt{T}}$$

See chapter 17 of Hull.

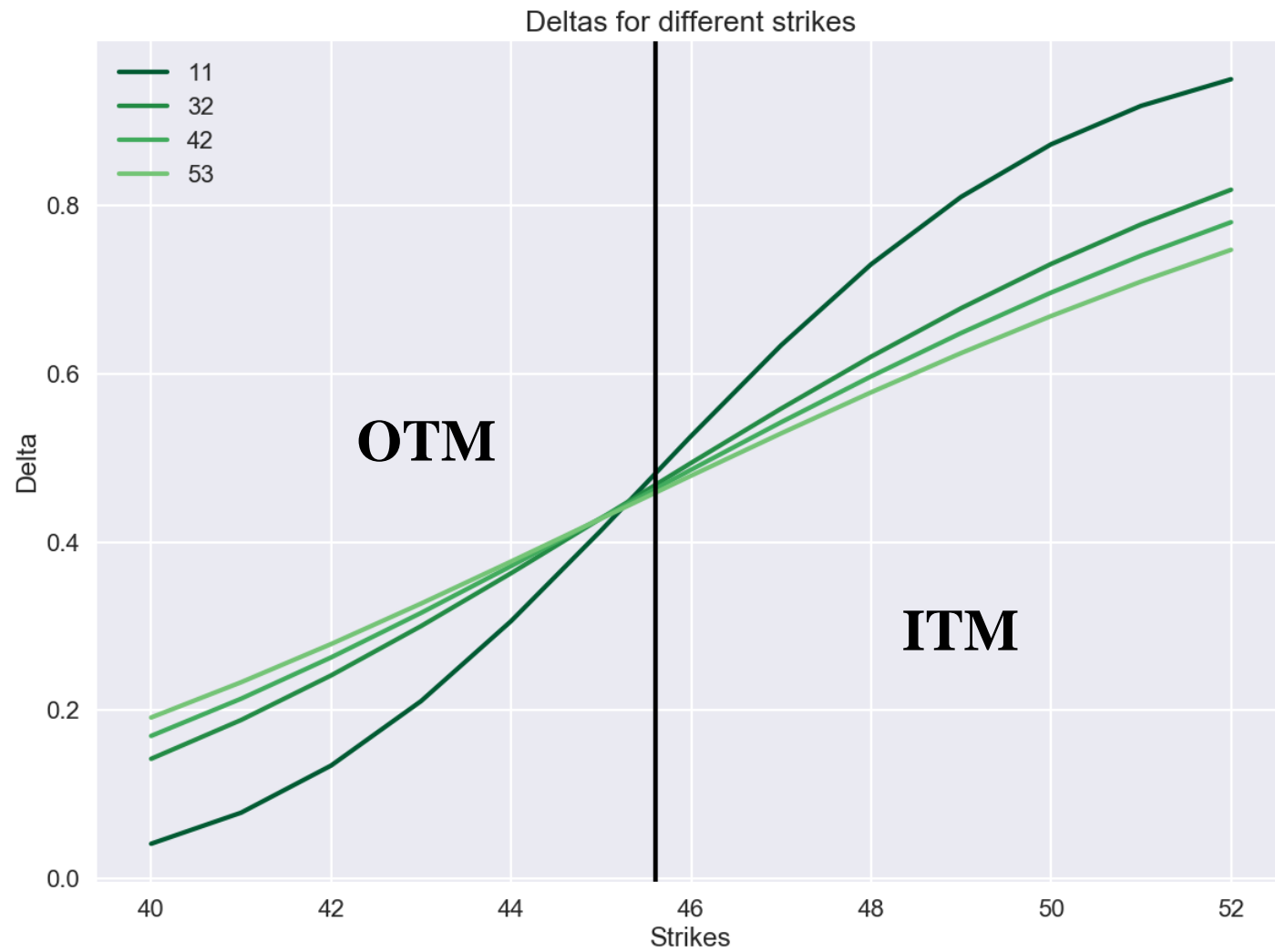
The Gamma formula treats time as one year so would have to be converted if using daily volatility.

N is the value of the cumulative standard normal density function up to d and d_1 and d_2 have the same meaning as in previous lectures.



call_delta_map.py

... calculating call delta mappings



put delta mappings

... how we did this

```

45 # Calculating the BSM call option price, traditional model. This requires the
46 # user to provide stock price, strike price, daily volatility, risk-free interest
47 # rate and days to expiry. (To calculate days use method daysto above).
48 # This returns the call price, the delta, and duration volatility as a tuple
49 # array. See popo below for puts.
50 #
51 def copo(stock,strike,dayvol,days,rfir):
52     d1 = math.log(stock/strike)+((rfir/365)+(dayvol**2)/2)*days
53     durvol = dayvol*math.sqrt(days)
54     delta = csnd(d1/durvol)
55     cumd2 = csnd((d1/durvol) - durvol)
56     discount = math.exp(-rfir*days/365)
57     callpr = (stock*delta)-(strike*discount*cumd2)
58     return [callpr,delta,durvol]
59 #

```

The top method copo is from our utility file. Note that it passes out a tuple.

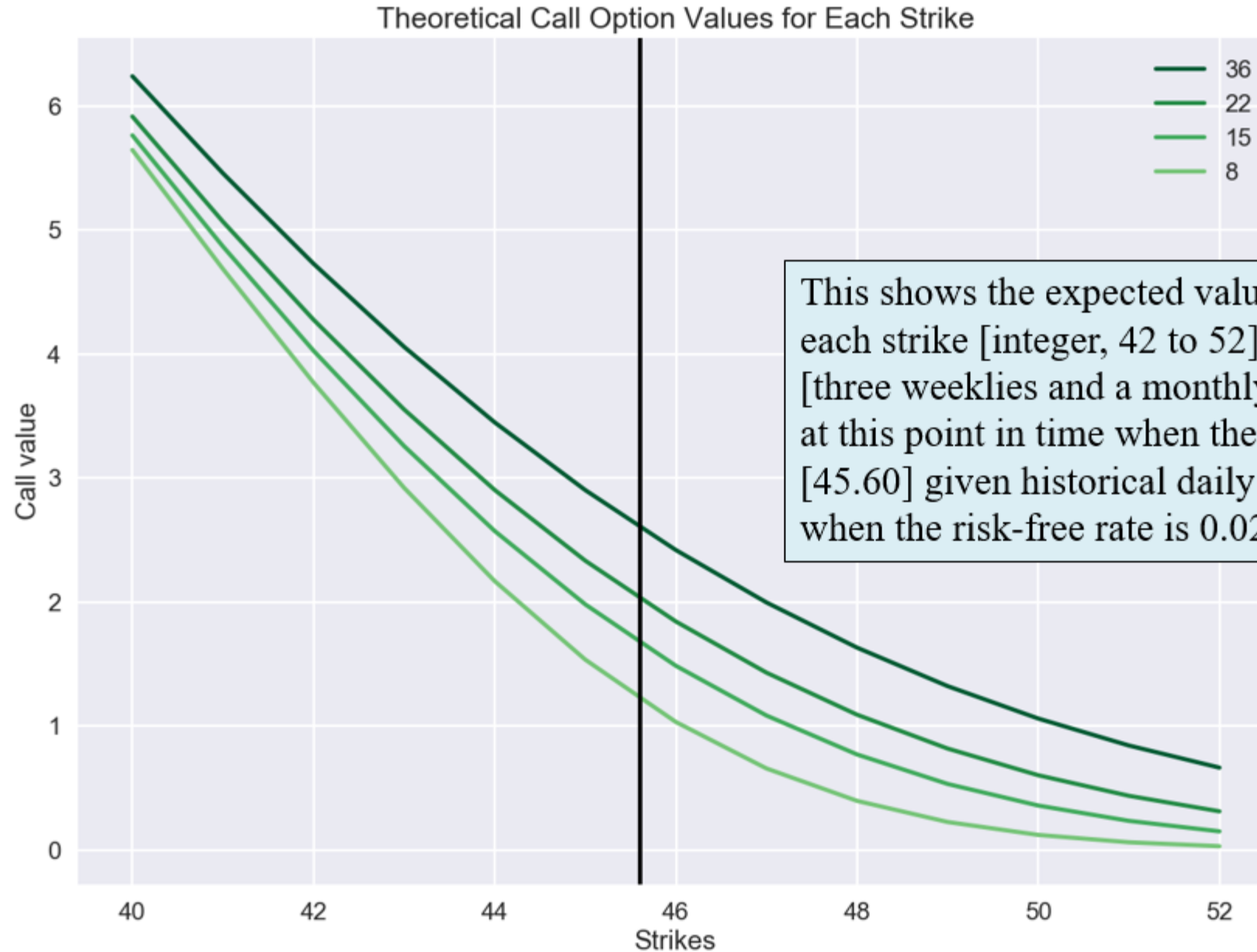
On the call_delta_map program below we call fu.copo over and over for the sensitivity mapping.

Here we call upon our core BSM call model (or similar) from our utility file.

```

In [4]: for i in range(0,num_strike): # 3
        for j in range(0,size): # 11
            c_value, c_delta, c_durvol = fu.copo(stopr,strike[j],dayvol,days[i],rfir)
            delta[i,j] = c_delta

```



Make sure you understand this slide. The use of the COPO and POPO programs allow us to do things like this ...

This shows the expected value of all call options at each strike [integer, 42 to 52] for four expiries [three weeklies and a monthly] for this stock [AA] at this point in time when the stock is at this price [45.60] given historical daily volatility [0.023434] when the risk-free rate is 0.0255.

The Theta

$$\Theta = \frac{\partial P_c}{\partial t}$$

The theta captures the sensitivity of the price of the option to elapsed time as the option approaches maturity. The theta more or less represents the "time decay" of an option.

Theta is represented as a negative number, because it is meant to measure how much the value of the option will decline if time passes *and nothing else happens*, such as a change in the value of the stock or a change in the stock's volatility.

If you look at Hull's equation 17.2. Theta for a call (see book for a put) can be expressed as

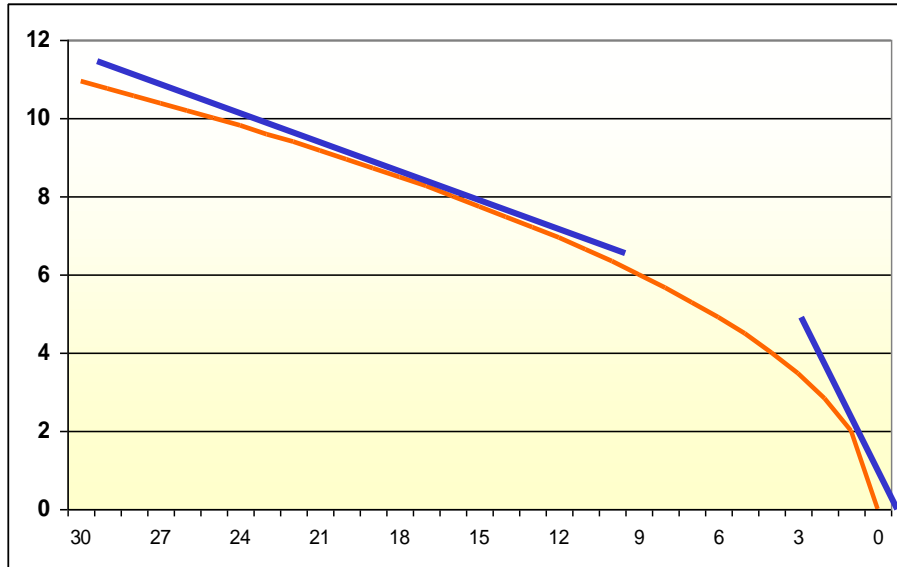
$$\Theta_c = \frac{P_0 \left(\frac{1}{\sqrt{2\pi}} e^{-d_1^2/2} \right) \sigma}{2\sqrt{t}} - rKe^{-rt} \times N(d_2)$$

This implies that the theta for a call option with a short duration will be greater (in absolute value) than one with a longer duration, i.e. more sensitive to the effect of one passing day if nothing changes.

DWBH: I am not going to require that you remember any of the BSMDE-derived greeks except delta. Just remember the concepts. We use sensitivity.

Theta and Time Decay

(atm call option)

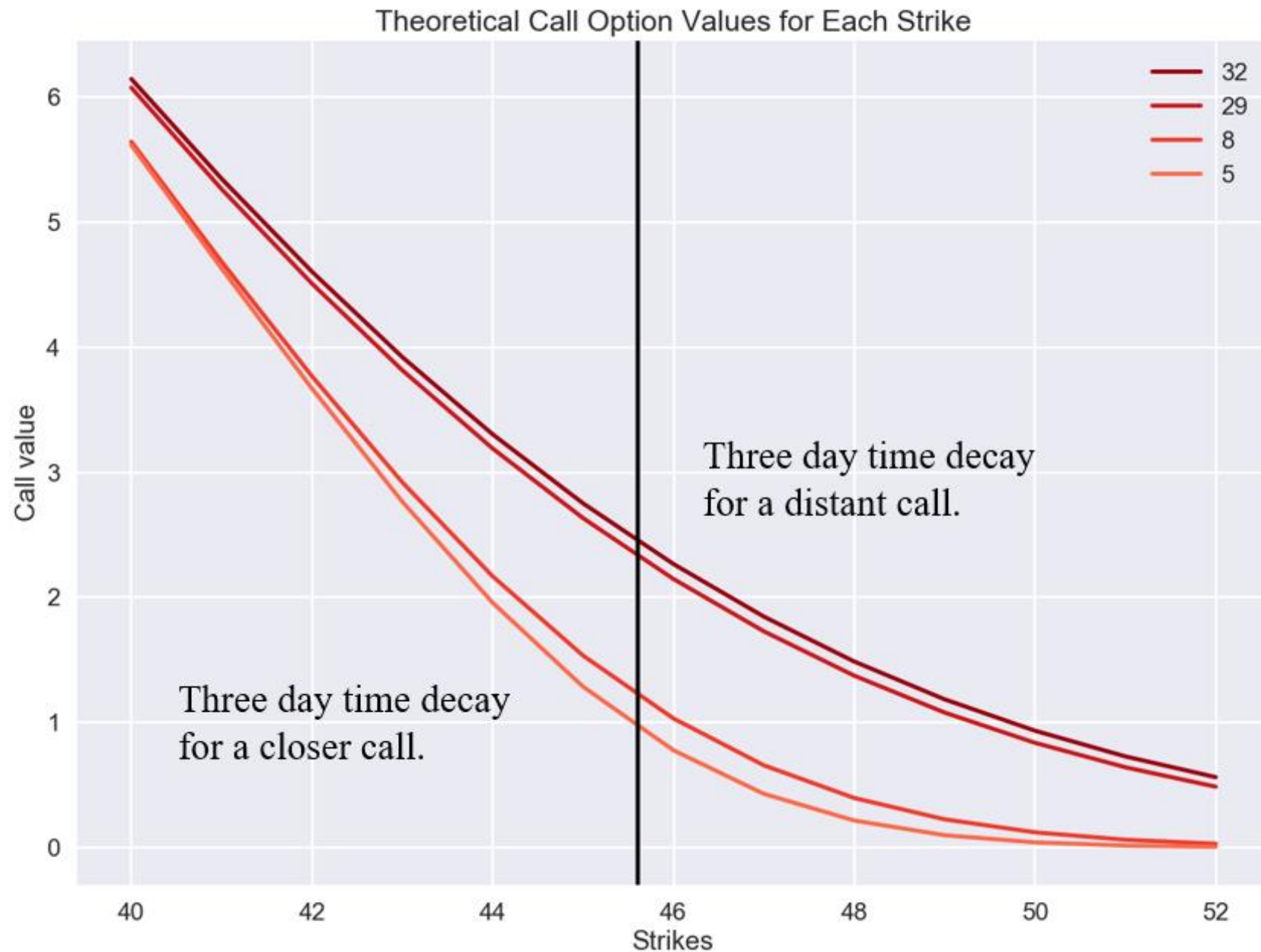


Theta is not time decay, it represents the rate of time decay. As such, the absolute value of theta rises, representing a quickly deteriorating time decay in the days just prior to expiration.

Shown here is the characteristic time decay of an atm call option that stays

atm throughout its life. The ordinate axis is a relative measure of the *time premium* of the option.

Generally theta (absolute value) rises slightly in the first 20 days, producing a near-linear decay. Then in the final week theta becomes much larger and the time decay become pronounced.



**Sensitivity time
decay ...**

The Vega

$$V = \frac{\partial P_c}{\partial \sigma_s}$$

The vega is the all-important sensitivity of the price of the option to the change in volatility. Clearly this will be an important positive number. It should be understood that the variance term represented here is daily volatility of the

stock, even though the relevant variance one must consider when entering an option is the volatility of the time period remaining in the option's life (which will equal this times the square root of time).

$$V = P_0 \sqrt{T} \left(\frac{1}{\sqrt{2\pi}} e^{-d_1^2/2} \right)$$

The Rho (interest rate sensitivity)

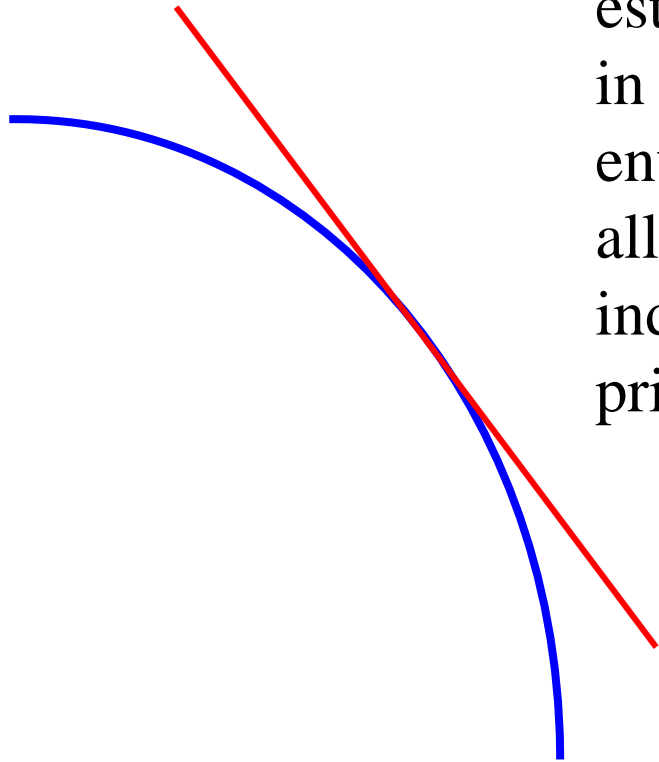
$$Rho_c = KTe^{-rT}N(d_2)$$

$$Rho_p = KTe^{-rT}N(-d_2)$$

This is not going to matter except in the case of extreme interest rate moves. I usually assume 0.01 for r and forget about it.

The real problem with the "classical" Greeks:

Using calculus, they are simply estimating the slope of a tangent in a very non-linear environment, so are biased for all but tiny moves in the independent variable (like stock price, volatility, etc.).

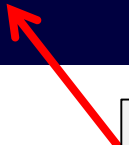


But then why bother ?

Once you have any kind of option or derivative pricing model set up with all of the key variables represented, then it is trivial to tweak with loops such that it can use **sensitivity analysis** to measure the response to

- any variable with small *or large* changes
- any complex of variables simultaneously

```
34 # Below we calculate implied daily volatility
35 #
36 while tempcp < callpr:
37     cipd = cipd + 0.00001
38     d1 = math.log(stopr/strike)+((rfir/365)+(cipd**2)/2)*days
39     durvol = cipd*math.sqrt(days)
40     cumd1 = csnd(d1/durvol)
41     cumd2 = csnd((d1/durvol) - durvol)
42     discount = math.exp(-rfir*days/365)
43     tempcp = (stopr*cumd1)-(strike*discount*cumd2)
44 #
45 # Below we calculate one day time decay using our new value for volatility
46 #
47 days = days - daystd
48 d1 = math.log(stopr/strike)+((rfir/365)+(cipd**2)/2)*days
49 durvol = cipd*math.sqrt(days)
50 cumd1 = csnd(d1/durvol)
51 cumd2 = csnd((d1/durvol) - durvol)
52 discount = math.exp(-rfir*days/365)
53 newcp = (stopr*cumd1)-(strike*discount*cumd2)
54 timedecay = callpr - newcp
55 #
```



Look at how callidv does this for time decay .. and note that it is using cipd!

Can you do the same (sensitivity analysis) for Delta and Vega?

Of course you can. But in the case of Delta and Vega, you must either do a mapping and you must specify by how much you want the stock price to move in the case of the delta, or volatility to move in the case of the Vega.

For example, it is easy to ask “What is the delta for a 10% (log) move in the price of the stock?”.

“What is the Vega for a rise in volatility (in relative terms) of 10%?”

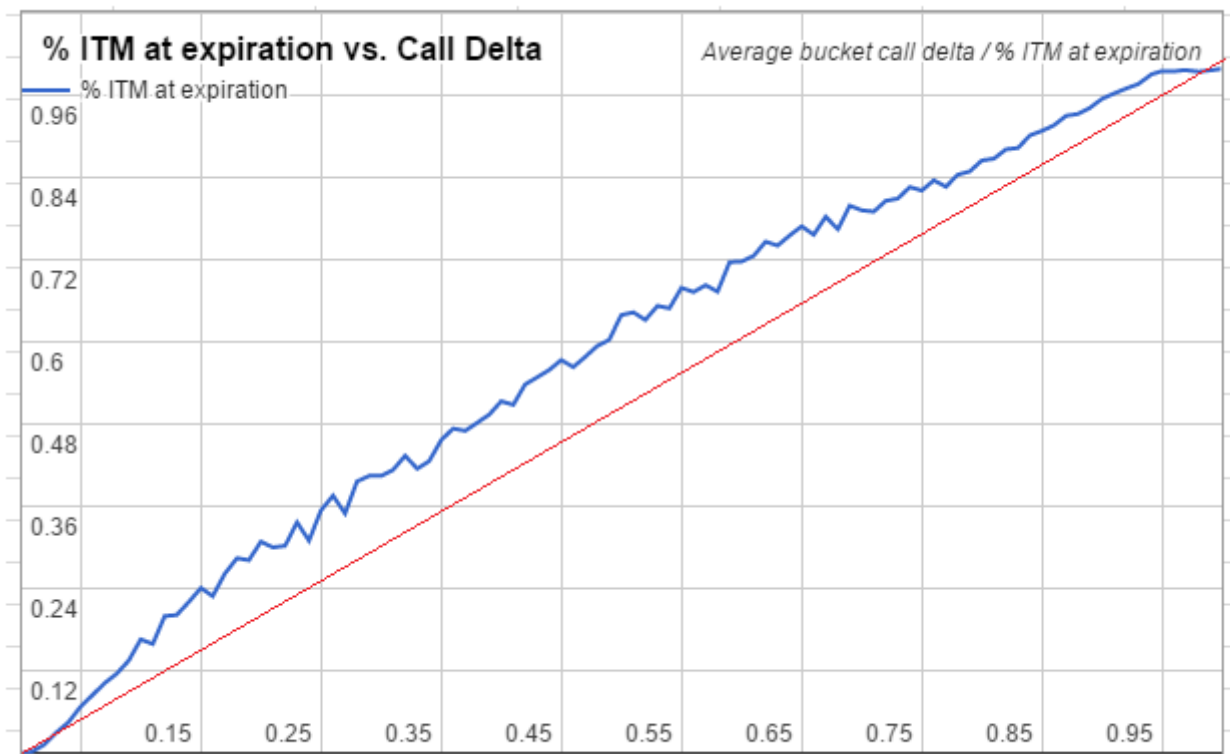
They are all designed just like the time decay estimator. Just use COPO or POPO over and over to get what you need.

<http://s0hungry.com/2017/01/06/option-delta-vs-actual-in-the-money/>

OPTION DELTA VS ACTUAL % IN-THE-MONEY

January 6, 2017  Rodrigo Salazar  0 Comment

I originally posted this on medium, but have since moved it here to consolidate my blog websites.



Each trial of this experiment goes like this:

- Choose a random instance in time in the last year (08/03/2015–07/29/2016).
- Choose any available SPY option **with 45 days of expiration or less** and record it's delta.
- Look ahead to the future data and record whether this option expires in the money.

I repeat this trial 1 million times for call options and again 1 million times for put options.

