Notebook for Hedging Calculations

See "Calculations on Rolling Hedges" by Steve Kimbrough, rolling-hedges.tex/pdf for development of the model for rolling hedges, which is implemented here.

Markdown reference: https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Working%20With%20Markdown%20Ce (https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Working%20With%20Markdown%20Ce

With:

Henry Hub spot prices: https://www.eia.gov/dnav/ng/hist/rngwhhdM.htmhttps://www.eia.gov/dnav/ng/hist/rngwhhdM.htr (https://www.eia.gov/dnav/ng/hist/rngwhhdM.htmhttps://www.eia.gov/dnav/ng/hist/rngwhhdM.ht

EIA natural gas prices: https://www.eia.gov/dnav/ng/ng_pri_sum_dcu_nus_m.htm (https://www.eia.gov/dnav/ng/ng_pri_sum_dcu_nus_m.htm)

EIA gas prices for utilities: https://www.eia.gov/dnav/ng/hist/n3045us3m.htm (https://www.eia.gov/dnav/ng/hist/n3045us3m.htm)

EIA Henry Hub prices https://www.eia.gov/dnav/ng/hist/n3045us3m.htm (https://www.eia.gov/dnav/ng/hist/n3045us3m.htm)

Henry Hub futures: https://www.cmegroup.com/trading/energy/natural-gas/natural-gas.html (https://www.cmegroup.com/trading/energy/natural-gas/natural-gas.html)

In [1]:
```
import math
```

Capital Recovery Function

A(nnual) payments given P(resent) outlay (for n years at interest rate r)

In [2]:
```
def AgivenP(K, r, n):
    '''
    Given a present outlay of capital (debt) K, an interest rate
    or hurdle rate, r, and a period of n years to repay the
    debt, returns the annual payments. 5% interest means r = 0.05.
    '''
    numer = r * (1 + r) ** n
    denom = (1 + r) ** n - 1
    return K * numer / denom
```

In [3]:
```
def crf(r,n):
    '''
    Capital recovery factor. See AgivenP, above.

    '''
    numer = r * (1 + r) ** n
    denom = (1 + r) ** n - 1
    return numer / denom
```

Hedging Setup

l lifetime of the project

n periods, assuming monthly duration n = 12*l

r annual interest rate

S_0 spot price at time 0 (S(0) when typeset)

F_t futures price at time $t - 1$

With simplifications (see the paper), the initial outlay is $nF(1)$.

In [4]:
```
#### Actually used:

i = 120 # test period index
b = 0.02/12 # monthly interest rate on money in brokerage account
c = 0.06/12 # firm's weighted cost of capital
F_1 = 2.67 # Initial futures price at t = 0.
n = 240 # months in 20 years
print(n,b,c, F_1)
```

```
240 0.0016666666666666668 0.005 2.67
```

Comparing Henry Hub and Utility Prices

Electric utilities get a deal on natural gas, but it is still higher than the Henry Hub price, for obvious reasons.

In June 2019, the Henry Hub price was \$2.40 per million Btu and the utility price was \$2.67. As a percentage the premium is

In [5]:
```
premium = 2.67/2.40
print(premium)
```

```
1.1125
```

2019-09-03

See my lab notebook for today, page 8

Background

We want to hedge (natural gas) prices for all of n periods (assuming months) going forward.

First calculations for a single line, i.

In [6]:
```
def closingAmount(i,F,b):
     return F*math.e**(b*i) - F
closingAmt = closingAmount(i,F_1,b)
print(closingAmt)
```

```
0.5911453642876534
```

In [7]:
```
def opportunityCost(i,F,c):
    return F_1*math.e**(c*i) - F_1

oppCost = opportunityCost(i,F_1,c)
print(oppCost)
```

```
2.195057197042659
```

In [8]:
```
def realPrice(i,F,b,c):
    return opportunityCost(i,F,c) - closingAmount(i,F,b)
realPrice_i = realPrice(i,F_1,b,c)

print(realPrice_i)
```

```
1.6039118327550055
```

In [9]:
```
def PV(i,F,b,c):
    return realPrice(i,F,b,c)/math.e**(c*i)

PV_i = PV(i,F_1,b,c)
print(PV_i)
```

```
0.8802454770848431
```

In [10]:
```
def NPV(i,F,b,c):
    return F + PV(i,F,b,c)

NPV_i = NPV(i,F_1,b,c)
print(NPV_i)
```

```
3.550245477084843
```

In [11]:
```
def NPVS(i,F,b,c,n):
    toReturn = 0
    for i in range(1,n+1):
        toReturn += NPV(i,F,b,c)
    return toReturn

SystemNPV = NPVS(i,F_1,b,c,n)
```

In [12]:
```
SystemNPV # The net present value of all outlays for the hedge
```

Out[12]:
```
841.2472376765215
```

In [13]:
```
def monthlyUnitCost(i,F,b,c,n):
    '''
    = SystemNPV*crf
    '''
    return NPVS(i,F,b,c,n)*crf(c,n)
monthUnitCost = monthlyUnitCost(i,F_1,b,c,n)
```

In [14]:
```
monthUnitCost
```

Out[14]:
```
6.026956489304293
```

Using current futures prices

Henry Hub futures: [https://www.cmegroup.com/trading/energy/natural-gas/natural-gas.html (https://www.cmegroup.com/trading/energy/natural-gas/natural-gas.html)](https://www.cmegroup.com/trading/energy/natural-gas/natural-gas.html)

October 2019 is 2.419

In [15]:
```
F_Oct = 2.419
```

In [16]:
```
daUnitCost = monthlyUnitCost(i,F_Oct,b,c,n)
print(daUnitCost)
```

```
5.641258912991038
```

In [18]:
```
levelizedCostOfNaturalGas = 5.64*7.6 # 7.6 is the heat rate in
#mmBtu/MWh of
# a CCGT plant with typical or average efficiency today.
#LCONG then has to be
# compared with the LCOE from a VRE PPA.
# levelizedCostOfNaturalGas has dimensions dollars/MWh.

print(levelizedCostOfNaturalGas)
```

```
42.864
```

In [ ]:
```

```