Computational Problem Set #1 Solutions

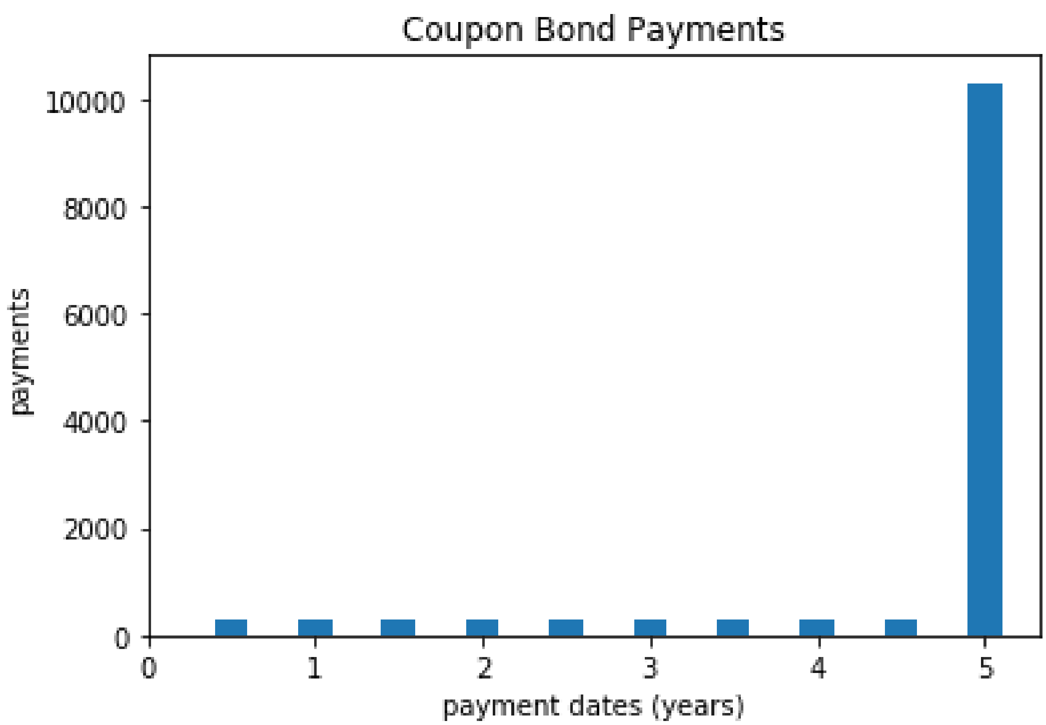
September 9, 2019

 # 1

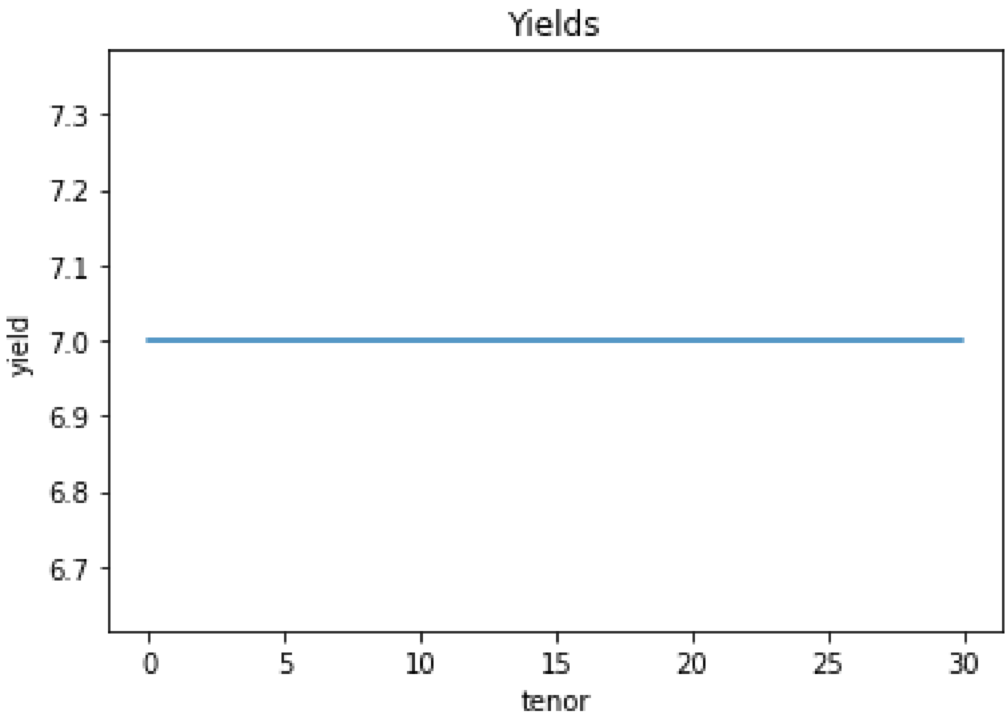
First, we create the bond and curve objects. Note that to instantiate a flat yield curve, we can use any positive maturity we want, but it will be most convenient choose a maturity greater than that of any bond we will be pricing, to avoid getting warning messages about going out of range.

```
In [1]: import fixedincome as fi
        import numpy as np
        import matplotlib.pyplot as plt
```

```
In [2]: bond1 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=6, frequency=2)
        bond1.plot_payments()
```



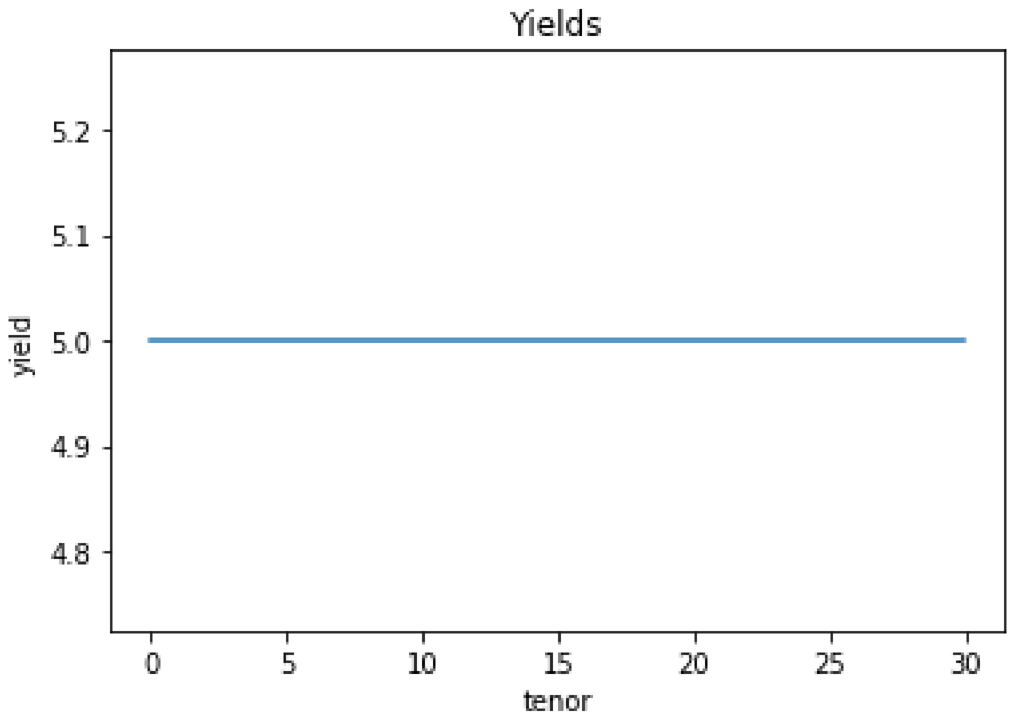

```
In [3]: yc = fi.curve_factory(dates=[30], rates=[7.0])
        yc.plot_yields()
```

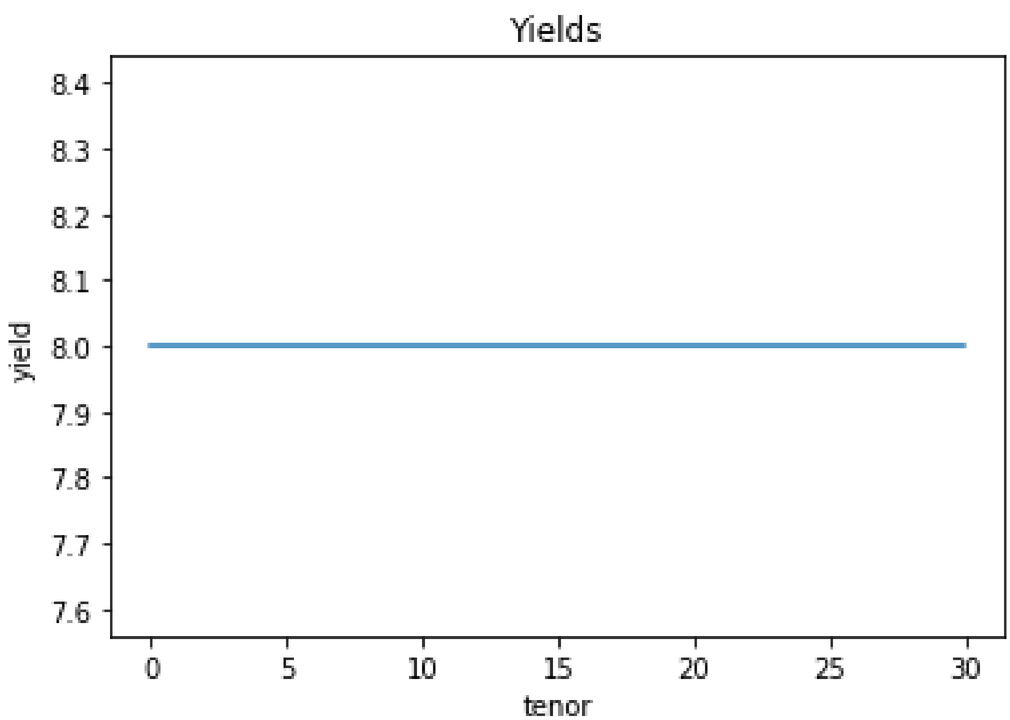


(a)

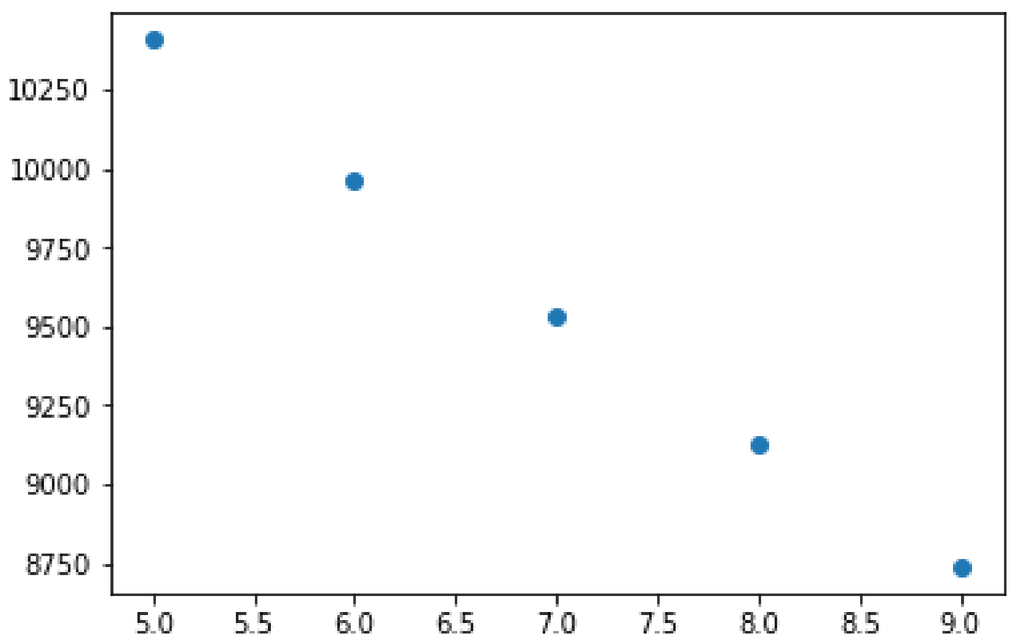
Starting from the basic yield curve we just instantiated, we use the shift method of the curve class to generate flat yield curves with constant interest rates of 5%, 6%, 7%, 8%, and 9%. We price the bond we have instantiated with each yield curve, and plot the prices as functions of the discountint rate.

```
In [4]: yc1 = yc.shift(-2.0)
        yc2 = yc.shift(-1.0)
        yc3 = yc
        yc4 = yc.shift(1.0)
        yc5 = yc.shift(2.0)
        yc1.plot_yields()
        yc4.plot_yields()
```


Yields
yield
5.2
5.1
5.0
4.9
4.8
0
5
10
15
20
25
30
tenor


Yields
yield
8.4
8.3
8.2
8.1
8.0
7.9
7.8
7.7
7.6
0
5
10
15
20
25
30
tenor

```
In [5]: price1 = bond1.price(yc1)
        price2 = bond1.price(yc2)
        price3 = bond1.price(yc3)
        price4 = bond1.price(yc4)
        price5 = bond1.price(yc5)

In [6]: prices = [price1, price2, price3, price4, price5]
        prices

Out[6]: [10409.356799388404,
         9961.317116521015,
         9534.087448559138,
         9126.677793157278,
         8738.145716053792]

In [7]: rates = [5.0, 6.0, 7.0, 8.0, 9.0]

In [8]: fig, ax = plt.subplots()
        ax.scatter(rates, prices)

Out[8]: <matplotlib.collections.PathCollection at 0x7f5c00d98198>
```



So, as the interest rate, the rate at which we are discounting the bond's coupons, goes up, the price goes down, as we would expect. Note that the appearence of the relationship being a straight line is an illusion, due to the fact that we are only displaying a short interval of discount rates. Over a longer interval of such rates, one would see that it is not a straight line.

(b)

In [9]:
```
bond2 = fi.create_coupon_bond(maturity=6, face=10000, rate=6, frequency=2)
bond3 = fi.create_coupon_bond(maturity=8, face=10000, rate=6, frequency=2)
bond4 = fi.create_coupon_bond(maturity=10, face=10000, rate=6, frequency=2)
bond5 = fi.create_coupon_bond(maturity=15, face=10000, rate=6, frequency=2)
bond6 = fi.create_coupon_bond(maturity=20, face=10000, rate=6, frequency=2)
bonds = [bond1, bond2, bond3, bond4, bond5, bond6]
```

In [10]:
```
bond_prices = [prices, [], [], [], [], []]
```

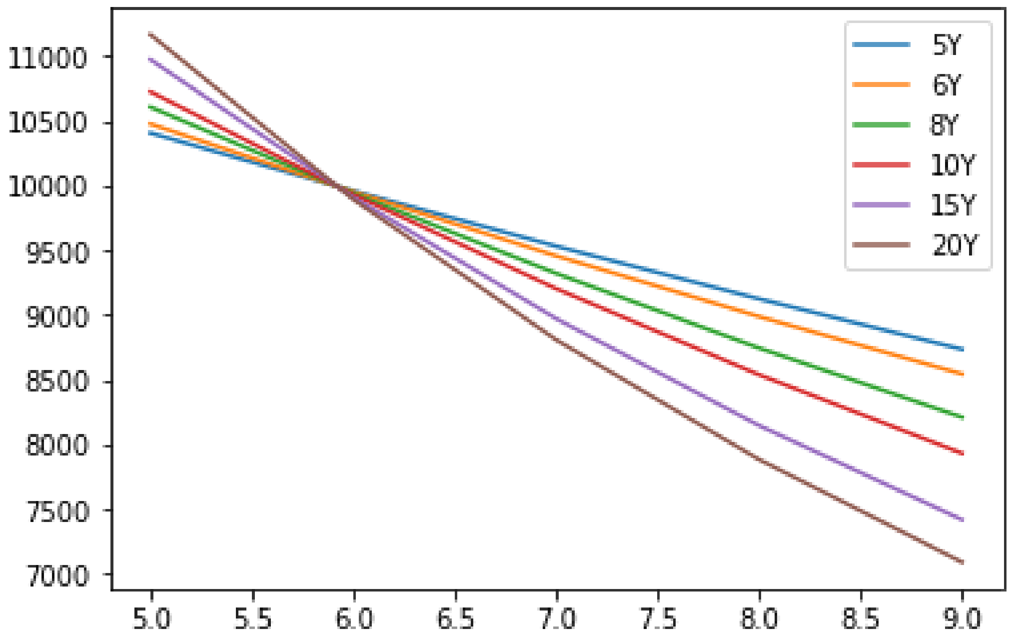
In [11]:
```
yield_curves = [yc1, yc2, yc3, yc4, yc5]
```

In [12]:
```
for i in range(1,6):
    for j in range(5):
        price = bonds[i].price(yield_curves[j])
        bond_prices[i].append(price)
```

In [13]:
```
bond_prices
```

Out[13]:
```
[[10409.356799388404,
  9961.317116521015,
  9534.087448559138,
  9126.677793157278,
  8738.145716053792],
 [10479.648278663539,
  9954.878188265033,
  9458.923986328464,
  8990.157194729078,
  8547.042471477045],
 [10610.113962658972,
  9943.103416938471,
  9323.498063770527,
  8747.797618684981,
  8212.764409152844],
 [10728.164195307694,
  9932.660131611272,
  9205.764422708304,
  8541.272411262518,
  7933.551901365156],
 [10976.451644898596,
  9911.430515040336,
  8974.399096810812,
  8148.863448721311,
  7420.52023449893],
 [11169.81810506679,
  9895.703228266499,
  8811.358707276962,
  7885.823854885893,
  7093.39680103158]]
```

```
In [14]: fig, ax = plt.subplots()
         ax.plot(rates, bond_prices[0], label='5Y')
         ax.plot(rates, bond_prices[1], label='6Y')
         ax.plot(rates, bond_prices[2], label='8Y')
         ax.plot(rates, bond_prices[3], label='10Y')
         ax.plot(rates, bond_prices[4], label='15Y')
         ax.plot(rates, bond_prices[5], label='20Y')
         ax.legend()

Out[14]: <matplotlib.legend.Legend at 0x7f5c00d16588>
```



The longer the bond maturity, the steeper is the slope of the price as a function of the discounting rate. This indicates that the longer maturity bonds are more sensitive to interest rate fluctuations than shorter maturity bonds. This is what one would expect. The longer maturity bonds have more coupons that need to be discounted, and so one would expect them to be more sensitive to interest rate fluctuations.

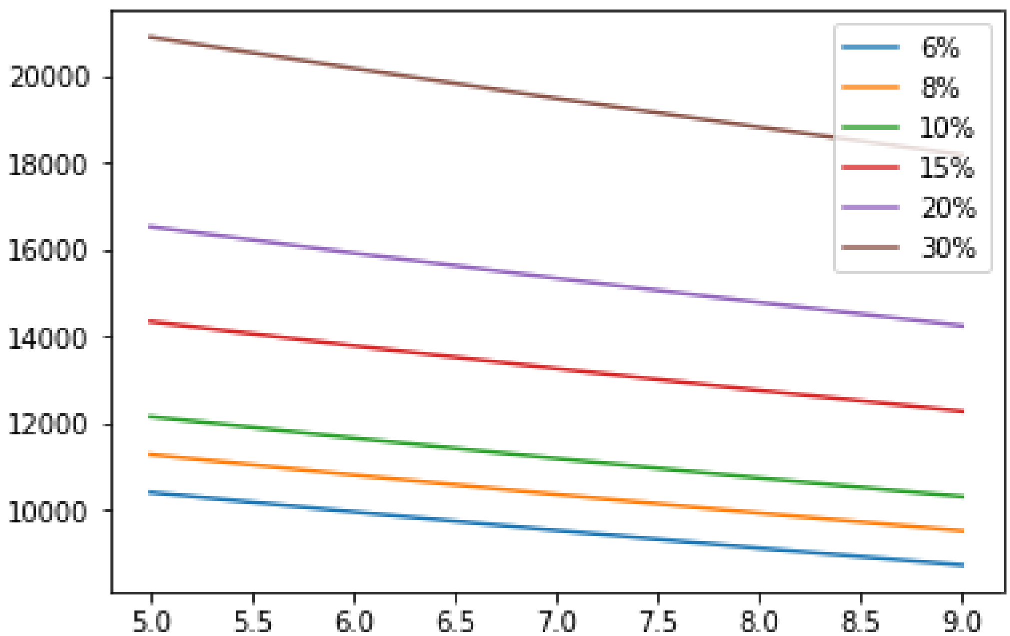
(c)

```
In [15]: bond11 = bond1
         bond12 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=8, frequency=2)
         bond13 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=10, frequency=2)
         bond14 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=15, frequency=2)
         bond15 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=20, frequency=2)
         bond16 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=30, frequency=2)
         bonds = [bond11, bond12, bond13, bond14, bond15, bond16]

In [16]: for i in range(6):
             bond_prices[i].clear()
```

```
        for j in range(5):
            price = bonds[i].price(yield_curves[j])
            bond_prices[i].append(price)
```

```
In [17]: fig, ax = plt.subplots()
         ax.plot(rates, bond_prices[0], label='6%')
         ax.plot(rates, bond_prices[1], label='8%')
         ax.plot(rates, bond_prices[2], label='10%')
         ax.plot(rates, bond_prices[3], label='15%')
         ax.plot(rates, bond_prices[4], label='20%')
         ax.plot(rates, bond_prices[5], label='30%')
         ax.legend()
```

```
Out[17]: <matplotlib.legend.Legend at 0x7f5c00ca6f60>
```



While it is no surprise that the bond price is an increasing function of the coupon value, a consequence of the seeming (again, it is not really true) constancy of the slope for the different bonds, is that in relative terms the higher coupon bonds are less sensitive to price fluctuations.

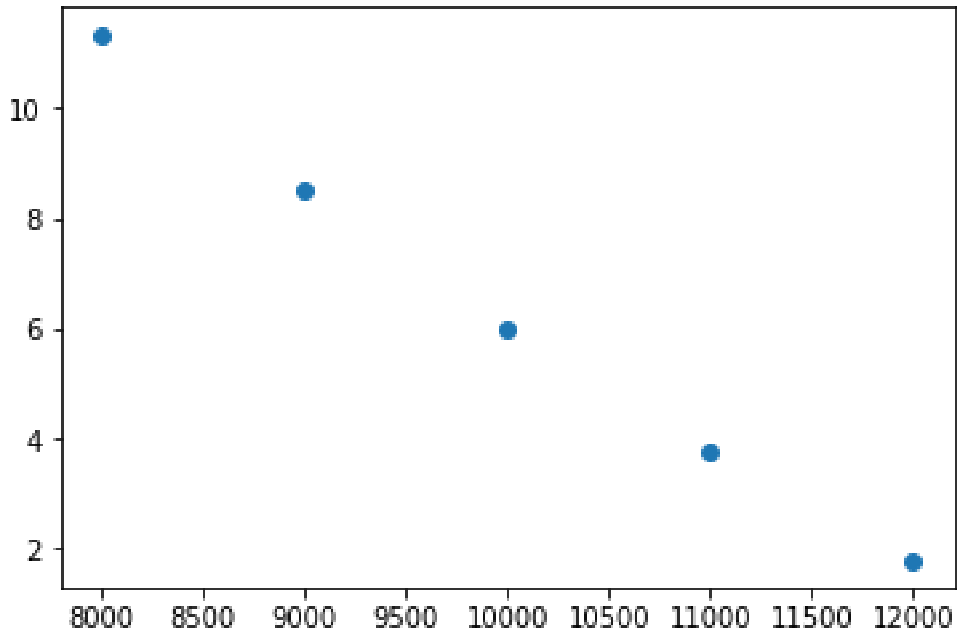
#2

```
In [18]: bond1 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=6, frequency=2)
```

```
In [19]: ytm1 = bond1.YTM(8000)
         ytm2 = bond1.YTM(9000)
         ytm3 = bond1.YTM(10000)
         ytm4 = bond1.YTM(11000)
         ytm5 = bond1.YTM(12000)
```

```
In [20]: ytm_list = [ytm1, ytm2, ytm3, ytm4, ytm5]
         prices = [8000, 9000, 10000, 11000, 12000]
         fig, ax = plt.subplots()
         ax.scatter(prices, ytm_list)
```

```
Out[20]: <matplotlib.collections.PathCollection at 0x7f5c00c030f0>
```



This is another example of the inverse relationship between prices and interest rates. Recall that the yield-to-maturity is that constant discount rate such that the given bond would have the asserted price. Or, more fundamentally, the higher the market bond price, the smaller must be the rate at which the coupons are being discounted, hence the inverse relationship depicted in the plot.

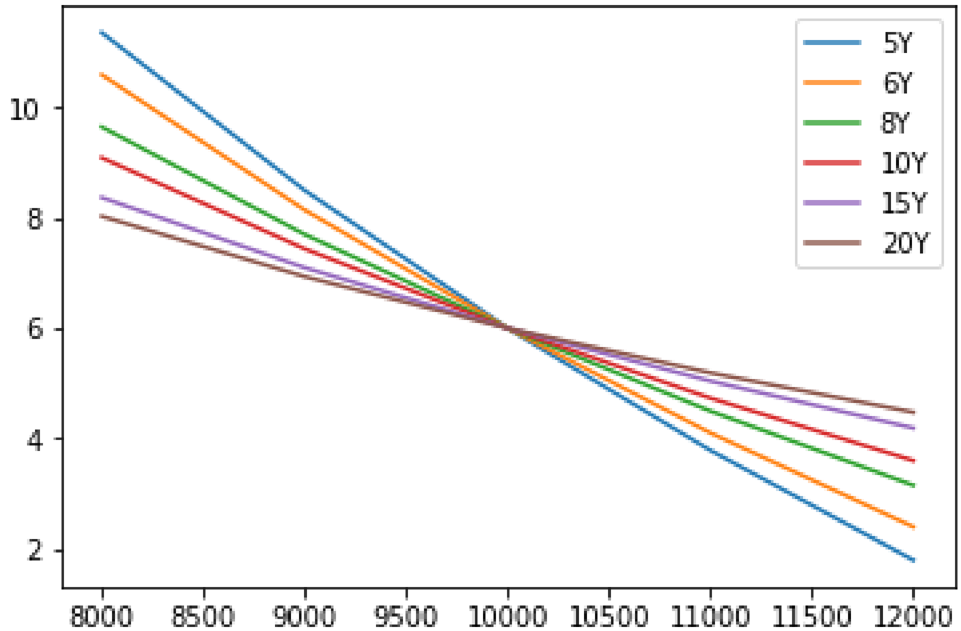
```
In [21]: bond2 = fi.create_coupon_bond(maturity=6, face=10000, rate=6, frequency=2)
         bond3 = fi.create_coupon_bond(maturity=8, face=10000, rate=6, frequency=2)
         bond4 = fi.create_coupon_bond(maturity=10, face=10000, rate=6, frequency=2)
         bond5 = fi.create_coupon_bond(maturity=15, face=10000, rate=6, frequency=2)
         bond6 = fi.create_coupon_bond(maturity=20, face=10000, rate=6, frequency=2)
         bonds = [bond1, bond2, bond3, bond4, bond5, bond6]
```

```
In [22]: ytm_list = [[], [], [], [], [], []]
         for i in range(6):
             for j in range(5):
                 ytm = bonds[i].YTM(prices[j])
                 ytm_list[i].append(ytm)
```

```
In [23]: fig, ax = plt.subplots()
         ax.plot(prices, ytm_list[0], label='5Y')
```

```
ax.plot(prices, ytm_list[1], label='6Y')
ax.plot(prices, ytm_list[2], label='8Y')
ax.plot(prices, ytm_list[3], label='10Y')
ax.plot(prices, ytm_list[4], label='15Y')
ax.plot(prices, ytm_list[5], label='20Y')
ax.legend()
```

Out[23]: <matplotlib.legend.Legend at 0x7f5c00b90470>



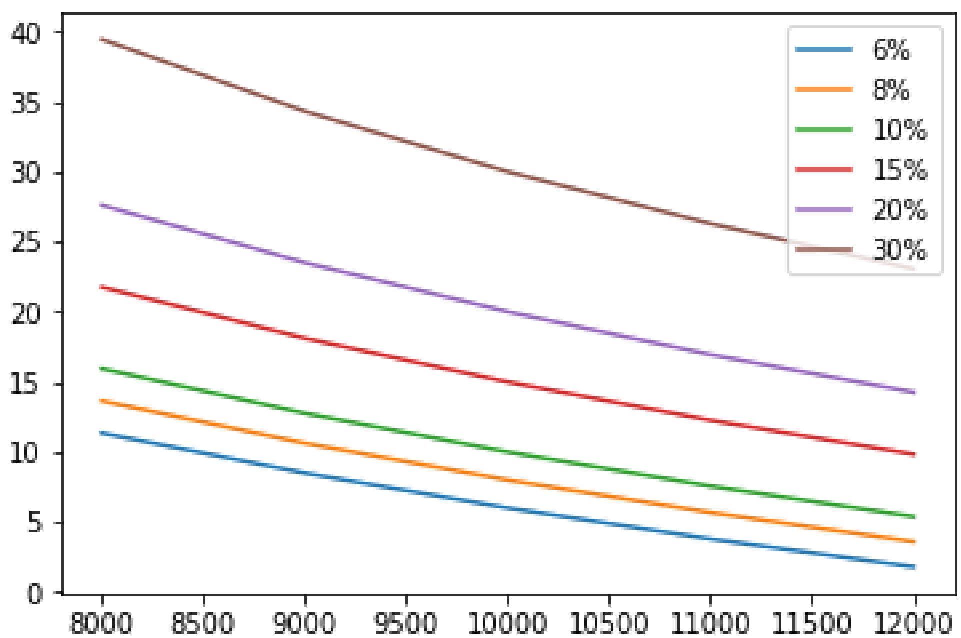
This plot exhibits the inverse of the greater sensitivity of prices to changes in rates for longer maturity bonds. The converse of that observation is that for higher maturity bonds, a given change in the price implies a smaller change in the yield-to-maturity.

In [24]:
```
bond11 = bond1
bond12 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=8, frequency=2)
bond13 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=10, frequency=2)
bond14 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=15, frequency=2)
bond15 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=20, frequency=2)
bond16 = fi.create_coupon_bond(maturity=5.0, face=10000, rate=30, frequency=2)
bonds = [bond11, bond12, bond13, bond14, bond15, bond16]
```

In [25]:
```
for i in range(6):
    ytm_list[i].clear()
    for j in range(5):
        ytm = bonds[i].YTM(prices[j])
        ytm_list[i].append(ytm)
```

```
In [26]: fig, ax = plt.subplots()
         ax.plot(prices, ytm_list[0], label='6%')
         ax.plot(prices, ytm_list[1], label='8%')
         ax.plot(prices, ytm_list[2], label='10%')
         ax.plot(prices, ytm_list[3], label='15%')
         ax.plot(prices, ytm_list[4], label='20%')
         ax.plot(prices, ytm_list[5], label='30%')
         ax.legend()
```

```
Out[26]: <matplotlib.legend.Legend at 0x7f5c00b1dda0>
```



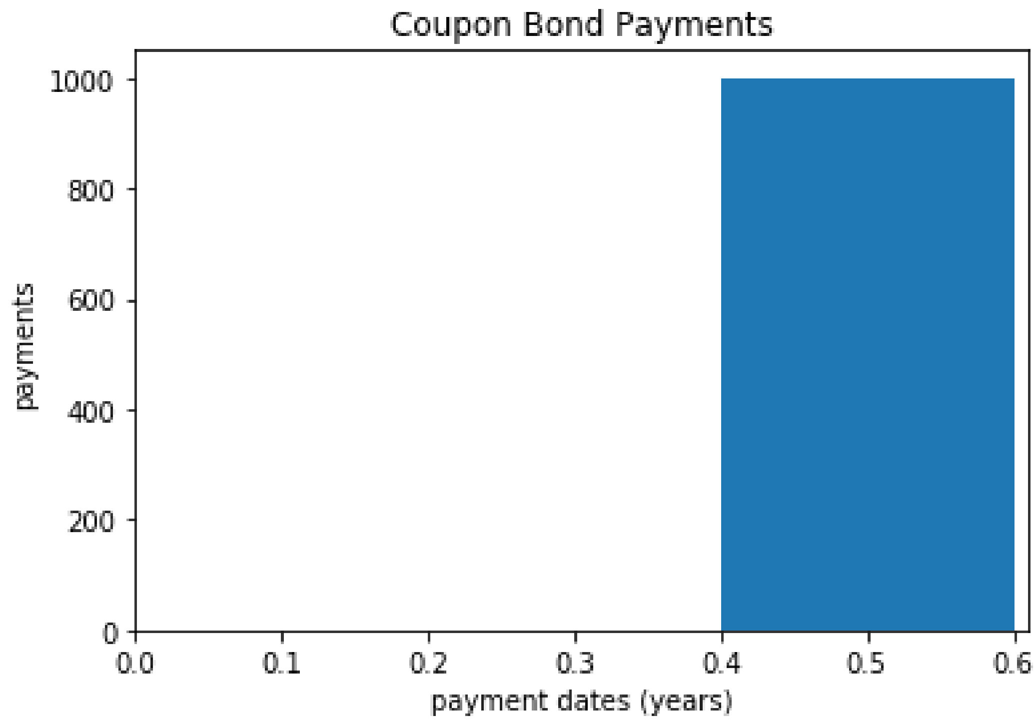
This graph represents the inverse of the relationships from the corresponding plot from problem # 1. The larger the coupon rate, the higher must be the discounting to bring the bond down to a given price level.

#3

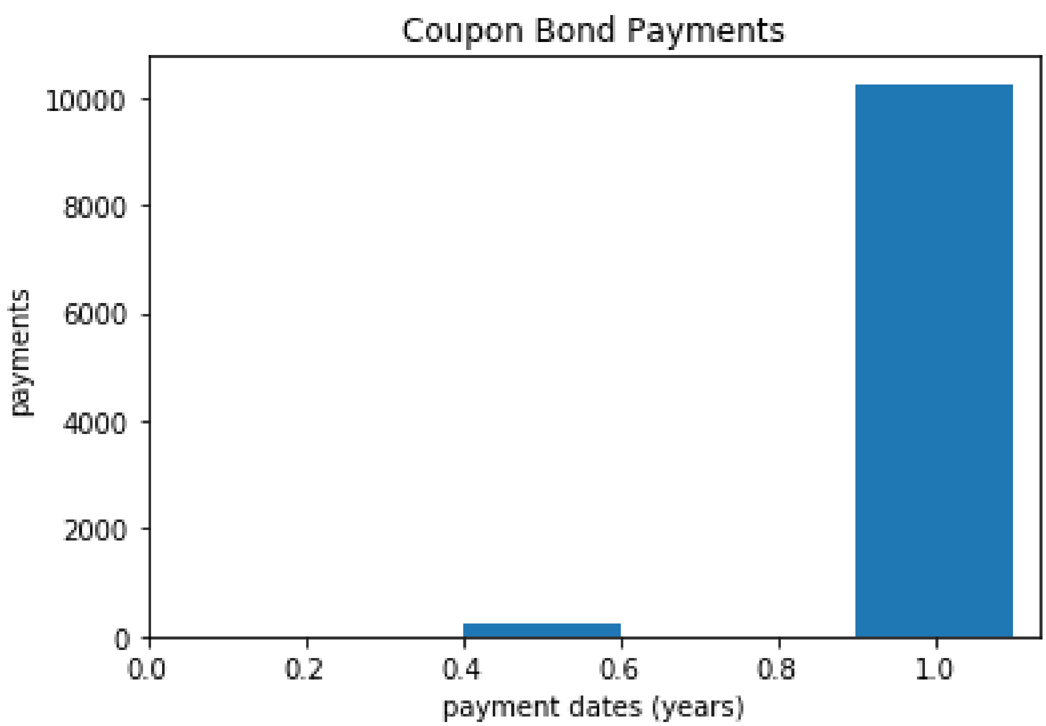
(a)

```
In [27]: bond1 = fi.bond_factory(dates=[0.5], rates=[1000])
         bond1.set_price(985)
         bond2 = fi.create_coupon_bond(maturity=1.0, face=10000, rate=5, frequency=2)
         bond2.set_price(10124)
         bond3 = fi.create_coupon_bond(maturity=2.0, face=10000, rate=7, frequency=2)
         bond3.set_price(10507)
```

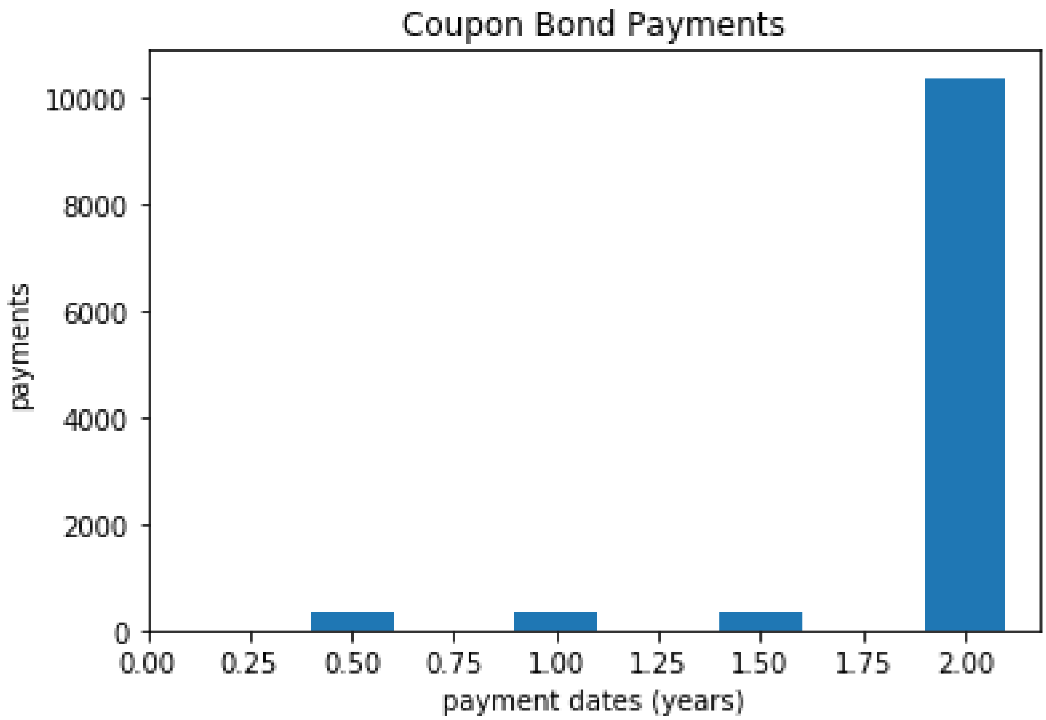
```
In [28]: bond1.plot_payments()
         bond2.plot_payments()
         bond3.plot_payments()
```



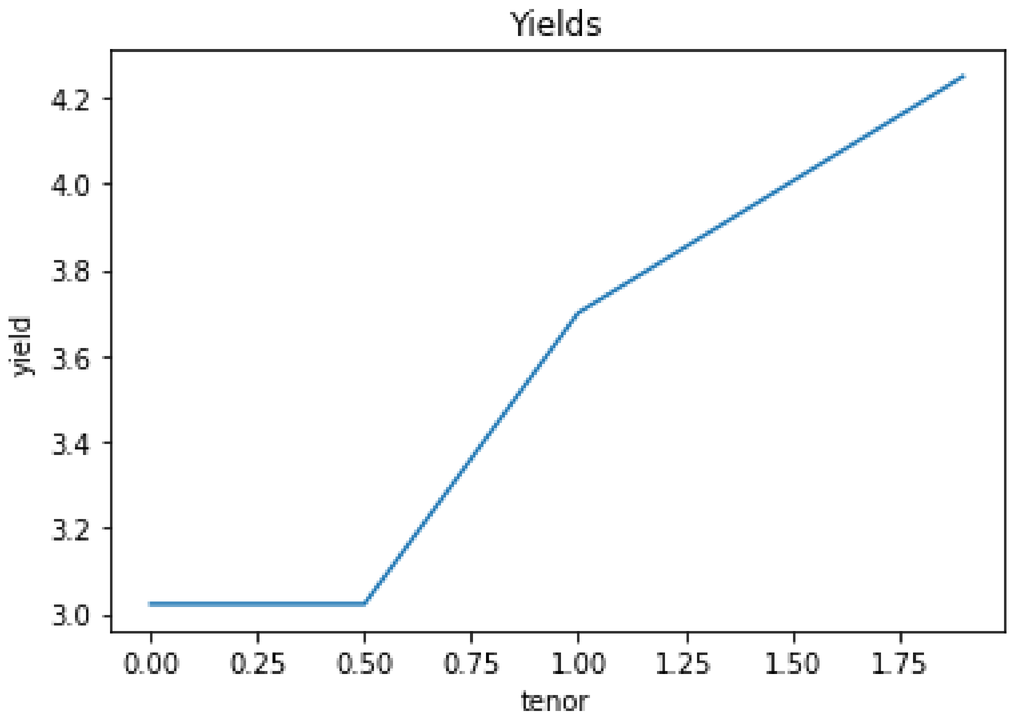
Coupon Bond Payments
1000
800
600
400
200
0
payments
0.0
0.1
0.2
0.3
0.4
0.5
0.6
payment dates (years)



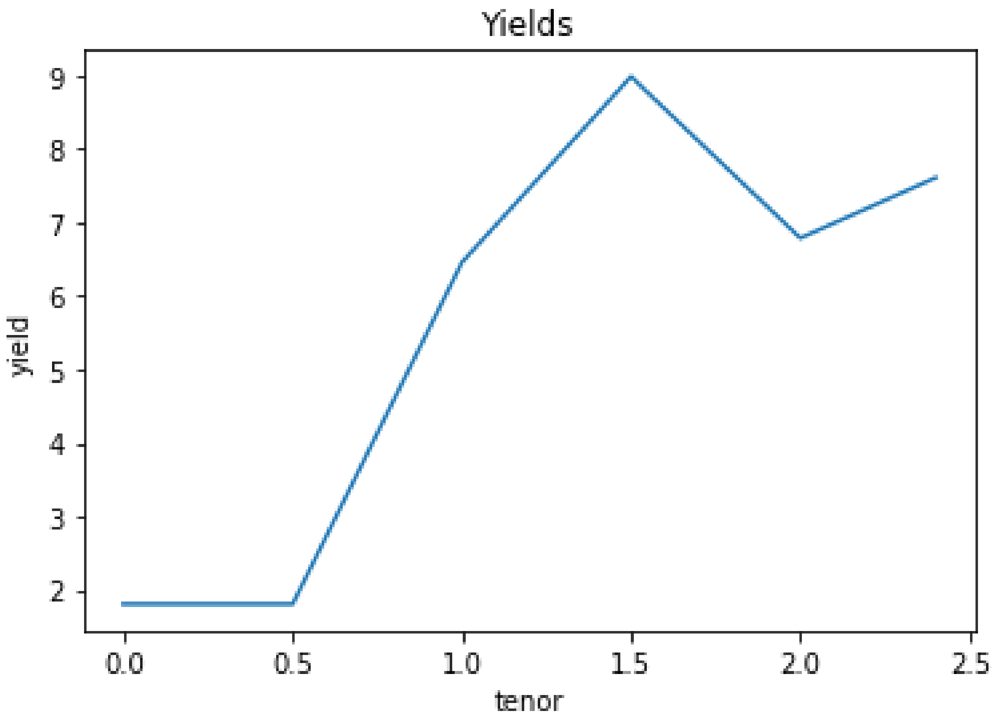
Coupon Bond Payments
10000
8000
6000
4000
2000
0
payments
0.0
0.2
0.4
0.6
0.8
1.0
payment dates (years)




```
In [29]: bonds = [bond1, bond2, bond3]
         yc = fi.curve_factory(bondlist=bonds)

In [30]: yc.plot_yields()
```




```
In [31]: bond11 = fi.bond_factory(dates=[0.5], rates=[10000])
         bond11.set_price(9910)
         bond12 = fi.create_coupon_bond(maturity=1.0, face=10000, rate=7, frequency=2)
         bond12.set_price(10050)
         bond13 = fi.create_coupon_bond(maturity=1.5, face=50000, rate=4, frequency=2)
         bond13.set_price(46500)
         bond14 = fi.create_coupon_bond(maturity=2.0, face=100000, rate=5, frequency=2)
         bond14.set_price(96500)
         bond15 = fi.create_coupon_bond(maturity=2.5, face=100000, rate=7, frequency=2)
         bond15.set_price(98000)

In [32]: bonds = [bond11, bond12, bond13, bond14, bond15]
         yc1 = fi.curve_factory(bondlist=bonds)

In [33]: yc1.plot_yields()
```



This is the same curve built in problem set # 1, although the interpolation algorithm used in the code is somewhat different, for this example the results will be the same. The non-monotonicty seen in the curve, ie the local minimum at the 2 year point, is an example of a yield curve inversion. An inverted yield curve is not a normal event, though it certainly happens, and is often a worrying sign for economic conditions. If you see an yield curve inversion like this when building a curve, it should give you pause, and you should be especially wary of your inputs. It could, for instance, indicate bad price data, and you should be especially careful. In this case, given the prices of the calibration instruments, the result is correct as the same result was derived when working the problem by hand in problem set # 1.

(b)

```
In [34]: bond1.price(yc)

Out[34]: 985.0

In [35]: bond2.price(yc)

Out[35]: 10124.000000000256

In [36]: bond3.price(yc)

Out[36]: 10506.999999999998

In [37]: bond11.price(yc1)

Out[37]: 9910.0
```

In [38]: bond12.price(yc1)

Out[38]: 10050.0

In [39]: bond13.price(yc1)

Out[39]: 46500.0

In [40]: bond14.price(yc1)

Out[40]: 96499.99999999999

In [41]: bond15.price(yc1)

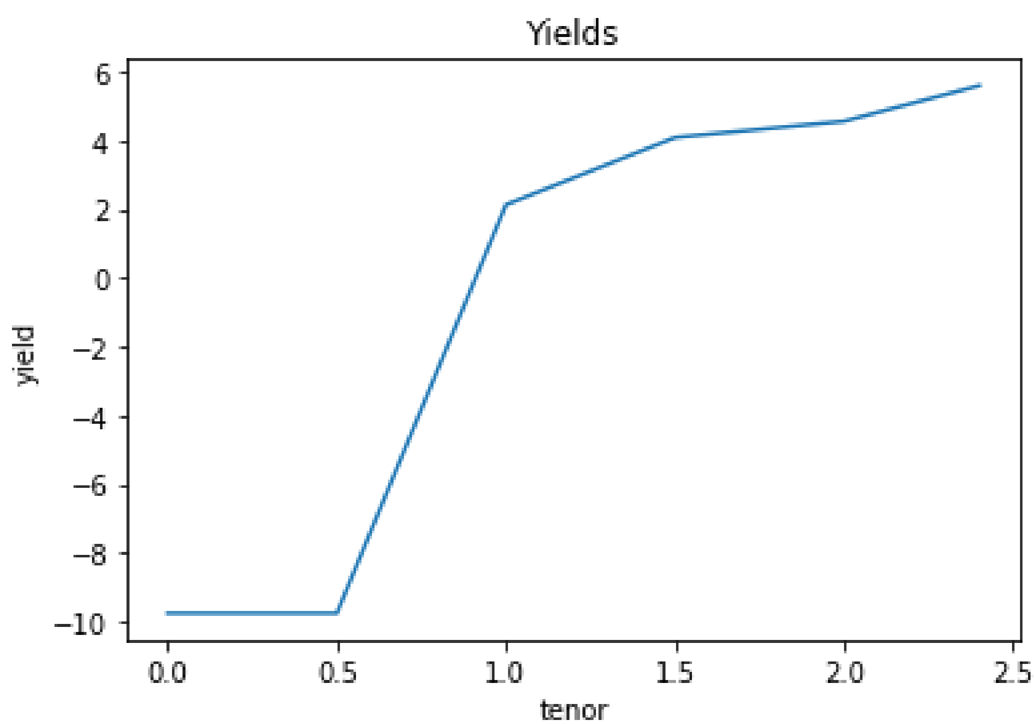
Out[41]: 98000.0

All the prices check out, and we have correctly reproduced the prices of our calibration instruments for both yield curves.

(c)

In [42]:
```
bond11.set_price(10500)
bond12.set_price(10500)
bond13.set_price(50000)
bond14.set_price(101000)
bond15.set_price(103000)
```

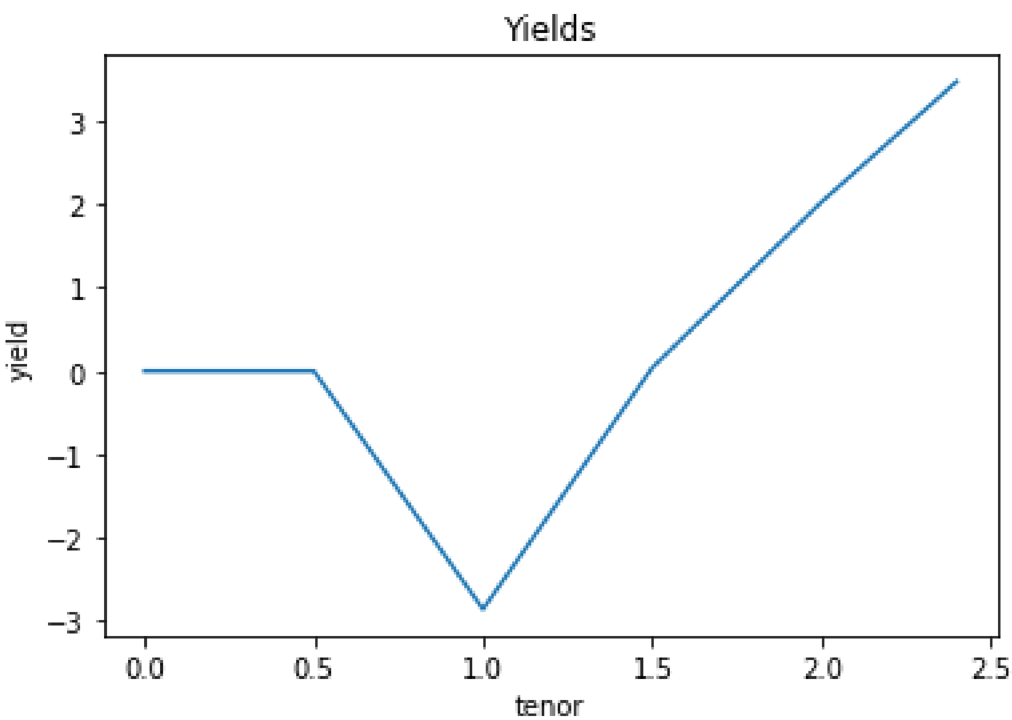
In [43]:
```
yc2 = fi.curve_factory(bondlist=bonds)
yc2.plot_yields()
```



Here we have essentially raised all the bond prices by a near constant increment. As a result, the implied yield curve has smaller yields (greater price implies less discounting which in turn implies lower rates). Why is the 6 month implied rate negative? Consider the 6 month bond and its price. It is a zero coupon bond with a $10,000 face value, but we have built this yield curve assuming it is trading at $10,500, even more than its face value, which is the only payment it makes. The only way to discount a payment and get an even larger monetary value than the payment being discounted is with a negative interest rate. Since we don't want to consider negative interest rates, from now on we will cap the price of this bond at $10,000. It should be noted, however, that there are no inversions in this curve so that, from some points of view, this might be a more reasonable or normal yield curve.

```
In [44]: bond11.set_price(10000)
         bond12.set_price(11000)
         bond13.set_price(53000)
         bond14.set_price(106000)
         bond15.set_price(108000)
```

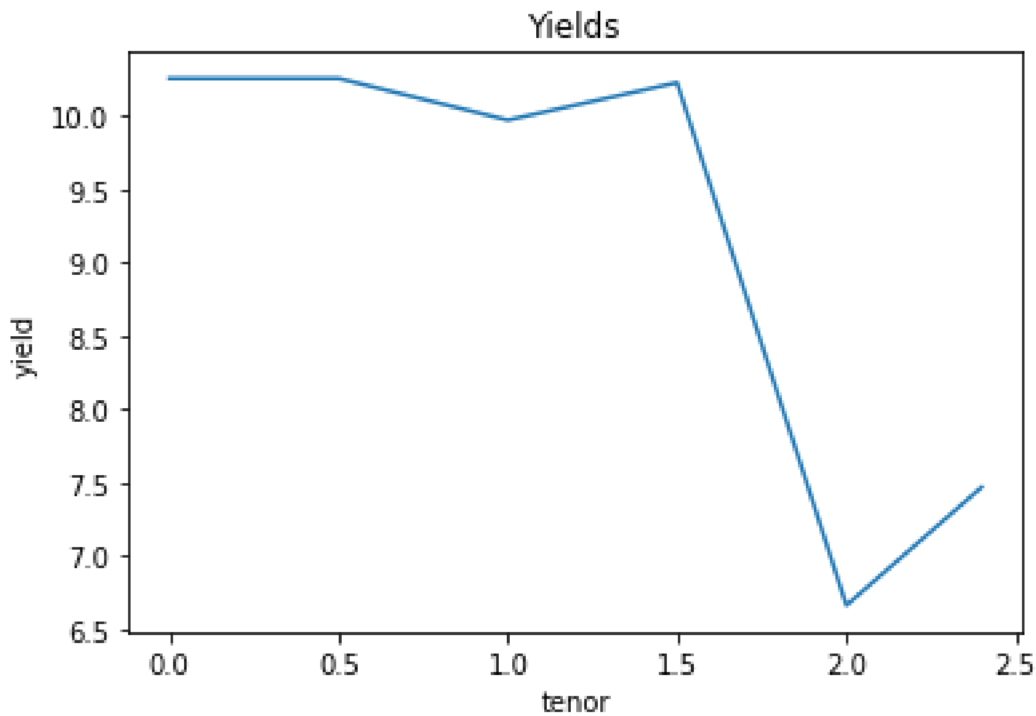
```
In [45]: yc3 = fi.curve_factory(bondlist=bonds)
         yc3.plot_yields()
```



Since the first bond, a zero coupon bond, is now trading at par, this implies a discount factor of 1, or equivalently, the 6 month rate is 0. With a market price of $11,000 for the second bond, which has a $10,000 face value and pays only 2 coupons of $700 the price of $11,000 is greater than the sum of the values of all payments, even without discounting, so this implies a negative 1 year rate.

```
In [46]: bond11.set_price(9500)
         bond12.set_price(9700)
         bond13.set_price(45600)
         bond14.set_price(96500)
         bond15.set_price(98000)
```

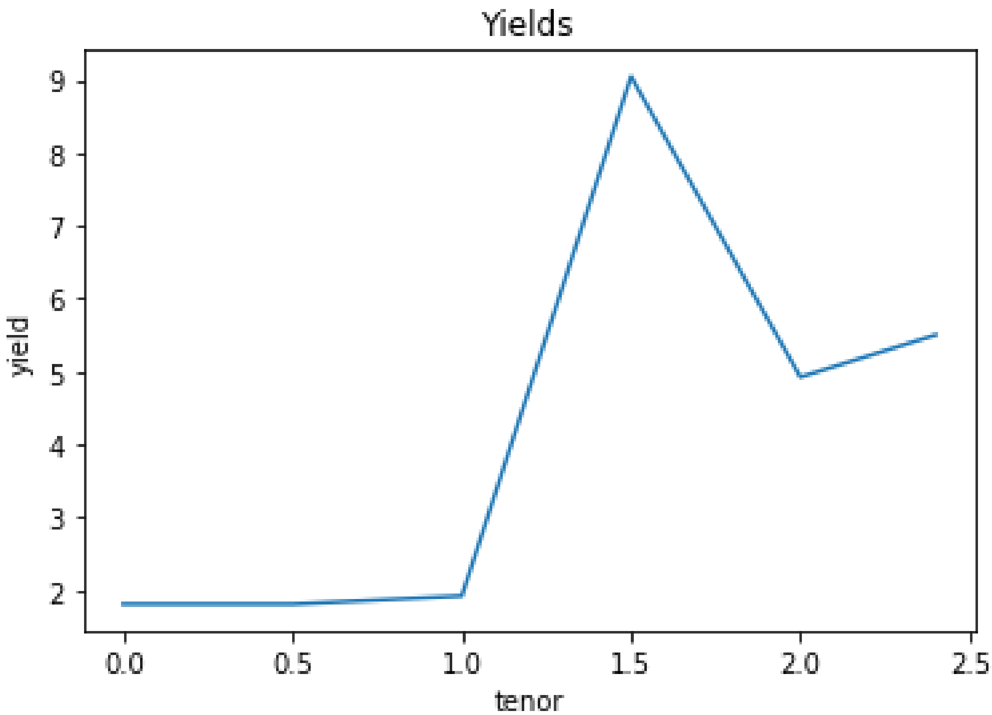
```
In [47]: yc4 = fi.curve_factory(bondlist=bonds)
         yc4.plot_yields()
```



The multiple inversions in this yield curve suggest that this realization of prices may not be the most realistic. There is massive discounting at the front end necessary because of how low the prices of the shorter maturity bonds are. Because of the significantly less distressed longer maturity bonds, the curve must compensate for this at the long end with significantly reduced rates.

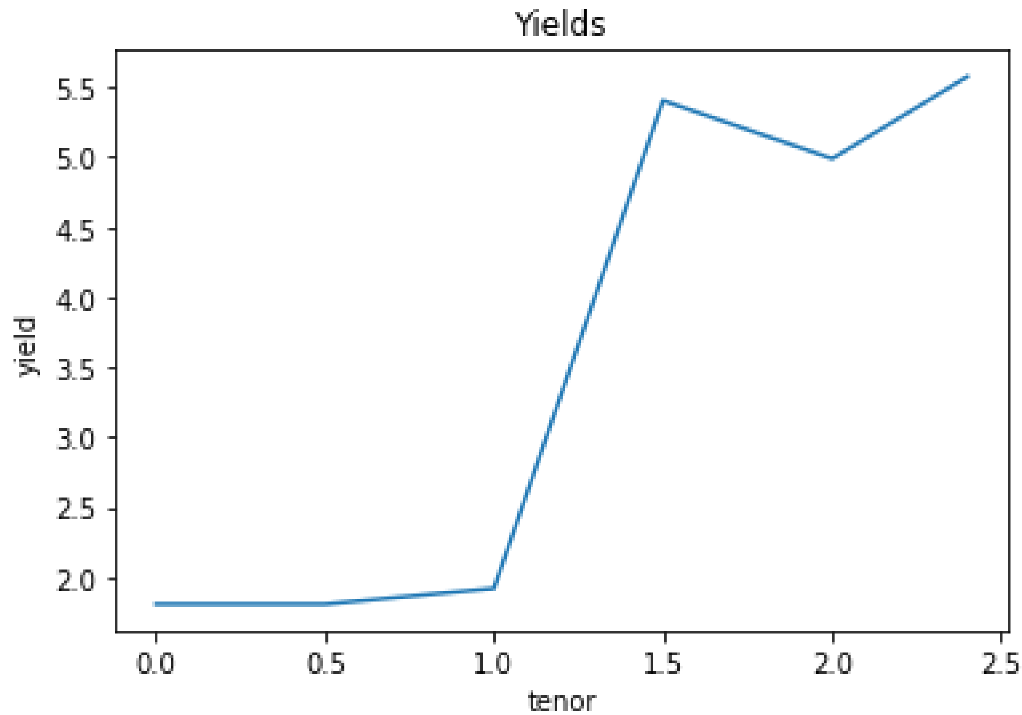
```
In [48]: bond11.set_price(9910)
         bond12.set_price(10500)
         bond13.set_price(46500)
         bond14.set_price(100000)
         bond15.set_price(103000)
```

```
In [49]: yc5 = fi.curve_factory(bondlist=bonds)
         yc5.plot_yields()
```



We see similar phenomena here to what we observed in the previous figure. The shorter maturity bonds have been restored to their original price levels, but we have also raised the prices of the longer bonds, implying much less discounting for the longer tenors, and thus we get the inversion. The plot seems to suggest that the 1.5 year bond, with a considerably discounted price at $46,500 may be a major reason for the strange looking yield curve, so we will see what happens when we push the price of that bond up:

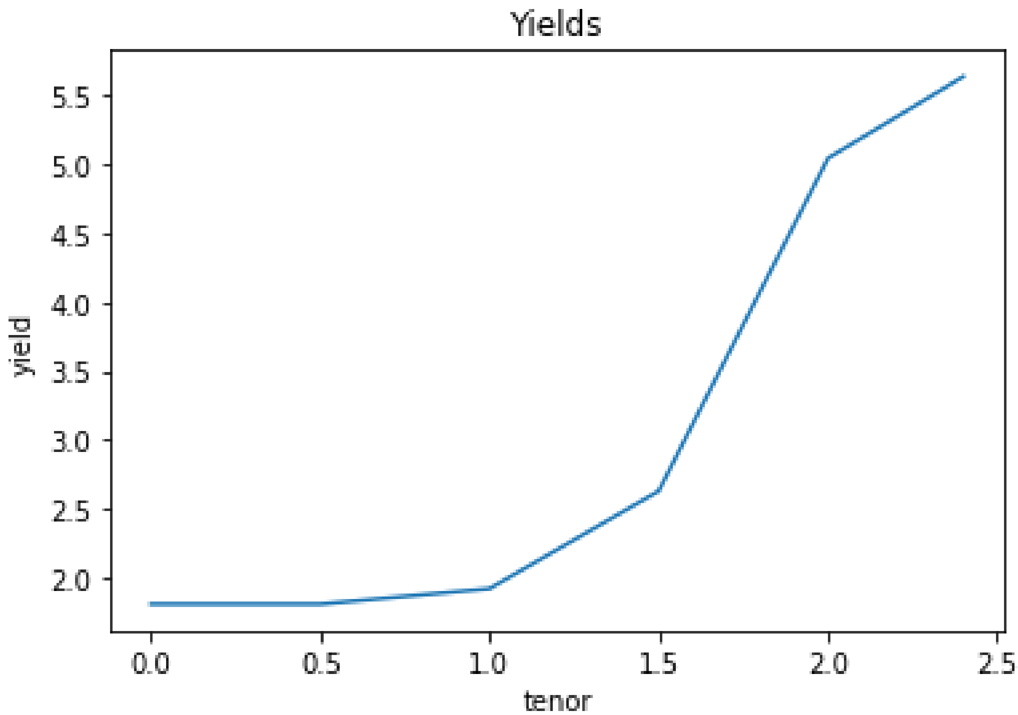
```
In [50]: bond11.set_price(9910)
         bond12.set_price(10500)
         bond13.set_price(49000)
         bond14.set_price(100000)
         bond15.set_price(103000)

In [51]: yc6 = fi.curve_factory(bondlist=bonds)
         yc6.plot_yields()
```



We seem to have a more reasonable looking yield curve now, but there is still an inversion, so we'll try pushing the price of the middle bond up a bit more:

```
In [52]: bond11.set_price(9910)
         bond12.set_price(10500)
         bond13.set_price(51000)
         bond14.set_price(100000)
         bond15.set_price(103000)
```

```
In [53]: yc7 = fi.curve_factory(bondlist=bonds)
         yc7.plot_yields()
```

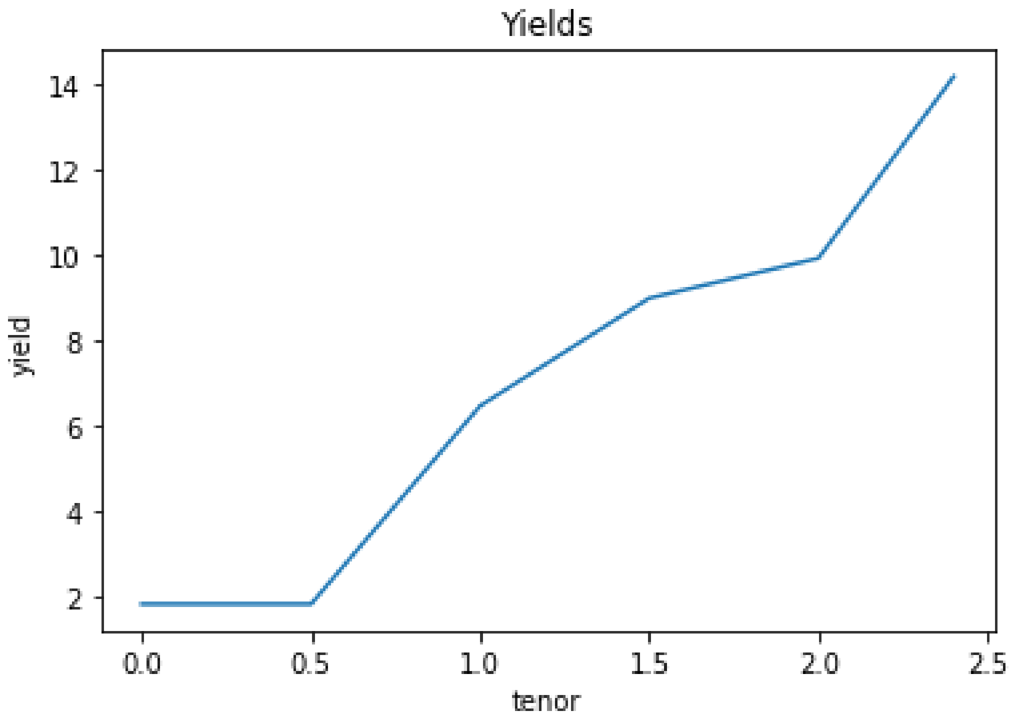


That seems to have done the trick. There are no inversions in this curve, suggesting that this might be a more reasonable distribution of prices for these bonds.

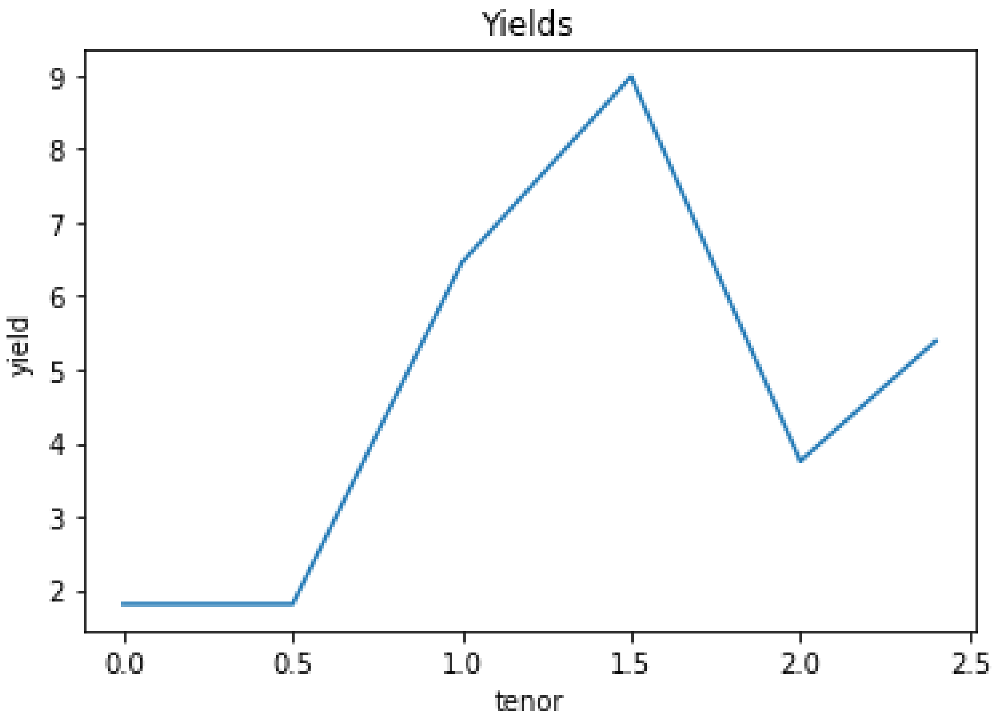
If instead of adjusting the prices we alter the coupons of the bonds we can predict what the effects should be, in light of the observations we just made. The effects will be similar, but in the opposite direction. Raising the value of the coupons, while keeping the bond prices constant, will force the discounting to be stronger, and thus will raise the interest rates. For instance, leaving all the bond prices the same as our original assumptions, but just raising the coupons will cause the long end of the yield curve to adjust up, whereas lowering them will reduce long term rates, and may ultimately lead to another inverted yield curve.

```
In [54]: bond11 = fi.bond_factory(dates=[0.5], rates=[10000])
         bond11.set_price(9910)
         bond12 = fi.create_coupon_bond(maturity=1.0, face=10000, rate=7, frequency=2)
         bond12.set_price(10050)
         bond13 = fi.create_coupon_bond(maturity=1.5, face=50000, rate=4, frequency=2)
         bond13.set_price(46500)
         bond14 = fi.create_coupon_bond(maturity=2.0, face=100000, rate=8, frequency=2)
         bond14.set_price(96500)
         bond15 = fi.create_coupon_bond(maturity=2.5, face=100000, rate=11, frequency=2)
         bond15.set_price(92000)
         bonds = [bond11, bond12, bond13, bond14, bond15]
```

```
In [55]: yc8 = fi.curve_factory(bondlist=bonds)
         yc8.plot_yields()
```




```
In [56]: bond14 = fi.create_coupon_bond(maturity=2.0, face=100000, rate=2, frequency=2)
         bond14.set_price(96500)
         bond15 = fi.create_coupon_bond(maturity=2.5, face=100000, rate=5, frequency=2)
         bond15.set_price(98000)
         bonds = [bond11, bond12, bond13, bond14, bond15]

In [57]: yc9 = fi.curve_factory(bondlist=bonds)
         yc9.plot_yields()
```



One can continue to experiment this way, adjusting the prices, coupon, maturities, payment frequencies, and see the effect this has on the implied yield curves, and students should carry out experiments like this in order to build intuition about bonds and yield curves.I