Case Assignment 1

-- adapted by Thomas Cassidey from *Practical Management Science*

Base Case

James Judson is the financial manager in charge of the company pension fund at Armco Incorporated. James knows that the fund must be sufficient to make the payments listed in Table 4.4. Each payment must be made on the first day of each year. James is going to finance these payments by purchasing bonds. It is currently January 1 of year 1, and three bonds are available for immediate purchase. The prices and coupons for the bonds are as follows. (We assume that all coupon payments are received on January 1 and arrive in time to meet cash demands for the year on which they arrive.)

- Bond 1 costs $980 and yields a 60 coupon in the years 2 through 5 and a $1060 payment on maturity in the year 6.
- Bond 2 costs $970 and yields a 65 coupon in the years 2 through 11 and a $1065 payment on maturity in the year 12.
- Bond 3 costs $1050 and yields a 75 coupon in the years 2 through 14 and a $1075 payment on maturity in the year 15.

James must decide how much cash to allocate (from company coffers) to meet the initial $11,000 payment and buy enough bonds to make future payments. He knows that any excess cash on hand can earn an annual rate of 4% in a fixed-rate account. How should he proceed?

Develop an LP model that relates initial allocation of money and bond purchases to future cash availabilities, and to minimize the initialize allocation of money required to meet all future pension fund payments.

In [2]:
```
import gurobipy as gp
from gurobipy import GRB
from itertools import product
import numpy as np

def get_solution(m):
    print("Solution Values:")
    for var in m.getVars():
        print(f"    {var.VarName}: {var.X}")

def get_shadow(m):
    print(f"Shadow Price Information:")
    for constraint in m.getConstrs():
```

```
        print(f"    {constraint.ConstrName}: {constraint.Pi}, (RHS = {constraint.RH

def get_rc(m):
    print(f"Reduced Cost Information")
    for var in m.getVars():
        print(f"    {var.VarName}: {var.RC}, (Coef LB = {round(var.SAObjLow, 2)}),
```

Parameters

In [3]:
```
T = 15 # length of the time horizon
B = [0, 1, 2] # bonds
COSTS = [980, 970, 1050] # costs in year one
D = [6, 12, 15] # duration (year of payout)

BASE_INCOMES = [60, 65, 75] # incomes in years 2 through (D_B - 1)
INCOMES = np.zeros((len(B), T))
PAYOUTS = [1060, 1065, 1075]
for b in B:
    for t in range(1, D[b] - 1):
        INCOMES[b, t] = BASE_INCOMES[b]
    INCOMES[b, D[b] - 1] = PAYOUTS[b]
    print(f"Bond {b} Incomes: {[x for x in INCOMES[b]]}")

BASE_r = .04

# Base Case
BASE_PAYMENTS = [x*1000 for x in [11, 12, 14, 15, 16, 18, 20, 21, 22, 24, 25, 30, 3
BASE_CASE = {
    'r': [BASE_r] * T, # interest rates are level
    'c' : BASE_PAYMENTS # payments go up steadily
}

# Bad Case
BAD_PAYMENTS = [11, 12, 14, 20, 20, 22, 23, 24, 26, 30, 31, 31, 31, 31, 31]
BAD_CASE = {
    'r': [.04, .04, .04, .03, .01, .0, .0, .0, .0, .0, .0, .01, .02, .03, .04], # i
    'c': [x*1000 for x in BAD_PAYMENTS] # payments increase dramatically in year 4
}
```

```
Bond 0 Incomes: [0.0, 60.0, 60.0, 60.0, 60.0, 1060.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0]
Bond 1 Incomes: [0.0, 65.0, 65.0, 65.0, 65.0, 65.0, 65.0, 65.0, 65.0, 65.0, 65.0, 10
65.0, 0.0, 0.0, 0.0]
Bond 2 Incomes: [0.0, 75.0, 75.0, 75.0, 75.0, 75.0, 75.0, 75.0, 75.0, 75.0, 75.0, 7
5.0, 75.0, 75.0, 1075.0]
```

Base Case Model

In [6]:
```
payments = BASE_CASE['c']

# Create a new model
m = gp.Model('base_case')
```

```
# Create variables
I = m.addVars(B, vtype=GRB.CONTINUOUS, name="I", lb=0.0) # Investment in each bond
C = m.addVar(vtype=GRB.CONTINUOUS, name="C", lb=0.0) # Cash needed up front
A = m.addVars(range(T), vtype=GRB.CONTINUOUS, name="A", lb=0.0) # Cash Available in
m.setObjective(C, GRB.MINIMIZE)

# Constraints
for t in range(T):
    # Set up the cash available at each time step t
    # Initial cash investment for t = 0 includes bond purchases and cash allocation
    if t == 0:
        m.addConstr(A[t] == C - gp.quicksum(COSTS[b] * I[b] for b in B), name=f"Ini
    else:
        # Cash available from previous year, adjusted for interest rate
        cash_from_previous = A[t - 1] * (1 + BASE_CASE['r'][t - 1])
        # Cash inflows from bond coupons and payouts for year t
        bond_cash_flows = gp.quicksum(INCOMES[b, t] * I[b] for b in B)
        # Total cash available for year t
        m.addConstr(A[t] == cash_from_previous + bond_cash_flows - payments[t], nam

# Solve the model
m.optimize()

# Output results
if m.status == GRB.OPTIMAL:
    get_solution(m)
    get_shadow(m)
    get_rc(m)
else:
    print("No optimal solution found.")
```

```
Gurobi Optimizer version 11.0.3 build v11.0.3rc0 (win64 - Windows 11.0 (22631.2))

CPU model: 11th Gen Intel(R) Core(TM) i3-1115G4 @ 3.00GHz, instruction set [SSE2|AVX
|AVX2|AVX512]
Thread count: 2 physical cores, 4 logical processors, using up to 4 threads

Optimize a model with 15 rows, 19 columns and 63 nonzeros
Model fingerprint: 0xe16d298d
Coefficient statistics:
  Matrix range     [1e+00, 1e+03]
  Objective range  [1e+00, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+04, 3e+04]
Presolve removed 1 rows and 2 columns
Presolve time: 0.04s
Presolved: 14 rows, 17 columns, 57 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   3.459274e+04   0.000000e+00      0s
       9    1.8676840e+05   0.000000e+00   0.000000e+00      0s

Solved in 9 iterations and 0.06 seconds (0.00 work units)
Optimal objective  1.867683968e+05
Solution Values:
    I[0]: 73.69479810424208
    I[1]: 77.2083720154974
    I[2]: 28.837209302325583
    C: 186768.39680049528
    A[0]: 9376.304035863655
    A[1]: 9354.378962234478
    A[2]: 7331.576885660132
    A[3]: 4227.862726022812
    A[4]: 0.0
    A[5]: 67297.82086917837
    A[6]: 57171.06858262725
    A[7]: 45639.24620461409
    A[8]: 32646.15093148042
    A[9]: 17133.331847421392
    A[10]: 0.0
    A[11]: 54389.70689417916
    A[12]: 27728.08586762075
    A[13]: 0.0
    A[14]: 0.0
Shadow Price Information:
    Initial_Cash_Allocation: -1.0, (RHS = 0.0), (LB = -inf), (UB = 186768.4)
    Year_2: -0.9615384615384615, (RHS = -12000.0), (LB = -inf), (UB = -2248.64)
    Year_3: -0.9245562130177513, (RHS = -14000.0), (LB = -inf), (UB = -4271.45)
    Year_4: -0.8889963586709146, (RHS = -15000.0), (LB = -inf), (UB = -7375.16)
    Year_5: -0.8548041910297255, (RHS = -16000.0), (LB = -inf), (UB = -11603.02)
    Year_6: -0.7190625344760272, (RHS = -18000.0), (LB = -63634.39), (UB = 60116.49)
    Year_7: -0.6914062831500261, (RHS = -20000.0), (LB = -67459.76), (UB = 49989.73)
    Year_8: -0.6648137337981019, (RHS = -21000.0), (LB = -70358.15), (UB = 38457.91)
    Year_9: -0.6392439748058671, (RHS = -22000.0), (LB = -73332.48), (UB = 25464.82)
    Year_10: -0.6146576680825645, (RHS = -24000.0), (LB = -77385.78), (UB = 9952.0)
    Year_11: -0.5910169885409273, (RHS = -25000.0), (LB = -80521.21), (UB = -7181.3
3)
```

```
    Year_12: -0.44999411599795885, (RHS = -30000.0), (LB = -91212.56), (UB = 52226.9
2)
    Year_13: -0.4326866499980373, (RHS = -31000.0), (LB = -94661.06), (UB = 25565.3)
    Year_14: -0.4160448557673435, (RHS = -31000.0), (LB = -97207.5), (UB = -2162.79)
    Year_15: -0.3593845096134623, (RHS = -31000.0), (LB = -95998.23), (UB = 0.0)
Reduced Cost Information
    I[0]: 0.0, (Coef LB = -111.65), (Coef UB = 109.04)
    I[1]: 0.0, (Coef LB = -39.71), (Coef UB = 125.98)
    I[2]: 0.0, (Coef LB = -386.34), (Coef UB = 43.71)
    C: 0.0, (Coef LB = 0.0), (Coef UB = inf)
    A[0]: 0.0, (Coef LB = -0.1), (Coef UB = 2.49)
    A[1]: 0.0, (Coef LB = -0.1), (Coef UB = 3.26)
    A[2]: 0.0, (Coef LB = -0.1), (Coef UB = 4.79)
    A[3]: 0.0, (Coef LB = -0.11), (Coef UB = 9.4)
    A[4]: 0.10697915517465717, (Coef LB = -0.11), (Coef UB = inf)
    A[5]: 0.0, (Coef LB = -0.11), (Coef UB = 1.4)
    A[6]: 0.0, (Coef LB = -0.11), (Coef UB = 1.72)
    A[7]: 0.0, (Coef LB = -0.12), (Coef UB = 2.25)
    A[8]: 0.0, (Coef LB = -0.12), (Coef UB = 3.32)
    A[9]: 0.0, (Coef LB = -0.12), (Coef UB = 6.5)
    A[10]: 0.1230231079030501, (Coef LB = -0.12), (Coef UB = inf)
    A[11]: 0.0, (Coef LB = -0.04), (Coef UB = 2.73)
    A[12]: 0.0, (Coef LB = -0.04), (Coef UB = 5.36)
    A[13]: 0.04228496576934271, (Coef LB = -0.04), (Coef UB = inf)
    A[14]: 0.3593845096134623, (Coef LB = -0.36), (Coef UB = inf)
```

Bad Case

The model above assumes a steady interest rate for all time periods $t = 1, \ldots, T$, but that is probably not a safe assumption. For example, an economic downturn is often accompanied by a dropping of interest rates. Further, this may be accompanied by an icrease in payments to the pension fund due to early retirements, etc. Develop a LP model in the cell below that uses the interest rates and payments of the `BAD_CASE` shown above. Assume that the interest rate for cash carried from year t to year t+1 is the interest rate for year t.

Bad Case Model

In [5]:
```
payments = BAD_CASE['c']
interest_rates = BAD_CASE['r']

# Create a new model
m = gp.Model('base_case')

# Create variables
I = m.addVars(B, vtype=GRB.CONTINUOUS, name="I", lb=0.0) # Investment in each bond
C = m.addVar(vtype=GRB.CONTINUOUS, name="C", lb=0.0) # Cash needed up front
A = m.addVars(range(T), vtype=GRB.CONTINUOUS, name="A", lb=0.0) # Cash Available in

m.setObjective(C, GRB.MINIMIZE)

# Constraints
```

```
for t in range(T):
    # Set up the cash available at each time step t
    # Initial cash investment for t = 0 includes bond purchases and cash allocation
    if t == 0:
        m.addConstr(A[t] == C - gp.quicksum(COSTS[b] * I[b] for b in B), name=f"Ini
    else:
        # Cash available from previous year, adjusted for interest rate
        cash_from_previous = A[t - 1] * (1 + BASE_CASE['r'][t - 1])
        # Cash inflows from bond coupons and payouts for year t
        bond_cash_flows = gp.quicksum(INCOMES[b, t] * I[b] for b in B)
        # Total cash available for year t
        m.addConstr(A[t] == cash_from_previous + bond_cash_flows - payments[t], nam

# Solve the model
m.optimize()

# Output results
if m.status == GRB.OPTIMAL:
    get_solution(m)
    get_shadow(m)
    get_rc(m)
else:
    print("No optimal solution found.")
```

```
Gurobi Optimizer version 11.0.3 build v11.0.3rc0 (win64 - Windows 11.0 (22631.2))

CPU model: 11th Gen Intel(R) Core(TM) i3-1115G4 @ 3.00GHz, instruction set [SSE2|AVX
|AVX2|AVX512]
Thread count: 2 physical cores, 4 logical processors, using up to 4 threads

Optimize a model with 15 rows, 19 columns and 63 nonzeros
Model fingerprint: 0x7b2ec187
Coefficient statistics:
  Matrix range     [1e+00, 1e+03]
  Objective range  [1e+00, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+04, 3e+04]
Presolve removed 1 rows and 2 columns
Presolve time: 0.00s
Presolved: 14 rows, 17 columns, 57 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   3.910745e+04   0.000000e+00      0s
       9    2.1181852e+05   0.000000e+00   0.000000e+00      0s

Solved in 9 iterations and 0.00 seconds (0.00 work units)
Optimal objective  2.118185235e+05
Solution Values:
    I[0]: 95.33811594860505
    I[1]: 78.14733915164764
    I[2]: 28.837209302325583
    C: 211818.5235016796
    A[0]: 12305.181127506574
    A[1]: 13760.043072054656
    A[2]: 13273.09949438466
    A[3]: 6766.678173607866
    A[4]: 0.0
    A[5]: 86300.77064805286
    A[6]: 73995.1692165065
    A[7]: 60197.34372769829
    A[8]: 43847.60521933774
    A[9]: 22843.87717064277
    A[10]: 0.0
    A[11]: 54389.70689417916
    A[12]: 27728.08586762075
    A[13]: 0.0
    A[14]: 0.0
Shadow Price Information:
    Initial_Cash_Allocation: -1.0, (RHS = 0.0), (LB = -inf), (UB = 211818.52)
    Cash_Available_Year_2: -0.9615384615384615, (RHS = -12000.0), (LB = -inf), (UB =
797.39)
    Cash_Available_Year_3: -0.9245562130177513, (RHS = -14000.0), (LB = -inf), (UB =
-690.72)
    Cash_Available_Year_4: -0.8889963586709146, (RHS = -20000.0), (LB = -inf), (UB =
-6195.98)
    Cash_Available_Year_5: -0.8548041910297255, (RHS = -20000.0), (LB = -inf), (UB =
-12962.65)
    Cash_Available_Year_6: -0.7190625344760272, (RHS = -22000.0), (LB = -81889.2),
(UB = 79058.4)
    Cash_Available_Year_7: -0.6914062831500261, (RHS = -23000.0), (LB = -85284.77),
```

```
(UB = 66752.8)
    Cash_Available_Year_8: -0.6648137337981019, (RHS = -24000.0), (LB = -88776.16),
(UB = 52954.98)
    Cash_Available_Year_9: -0.6392439748058671, (RHS = -26000.0), (LB = -93367.21),
(UB = 36605.24)
    Cash_Available_Year_10: -0.6146576680825645, (RHS = -30000.0), (LB = -100061.9),
(UB = 15601.51)
    Cash_Available_Year_11: -0.5910169885409273, (RHS = -31000.0), (LB = -103864.3
7), (UB = -7242.37)
    Cash_Available_Year_12: -0.44999411599795885, (RHS = -31000.0), (LB = -111333.5
3), (UB = 52226.92)
    Cash_Available_Year_13: -0.4326866499980373, (RHS = -31000.0), (LB = -114546.8
7), (UB = 25565.3)
    Cash_Available_Year_14: -0.4160448557673435, (RHS = -31000.0), (LB = -117888.7
5), (UB = -2162.79)
    Cash_Available_Year_15: -0.3593845096134623, (RHS = -31000.0), (LB = -116301.7
4), (UB = 0.0)
Reduced Cost Information
    I[0]: 0.0, (Coef LB = -111.65), (Coef UB = 109.04)
    I[1]: 0.0, (Coef LB = -39.71), (Coef UB = 125.98)
    I[2]: 0.0, (Coef LB = -386.34), (Coef UB = 43.71)
    C: 0.0, (Coef LB = 0.0), (Coef UB = inf)
    A[0]: 0.0, (Coef LB = -0.1), (Coef UB = 2.49)
    A[1]: 0.0, (Coef LB = -0.1), (Coef UB = 3.26)
    A[2]: 0.0, (Coef LB = -0.1), (Coef UB = 4.79)
    A[3]: 0.0, (Coef LB = -0.11), (Coef UB = 9.4)
    A[4]: 0.10697915517465717, (Coef LB = -0.11), (Coef UB = inf)
    A[5]: 0.0, (Coef LB = -0.11), (Coef UB = 1.4)
    A[6]: 0.0, (Coef LB = -0.11), (Coef UB = 1.72)
    A[7]: 0.0, (Coef LB = -0.12), (Coef UB = 2.25)
    A[8]: 0.0, (Coef LB = -0.12), (Coef UB = 3.32)
    A[9]: 0.0, (Coef LB = -0.12), (Coef UB = 6.5)
    A[10]: 0.1230231079030501, (Coef LB = -0.12), (Coef UB = inf)
    A[11]: 0.0, (Coef LB = -0.04), (Coef UB = 2.73)
    A[12]: 0.0, (Coef LB = -0.04), (Coef UB = 5.36)
    A[13]: 0.04228496576934271, (Coef LB = -0.04), (Coef UB = inf)
    A[14]: 0.3593845096134623, (Coef LB = -0.36), (Coef UB = inf)
```

Questions

Base case

Answer the following questions using only the base case model.

1. What are the obtimal Year 1 decision values, i.e. what are the values for C, and I_0, I_1, and I_2? (2.5)

    - Answer: I[0]: 73.69479810424208

    I[1]: 77.2083720154974 I[2]: 28.837209302325583 C: 186768.39680049528

2. In what years would it be valuable to reduce the pension payment needed, why?

- Answer: Years 1 to 5 are most valuable for reducing pension payments because their shadow prices are high in absolute value, indicating that reducing payments in these years would most significantly impact the initial cash allocation required which seems perfectly reasonable since the bond matures at the start of year 6 and we have more money available which reduces the effect of pension payment on intial cash allocation (we dont need as much moneyeffect of pension more money we hold in the starting years,the more returns we can expect in terms of coupon and interest.

3. For year 15, if the payment is reduced to zero, what would be the change in the objective function value? Give your answer in a formulaic format.

   - Answer:
     Change in Objective=shadow price×payment reduction=-0.3594×31,000=−11,131.4

4. For year 15, what is the maximum increase in payment value to keep the current basis solution optimal?

   - Answer: The maximum increase for year 15 is 0 since any increase in the value would result in increase in C which is something we can infer from looking at the reduced cost information for A[14](for every dollar added to the payment, C goes up by 0.36) and the upper bound of RHS of the shadow price.

Bad case comparison

Answer the following questions using both models.

5. How much more cash is needed for the optimal solution of the bad case?

   - Answer:C(base)-C(bad)= 211818.523-186768.396 = 25050.127

6. What is the most invested in bond in the bad case?

   - Answer: Bond 1 is the most invested bond in bad case and it makes sense since Bond 1 provides predictable payouts and a maturity payout earlier on, which can mitigate the reduced returns from holding cash as rates decline.

7. Do the years that are valuable for reducing pension payments change?

   - Answer: No, they dont since regardless of varying interest rates,the amount of money available for each year remains largely the same meaning the price sensitivity for payment reduction iss still going to be maximum for years 2 through 5 (since we do not get the bond matyration money till then).

In [ ]: