IE524 HW2 Solution

1 Problem 1

The problem can be formulated as:

$$\begin{array}{ll} \min & \frac{1}{T}\sum_{t=1}^{T} |\sum_{i=1}^{n}(R_i(t)-r_i)x_i| \\ \text{s.t.} & \sum_{i=1}^{n} r_i x_i \geq \bar{R} \\ & \sum_{i=1}^{n} x_i \leq 1 \\ & x_i \geq 0,\ \forall 1 \leq i \leq n. \end{array}$$

But the absolute value function is not linear so we have to reformulate it to an LP:

$$\begin{array}{ll} \min & \frac{1}{T}\sum_{t=1}^{T} z_t \\ \text{s.t.} & z_t \geq \sum_{i=1}^{n}(R_i(t)-r_i)x_i,\ \forall 1 \leq t \leq T \\ & z_t \geq -\sum_{i=1}^{n}(R_i(t)-r_i)x_i,\ \forall 1 \leq t \leq T \\ & \sum_{i=1}^{n} r_i x_i \geq \bar{R} \\ & \sum_{i=1}^{n} x_i \leq 1 \\ & x_i \geq 0,\ \forall 1 \leq i \leq n. \end{array}$$

2 Problem 2

2.1 Model

Decision variables we use:

$$\begin{array}{ll} x_t: & \text{credit line after month } t \\ y_i: & \text{amount of commercial paper } i \\ z_t: & \text{new cash position after month } t \end{array}$$

The linear programming formulation is

$$\begin{array}{ll} \max & V = z_6 \\ \text{s.t.} & x_1 + y_1 + y_2 - z_1 = 200 \\ & 1.005z_1 + x_2 + y_3 - 1.01y_1 - 1.02x_1 - z_2 = -100 \\ & 1.005z_2 + x_3 + y_4 - 1.02x_2 - z_3 = 150 \\ & 1.005z_3 + x_4 + y_5 - 1.05y_2 - 1.02x_3 - z_4 = -50 \\ & 1.005z_4 + x_5 - 1.04y_3 - 1.02x_4 - z_5 = 300 \\ & 1.005z_5 - 1.05y_4 - 1.03y_5 - 1.02x_5 - z_6 = -600 \\ & 0 \leq x_t \leq 100,\ y_i \geq 0,\ z_t \geq 0. \end{array}$$

[8]:
```
from amplpy import AMPL
```

[9]:
```
ampl1 = AMPL()
ampl1.reset()
ampl1.eval(r"""
    # set
    set MONTH = {1,2,3,4,5,6};
    set COMMERCIAL_PAPER = {1,2,3,4,5};

    # decision variables
    var x{MONTH}>=0, <=100;
    var y{COMMERCIAL_PAPER}>=0;
    var z{MONTH}>=0;

    # objective function
    maximize Final_Cash_Position: z[6];

    # constraints
    subject to month1: x[1]+y[1]+y[2]-z[1] = 200;
    subject to month2: 1.005*z[1]+x[2]+y[3]-1.01*y[1]-1.02*x[1]-z[2] = -100;
    subject to month3: 1.005*z[2]+x[3]+y[4]-1.02*x[2]-z[3] = 150;
    subject to month4: 1.005*z[3]+x[4]+y[5]-1.05*y[2]-1.02*x[3]-z[4] = -50;
    subject to month5: 1.005*z[4]+x[5]-1.04*y[3]-1.02*x[4]-z[5] = 300;
    subject to month6: 1.005*z[5]-1.05*y[4]-1.03*y[5]-1.02*x[5]-z[6] = -600;
""")
```

2.2 Solve the problem

[10]:
```
ampl1.option["solver"] = "gurobi"
ampl1.option["gurobi_options"] = 'outlev=0'
ampl1.solve()

FinalCashPosition = ampl1.get_objective("Final_Cash_Position")
print("The Optimal Final Cash Position is:", FinalCashPosition.value())
y = ampl1.get_variable("y").get_values()
print("\nThe amount of each commercial paper is:", y.to_pandas())
x = ampl1.get_variable("x").get_values()
print("\nThe credit line is:", x.to_pandas())
z = ampl1.get_variable("z").get_values()
print("\nThe resulting cash position in each month is:", z.to_pandas())
```

```
Gurobi 10.0.2:   tech:outlev = 0
Gurobi 10.0.2: optimal solution; objective 78.30606965
8 simplex iterations

The Optimal Final Cash Position is: 78.30606965174127
```

```
The amount of each commercial paper is:         y.val
1  200.000000
2    0.000000
3  102.000000
4  150.000000
5  254.557214

The credit line is:    x.val
1      0
2      0
3      0
4      0
5    100
6      0

The resulting cash position in each month is:         z.val
1    0.000000
2    0.000000
3    0.000000
4  304.557214
5    0.000000
6   78.306070
```

2.3 Using LP Answer Questions

2.3.1 Increasing the Interest Rate

[11]:
```
def model(IR):
    ampl = AMPL()
    ampl.reset()
    ampl.eval(r"""
        # set
        set MONTH = {1,2,3,4,5,6};
        set COMMERCIAL_PAPER = {1,2,3,4,5};

        # parameter
        param IR;

        # decision variables
        var x{MONTH}>=0, <=100;
        var y{COMMERCIAL_PAPER}>=0;
        var z{MONTH}>=0;

        # objective function
        maximize Final_Cash_Position: z[6];

        # constraints
        subject to month1: x[1]+y[1]+y[2]-z[1] = 200;
```

```
       subject to month2: 1.005*z[1]+x[2]+y[3]-1.01*y[1]-(1+IR)*x[1]-z[2] =␣
↪-100;
       subject to month3: 1.005*z[2]+x[3]+y[4]-(1+IR)*x[2]-z[3] = 150;
       subject to month4: 1.005*z[3]+x[4]+y[5]-1.05*y[2]-(1+IR)*x[3]-z[4] = -50;
       subject to month5: 1.005*z[4]+x[5]-1.04*y[3]-(1+IR)*x[4]-z[5] = 300;
       subject to month6: 1.005*z[5]-1.05*y[4]-1.03*y[5]-(1+IR)*x[5]-z[6] =␣
↪-600;
   """)
   ampl.param["IR"] = IR
   ampl.option["solver"] = "gurobi"
   ampl.solve()
   x = ampl.get_variable("x").get_values().to_list(skip_index = True)
   if all(x[i] ==0 for i in range(1,6)):
       print("Interest Rate:%.3f, the company not uses credit line"%IR)
   else:
       print("Interest Rate:%.3f, the company uses credit line"%IR)
```

First consider the case when interest rate = 0.03.

[12]:
```
model(0.03)
```

```
Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.81850746
8 simplex iterations

Interest Rate:0.030, the company not uses credit line
```

Next, use line search to find the minimum value under which the company never uses credit line.

[13]:
```
IR = 0.02
while(IR <= 0.03):
    print("\n")
    model(IR)
    IR += 0.001
```

```
Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 78.30606965
8 simplex iterations

Interest Rate:0.020, the company uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 78.20606965
8 simplex iterations

Interest Rate:0.021, the company uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 78.10606965
```

```
8 simplex iterations

Interest Rate:0.022, the company uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 78.00606965
8 simplex iterations

Interest Rate:0.023, the company uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.90606965
8 simplex iterations

Interest Rate:0.024, the company uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.81850746
8 simplex iterations

Interest Rate:0.025, the company not uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.81850746
8 simplex iterations

Interest Rate:0.026, the company not uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.81850746
8 simplex iterations

Interest Rate:0.027, the company not uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.81850746
8 simplex iterations

Interest Rate:0.028, the company not uses credit line


Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.81850746
8 simplex iterations

Interest Rate:0.029, the company not uses credit line
```

So when interest rate is approximately 2.5%, the company start to not use credit line.

2.3.2 One Commercial Paper not available

When one commercial paper is no longer available, we just need to remove the corresponding decision varaible y_3 from the model.

[14]:
```
ampl2 = AMPL()
ampl2.reset()
ampl2.eval(r"""
    # set
    set MONTH = {1,2,3,4,5,6};
    set COMMERCIAL_PAPER = {1,2,4,5};

    # decision variables
    var x{MONTH}>=0, <=100;
    var y{COMMERCIAL_PAPER}>=0;
    var z{MONTH}>=0;

    # objective function
    maximize Final_Cash_Position: z[6];

    # constraints
    subject to month1: x[1]+y[1]+y[2]-z[1] = 200;
    subject to month2: 1.005*z[1]+x[2]-1.01*y[1]-1.02*x[1]-z[2] = -100;
    subject to month3: 1.005*z[2]+x[3]+y[4]-1.02*x[2]-z[3] = 150;
    subject to month4: 1.005*z[3]+x[4]+y[5]-1.05*y[2]-1.02*x[3]-z[4] = -50;
    subject to month5: 1.005*z[4]+x[5]-1.02*x[4]-z[5] = 300;
    subject to month6: 1.005*z[5]-1.05*y[4]-1.03*y[5]-1.02*x[5]-z[6] = -600;
""")
ampl2.option["solver"] = "gurobi"
ampl2.solve()
FinalCashPosition = ampl2.get_objective("Final_Cash_Position")
print("The Optimal Final Cash Position is:", FinalCashPosition.value())
y = ampl2.get_variable("y").get_values()
print("\nThe amount of each commercial paper is:", y.to_pandas())
x = ampl2.get_variable("x").get_values()
print("\nThe credit line is:", x.to_pandas())
z = ampl2.get_variable("z").get_values()
print("\nThe resulting cash position in each month is:", z.to_pandas())
```

```
Gurobi 10.0.2:Gurobi 10.0.2: optimal solution; objective 77.80408354
8 simplex iterations

The Optimal Final Cash Position is: 77.80408354268258

The amount of each commercial paper is:         y.val
1   99.009901
2  100.990099
4  150.000000
5  255.044579
```

```
The credit line is:    x.val
1      0
2      0
3      0
4      0
5    100
6      0

The resulting cash position in each month is:         z.val
1    0.000000
2    0.000000
3    0.000000
4  199.004975
5    0.000000
6   77.804084
```

[ ]: