

Formulate and solve the following MINLP problem using GAMS/DICOPT or via a Python MINLP routine.
Submit your formulation, the code, and an output file.

The Jayhawk Chemical Company has two products $k = 1, 2$, one factory, two distribution centers $i = 1, 2$, and five clients $j = 1, \dots, 5$ whose product demands d_{jk} are known. The company must decide which products should be handled by which distribution center and how each client should be serviced. The goal is to minimize total cost, and the cost may be defined as

- A fixed cost F_{ik} if product k is handled by distribution center i .
- Fixed costs G_{ijk} to be paid if the demand of client j for product k is satisfied by center i .
- Shipping costs C_{ijk} per unit of product k squared, shipped to client j via center i .

Following the plant design example from Lecture 27, the goal of this MINLP is to write the problem in the following formulation such that it is easily implemented using standard discrete solvers.

$$\min_{\mathbf{x}, \mathbf{y}} f(\mathbf{x}) + \mathbf{c}^T \mathbf{y} \quad \text{s.t.} \quad h_i(\mathbf{x}) = 0 \quad \forall i \quad g_j(\mathbf{x}) + \mathbf{M}\mathbf{y} \leq 0 \quad \forall j, \quad \mathbf{x} \in \mathbb{R}, \quad \mathbf{y} \in \{0, 1\}$$

The binary variable $y_{ik} \in \{0, 1\}$ is defined to denote which of the two distribution centers will handle which product with associated fixed cost F_{ik} . The binary variable $z_{ijk} \in \{0, 1\}$ is defined to denote which distribution center serves which client for each product with associated fixed cost G_{ijk} . The continuous variable x_{ijk} is defined to denote the amount of product k to ship from distribution center i to client j . Together, these discrete and continuous variables tell us what to make where, who serves who, and how much to ship.

Minimizing the total cost to the company involves the minimization of the product fixed cost:

$$\sum_{i,k} F_{ik} y_{ik},$$

minimization of the client assignment fixed cost:

$$\sum_{i,j,k} G_{ijk} z_{ijk},$$

and minimization of the shipping cost (per unit of product squared):

$$\sum_{i,j,k} C_{ijk} x_{ijk}^2,$$

resulting in the full convex (linear+linear+quadratic) cost function:

$$f(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{i,k} F_{ik} y_{ik} + \sum_{i,j,k} G_{ijk} z_{ijk} + \sum_{i,j,k} C_{ijk} x_{ijk}^2.$$

The combined cost function is subject to the constraints in which each client demand must be met:

$$\sum_i x_{ijk} = d_{jk} \quad \forall j, k,$$

the product can't be shipped to a client unless the distribution center is serving that client:

$$x_{ijk} \leq M_{jk} z_{ijk} \quad \forall i, j, k,$$

and we cannot serve a product to a client from a distribution center unless that distribution center handles the product:

$$z_{ijk} \leq y_{ik} \quad \forall i, j, k$$

Combining all of the derived equations into the final MINLP optimization problem in which all equations are either linear or quadratic cumulatively making the system convex.

$$\min_{\mathbf{x}, \mathbf{y}, \mathbf{z}} \left[f(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{i,k} F_{ik} y_{ik} + \sum_{i,j,k} G_{ijk} z_{ijk} + \sum_{i,j,k} C_{ijk} x_{ijk}^2 \right]$$

s.t.

$$\sum_i x_{ijk} = d_{jk} \quad \forall j, k$$

$$x_{ijk} \leq M_{jk} z_{ijk} \quad \forall i, j, k$$

$$z_{ijk} \leq y_{ik} \quad \forall i, j, k$$

$$\mathbf{x} \in \mathbb{R}, \quad \mathbf{y}, \mathbf{z} \in \{0, 1\}$$

The problem is solved below by first defining the given number of products, distribution centers, and clients as well as example demand values for each client.

```
K = 2; % products
I = 2; % distribution centers
J = 5; % clients

% client (j) demand for product (i)
d = [8 4; 5 2; 6 3; 7 4; 9 1];
```

Next the fixed costs for each distribution center to make each product, the fixed costs for each distribution center to serve each client for each product, and the fixed costs for each distribution center to ship each client each product.

```
% costs for dist. center (i) to make product (k)
```

```

F = [40 55; 25 65];

% costs for dist. center (i) serving client (j) with product (k)
G = zeros(I,J,K);
G(1,:,:1) = [3 4 2 1 4]; G(1,:,:2) = [3 3 1 2 4];
G(2,:,:1) = [4 3 9 9 5]; G(2,:,:2) = [4 4 9 3 4];

% costs for dist. center (i) to ship each client (j) product (k)
C = zeros(I,J,K);
C(1,:,:1) = [0.8 0.1 0.1 0.7 0.9]; C(1,:,:2) = [0.6 0.2 1.0 0.5 0.7];
C(2,:,:1) = [0.5 0.1 0.5 0.8 0.6]; C(2,:,:2) = [0.4 1.0 0.9 0.6 0.5];

```

The optimization variables are defined below for shipment quantity x , distribution product binary y , and product assignment binary z , as well as the big-M constraint that allows for a continuous linear representation.

```

% optimization variables
x = optimvar('x',I,J,K,'LowerBound',0);
y = optimvar('y',I,K,'Type','integer','LowerBound',0,'UpperBound',1);
z = optimvar('z',I,J,K,'Type','integer','LowerBound',0,'UpperBound',1);

% big M parameter per (j, k)
M = d*100;

```

Next, the objective (cost function) is defined as outlined in the problem formulation.

```

% cost function
cost = sum(sum(F .* y)) + sum(G.*z,'all') + sum(C .* (x.^2),'all');

% denote as problem var type for matlab optimization toolbox
prob = optimproblem('Objective', cost, 'ObjectiveSense', 'min');

```

The constraints are similarly outlined below using the same logic as above.

```

% constraint 1: demand satisfaction
cons1 = optimconstr(J,K);
for j = 1:J
    for k = 1:K
        cons1(j,k) = sum(x(:,j,k)) == d(j,k);
    end
end

% constraint 2: product shipping
cons2 = optimconstr(I,J,K);
for i = 1:I
    for j = 1:J
        for k = 1:K
            cons2(i,j,k) = x(i,j,k) <= M(j,k)*z(i,j,k);
        end
    end
end

```

```

prob.Constraints.shipping = cons2;

% constraint 3: dist. center serving
cons3 = optimconstr(I,J,K);
for i = 1:I
    for j = 1:J
        for k = 1:K
            cons3(i,j,k) = z(i,j,k) <= y(i,k);
        end
    end
end
prob.Constraints.serving = cons3;

```

Finally, a genetic algorithm based solver is used to solve the framed MINLP.

```

% set MINLP solver options and run
gaopts =
optimoptions('ga','Display','iter','MaxGenerations',100,'MaxStallGenerations',100,'PopulationSize',500);
[sol,fval,exitflag,output] = solve(prob,'Solver','ga','Options',gaopts);

```

Solving problem using ga.

Single objective optimization:
44 Variables
24 Integer variables
40 Linear inequality constraints

Options:
CreationFcn: @gacreationuniform
CrossoverFcn: @crossoveralaplace
SelectionFcn: @selectiontournament
MutationFcn: @mutationpower

Generation	Func-count	Best Penalty	Mean Penalty	Stall Generations
1	998	0	2.189e+06	0
2	1491	0	1.202e+06	1
3	1985	0	9.074e+05	2
4	2478	0	6.989e+05	3
5	2972	0	6.195e+05	4
6	3466	0	5.936e+05	5
7	3960	0	5.888e+05	6
8	4454	0	5.984e+05	7
9	4948	0	5.969e+05	8
10	5442	0	6.3e+05	9
11	5936	0	6.096e+05	10
12	6430	0	6.742e+05	11
13	6924	0	6.652e+05	12
14	7418	0	7.149e+05	13
15	7912	0	7.988e+05	14
16	8406	0	8.63e+05	15
17	8899	0	8.725e+05	16
18	9393	0	9.121e+05	17
19	9887	0	9.839e+05	18
20	10381	0	8.143e+05	19
21	10874	0	8.554e+05	20
22	11368	0	8.881e+05	21
23	11861	0	8.801e+05	22

24	12355	0	7.959e+05	23
25	12848	0	8.682e+05	24
26	13342	0	9.063e+05	25
27	13835	0	8.17e+05	26
28	14329	0	8.667e+05	27
29	14822	0	8.711e+05	28

Generation	Func-count	Best Penalty	Mean Penalty	Stall Generations
30	15316	0	9.557e+05	29
31	15809	0	8.424e+05	30
32	16303	0	8.189e+05	31
33	16796	0	9.153e+05	32
34	17290	0	8.544e+05	33
35	17783	0	8.522e+05	34
36	18277	0	7.968e+05	35
37	18771	0	8.72e+05	36
38	19265	0	8.259e+05	37
39	19758	0	8.053e+05	38
40	20252	0	8.291e+05	39
41	20746	0	8.152e+05	40
42	21240	0	7.948e+05	41
43	21734	0	8.169e+05	42
44	22228	0	8.698e+05	43
45	22722	0	7.862e+05	44
46	23216	0	7.746e+05	45
47	23709	0	7.813e+05	46
48	24203	0	8.807e+05	47
49	24696	0	8.424e+05	48
50	25190	0	8.434e+05	49
51	25683	0	8.396e+05	50
52	26177	0	8.458e+05	51
53	26670	0	9.089e+05	52
54	27164	0	8.918e+05	53
55	27657	0	8.316e+05	54
56	28151	0	8.285e+05	55
57	28644	0	8.494e+05	56
58	29138	0	8.403e+05	57
59	29631	0	9.193e+05	58

Generation	Func-count	Best Penalty	Mean Penalty	Stall Generations
60	30125	0	9.15e+05	59
61	30618	0	8.606e+05	60
62	31112	0	7.919e+05	61
63	31605	0	8.237e+05	62
64	32099	0	8.974e+05	63
65	32592	0	8.059e+05	64
66	33086	0	8.396e+05	65
67	33579	0	8.44e+05	66
68	34073	0	7.784e+05	67
69	34567	0	8.252e+05	68
70	35061	0	9.178e+05	69
71	35554	0	8.83e+05	70
72	36048	0	7.541e+05	71
73	36541	0	8.578e+05	72
74	37035	0	8.223e+05	73
75	37528	0	7.661e+05	74
76	38022	0	8.699e+05	75
77	38516	0	8.881e+05	76
78	39010	0	8.529e+05	77
79	39503	0	8.139e+05	78
80	39997	0	8.352e+05	79
81	40490	0	8.339e+05	80

```

82      40984      0    7.764e+05      81
83      41477      0    7.965e+05      82
84      41971      0    8.878e+05      83
85      42464      0    8.187e+05      84
86      42958      0    8.817e+05      85
87      43452      0    8.627e+05      86
88      43946      0    8.026e+05      87
89      44440      0    7.865e+05      88

```

Generation	Func-count	Best Penalty	Mean Penalty	Stall Generations
90	44933	0	8.443e+05	89
91	45427	0	8.938e+05	90
92	45921	0	9.153e+05	91
93	46415	0	8.472e+05	92
94	46908	0	9.437e+05	93
95	47402	0	9.357e+05	94
96	47895	0	8.043e+05	95
97	48389	0	8.043e+05	96
98	48882	0	8.375e+05	97
99	49376	0	7.864e+05	98
100	49869	0	8.13e+05	99

ga stopped because it exceeded options.MaxGenerations.

```
disp('==> Solution Summary ==>');
```

```
==> Solution Summary ==>
```

```
disp(['Total cost: ', num2str(fval)]);
```

```
Total cost: 0
```

```
disp('y(i,k):'); disp(round(sol.y));
```

```
y(i,k):
```

```
 0      0
 0      0
```

```
disp('z(i,j,k):'); disp(round(sol.z));
```

```
z(i,j,k):
```

```
(:,:,1) =
```

```
 0      0      0      0      0
 0      0      0      0      0
```

```
(:,:,2) =
```

```
 0      0      0      0      0
 0      0      0      0      0
```

```
disp('x(i,j,k):'); disp(sol.x);
```

```
x(i,j,k):
```

```
(:,:,1) =
```

```
 0      0      0      0      0
 0      0      0      0      0
```

$(:,:,2) =$

0	0	0	0	0
0	0	0	0	0