Exam1

Contents

Problem 1

```
clear
disp('Problem 1');
```

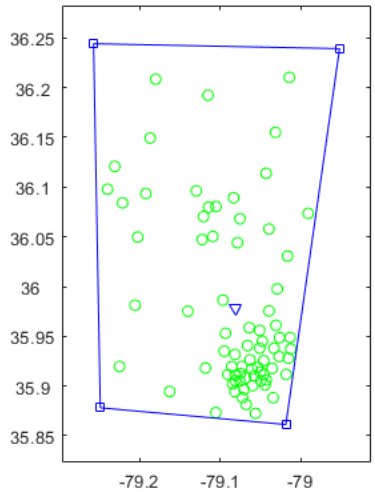
```
Problem 1
```

Define initial conditions

Geolocate aggreagate demand points

```
NC = uszip5(strcmp('NC', uszip5('ST')));
OC = uscounty(mand('NC', uscounty('ST'), 'Orange', uscounty('Name')));
s = uscenblkgrp(uscenblkgrp('SCfips') == OC.SCfips);
OCxy = [-79.257497 36.243968; % Coordinates of Orange county limit
    -78.951653 36.239008;
    -79.018391 35.861254;
    -79.249062 35.878076];

makemap(OCxy)
% pplot(NC.XY, 'r.')
OC_ctr = pplot(OC.XY, 'bv')
CBG = pplot(s.XY, 'go')
BRD = pplot([OCxy; OCxy(1,:)], 'bs-');
```

```
OC_ctr =

  Line with properties:

              Color: [0 0 1]
          LineStyle: 'none'
          LineWidth: 0.5000
             Marker: 'v'
         MarkerSize: 6
    MarkerFaceColor: 'none'
              XData: -79.0813
              YData: 38.6067
              ZData: [1×0 double]

  Use GET to show all properties


CBG =

  Line with properties:

              Color: [0 1 0]
          LineStyle: 'none'
          LineWidth: 0.5000
             Marker: 'o'
         MarkerSize: 6
    MarkerFaceColor: 'none'
              XData: [1×75 double]
              YData: [1×75 double]
              ZData: [1×0 double]

  Use GET to show all properties
```



Define the demand and distances

```
dem = s.Pop*4;

dafh = @(XY1,a1,XY2,a2) max(1.2*dists(XY1,XY2,'mi'),...
   0.675*max(sqrt(a1(:)),sqrt(a2(:)')));

Da = dafh(s.XY,s.LandArea,s.XY,s.LandArea);  % Area-adjusted distances
```

Build constatnt and variable cost matrices

```
C = Da.*dem/2000;
V = 8e4; % maximum capacity for the facility
k = repmat(50,1,size(C,1)); % fixed cost = 50
```

Create MILP model for CFL

```
clear mp
mp = Milp('CFL')
[n m] = size(C);
mp.addobj('min',k,C)
for i = 1:n
    mp.addcstr({V,{i}},'>=',{dem,{i,':'}})
end
for j = 1:m
   mp.addcstr(0,{':',j},'=',1)

end
mp.addub(1,1)
mp.addctype('B','C')
mp.Model
```

```
mp = 

  Milp with properties:

    Model: [1×1 struct]


ans = 

  struct with fields:

     name: 'CFL'
    sense: 'minimize'
      obj: [1×5700 double]
       lb: [1×5700 double]
       ub: [1×5700 double]
    ctype: 'BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        A: [150×5700 double]
      lhs: [150×1 double]
      rhs: [150×1 double]
```

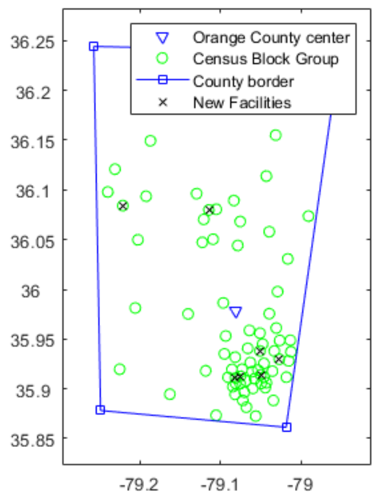
Solve using Gurobi

```
clear model param
model = mp.milp2gb;
```

```
params.outputflag = 1;
result = gurobi(model, params);
x = result.x;
x = mp.namesolution(x);
TC = result.objval;
idxNF = find(round(x.k));
nNF = sum(x.k);
NF = pplot(s.XY(idxNF,:), 'kx')
legend([OC_ctr CBG BRD NF], ["Orange County center"...
    "Census Block Group" "County border" "New Facilities"])
fprintf('The optimal number of centers with 0 fixed cost is %d.\n', nNF);
fprintf('They should be located at:\n');
lonlat2city(s.XY(idxNF,:), uscity)
```

```
Academic license - for non-commercial use only
Gurobi Optimizer version 9.0.3 build v9.0.3rc0 (win64)
Optimize a model with 150 rows, 5700 columns and 11325 nonzeros
Model fingerprint: 0xd99953f9
Variable types: 5625 continuous, 75 integer (75 binary)
Coefficient statistics:
  Matrix range     [1e+00, 8e+04]
  Objective range  [1e-01, 2e+02]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 5283.9475337
Presolve time: 0.01s
Presolved: 150 rows, 5700 columns, 11325 nonzeros
Variable types: 5625 continuous, 75 integer (75 binary)

Root relaxation: objective 5.369102e+02, 233 iterations, 0.01 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0  536.91021    0   26 5283.94753  536.91021  89.8%     -    0s
H    0     0                    1602.4077051  536.91021  66.5%     -    0s
H    0     0                    1353.1027811  536.91021  60.3%     -    0s
H    0     0                     709.2463427  536.91021  24.3%     -    0s
     0     0  573.16958    0   24  709.24634  573.16958  19.2%     -    0s
     0     0  573.17006    0   23  709.24634  573.17006  19.2%     -    0s
     0     0  598.29650    0   15  709.24634  598.29650  15.6%     -    0s
     0     0  599.86111    0   11  709.24634  599.86111  15.4%     -    0s
     0     0  602.21044    0    5  709.24634  602.21044  15.1%     -    0s
     0     0  602.27019    0    5  709.24634  602.27019  15.1%     -    0s
     0     0  602.28777    0    5  709.24634  602.28777  15.1%     -    0s
     0     0  602.44271    0    4  709.24634  602.44271  15.1%     -    0s
H    0     0                     700.6844649  602.44271  14.0%     -    0s
     0     0  602.59603    0    7  700.68446  602.59603  14.0%     -    0s
     0     0  603.01307    0    4  700.68446  603.01307  13.9%     -    0s
H    0     0                     700.1975765  603.01307  13.9%     -    0s
     0     0  603.07575    0    4  700.19758  603.07575  13.9%     -    0s
     0     0  603.07575    0    4  700.19758  603.07575  13.9%     -    0s
H    0     0                     699.5430952  603.07575  13.8%     -    0s
H    0     0                     604.8667441  603.07575  0.30%     -    0s
     0     0  603.07575    0   13  604.86674  603.07575  0.30%     -    0s
     0     0  603.07575    0    8  604.86674  603.07575  0.30%     -    0s
     0     0  603.07575    0    4  604.86674  603.07575  0.30%     -    0s
     0     0  603.20717    0    3  604.86674  603.20717  0.27%     -    0s
     0     0  603.37647    0    2  604.86674  603.37647  0.25%     -    0s
     0     0  603.45200    0    2  604.86674  603.45200  0.23%     -    0s
     0     0  603.59029    0    5  604.86674  603.59029  0.21%     -    0s
     0     0  603.59029    0    8  604.86674  603.59029  0.21%     -    0s
     0     0  603.59029    0    5  604.86674  603.59029  0.21%     -    0s
     0     0  603.59029    0    5  604.86674  603.59029  0.21%     -    0s
     0     0  603.59029    0    5  604.86674  603.59029  0.21%     -    0s
     0     0  603.67184    0    3  604.86674  603.67184  0.20%     -    0s
     0     0  603.70015    0    3  604.86674  603.70015  0.19%     -    0s
     0     0  603.73246    0    4  604.86674  603.73246  0.19%     -    0s
     0     0  603.73308    0    4  604.86674  603.73308  0.19%     -    0s
     0     0  603.73448    0    3  604.86674  603.73448  0.19%     -    0s
     0     0     cutoff    0       604.86674  604.86674  0.00%     -    0s

Cutting planes:
  Gomory: 1
  Implied bound: 38
  MIR: 1
  Relax-and-lift: 5

Explored 1 nodes (977 simplex iterations) in 0.46 seconds
Thread count was 4 (of 4 available processors)

Solution count 8: 604.867 699.543 700.198 ... 5283.95

Optimal solution found (tolerance 1.00e-04)
Best objective 6.048667440514e+02, best bound 6.048667440514e+02, gap 0.0000%

NF = 

  Line with properties:
```

```
            Color: [0 0 0]
        LineStyle: 'none'
        LineWidth: 0.5000
           Marker: 'x'
       MarkerSize: 6
  MarkerFaceColor: 'none'
            XData: [1×7 double]
            YData: [38.5230 38.5252 38.7319 38.7373 38.5268 38.5567 38.5463]
            ZData: [1×0 double]

  Use GET to show all properties

The optimal number of centers with 0 fixed cost is 7.
They should be located at:

X 1 is in Carrboro, NC
X 2 is in Carrboro, NC
X 3 is in Hillsborough, NC
X 4 is in Efland, NC
X 5 is in Chapel Hill, NC
X 6 is in Chapel Hill, NC
X 7 is in Chapel Hill, NC
```



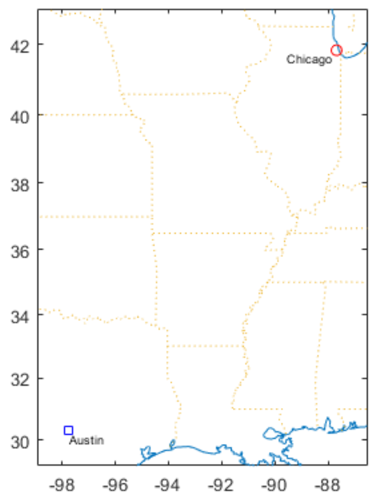
Problem 2

```
clear
disp('Problem 2');
```

```
Problem 2
```

Define initial conditions

```
f = [ 720 120 480];
wt = [16 150 40];
cu = [2 6 12];
uc = [500 200 100];
sv = [50 180 80];
Sup = uscity(mand('Chicago',uscity('Name'),'IL',uscity('ST')));
Cust = uscity(mand('Austin',uscity('Name'),'TX',uscity('ST')));
makemap([Sup.XY; Cust.XY]);
pplot(Sup.XY, 'ro'), pplot(Sup.XY, Sup.Name);
pplot(Cust.XY, 'bs'), pplot(Cust.XY, Cust.Name);
```



Define shipment and truck structures

```
ppiTL = 136.3; % Jan 2020 (P)
ppiLTL = 193.6; % Jan 2020 (P)
tr = struct('r',2*(ppiTL/102.7),'Kwt',25,'Kcu',2750);
hobs = (uc-sv)./uc; %obsolescence rate
h = 0.11+hobs;
sh = vec2struct('f', f.* wt/2000, 's', wt./cu, 'd',...
    1.2*dists(Sup.XY, Cust.XY, 'mi'), 'v', uc, 'h', h, 'a', 1);
sdisp(sh);
```

```
sh:    f      s         d       v     h   a
--:-------------------------------------
 1:  5.76   8.00  1,167.51  500  1.01  1
 2:  9.00  25.00  1,167.51  200  0.21  1
 3:  9.60   3.33  1,167.51  100  0.31  1
```

find the optimal TLC for separate shipping

```
[~,q,isLTL] = minTLC(sh,tr,ppiLTL);
[TLC,TC,IC] = totlogcost(q,transcharge(q,sh,tr),sh);
days = 365.25*q./[sh.f];
```

find the optimal TLC for aggregate shipping

```
ash = aggshmt(sh);
[~,qa,isLTLa] = minTLC(ash,tr,ppiLTL);
[TLCa,TCa,ICa] = totlogcost(qa,transcharge(qa,ash,tr),ash);
daysa = 365.25*qa./[ash.f];
```

display results

```
M = [ TLC sum(TLC); TC sum(TC); IC sum(IC); q NaN; days NaN; isLTL NaN]';
Ma = [TLCa TCa ICa qa daysa isLTLa'];
mdisp([M; Ma],{'1','2','3', 'Sum', 'Aggregate'},...
   {'TLC', 'TC', 'IC','q', 'days', 'isLTL'},'sh');

fprintf('\nThe optimal way to ship products is via aggregate TL.\n')
fprintf(['Aggregate shipment should be made every %.0f days'...
    ' with a size of %.2f tons.\n'], daysa, qa);
```

```
       sh:     TLC         TC        IC       q      days    isLTL
---------:-------------------------------------------------------
        1:   6,004.75   3,002.37  3,002.37   5.95    377.00    0
        2:   2,165.62   1,115.62  1,050.00  25.00  1,014.58    0
        3:   6,632.99   6,490.91    142.08   4.58    174.38    0
      Sum:  14,803.36  10,608.91  4,194.46
Aggregate:   9,783.87   8,925.00    858.87   8.46    126.82    0

The optimal way to ship products is via aggregate TL.
Aggregate shipment should be made every 127 days with a size of 8.46 tons.
```