The problem discussed here (1) involves a network of factories, warehouses and sales outlets. We need to find the least expensive flow of products from factories to warehouses to stores. One particularity is that each store gets its products from one warehouse.

Mathematical Model

We use the following indices:

- p: products
- f: factories
- w: warehouses
- s: stores

We introduce the following variables:

- $x_{p,f,w} \geq 0$: shipments of product p from factory f to warehouse w,
- $y_{s,w} \in \{0,1\}$: links each store s to a single warehouse w.

The data associated with the model is:

- $pcost_{f,p}$: unit production cost
- $tcost_p$: unit transportation cost
- $dist_{f,w}, dist_{s,w}$: distances
- $pcap_{f,p}$: factory production capacities
- $wcap_w$: warehouse capacity
- $d_{s,p}$: demand for product p at store s
- $turn_p$: product turnover rate

The optimization model looks like:

$$
\begin{aligned}
\min \sum_{p,f,w} (pcost_{f,p} + tcost_p \cdot dist_{f,w}) \cdot x_{p,f,w} + \sum_{s,w,p} d_{s,p} \cdot tcost_p \cdot dist_{s,w} \cdot y_{s,w} \\
\sum_w x_{p,f,w} \leq pcap_{f,p} \ \ \forall f,p \ \ \text{(production capacity)} \\
\sum_f x_{p,f,w} = \sum_s d_{s,p} \cdot y_{s,w} \ \ \forall p,w \ \ \text{(demand)} \\
\sum_{p,s} \frac{d_{s,p}}{turn_p} y_{s,w} \leq wcap_w \ \ \forall w \ \ \text{(warehouse capacity)} \\
\sum_w y_{s,w} = 1 \ \ \forall s \ \ \text{(one warehouse for a store)} \\
x_{p,f,w} \geq 0 \\
y_{s,w} \in \{0,1\}
\end{aligned}
$$

Matlab code

```
rng(1) % for reproducibility
N = 20; % N from 10 to 30 seems to work. Choose large values with
caution.
N2 = N*N;
f = 0.05; % density of factories
w = 0.05; % density of warehouses
s = 0.1; % density of sales outlets

F = floor(f*N2); % number of factories
W = floor(w*N2); % number of warehouses
S = floor(s*N2); % number of sales outlets
```

```
xyloc = randperm(N2,F+W+S); % unique locations of facilities
[xloc,yloc] = ind2sub([N N],xyloc);
```

Pick unique locations on a grid

```
P = 20; % 20 products

% Production costs between 20 and 100
pcost = 80*rand(F,P) + 20;

% Production capacity between 500 and 1500 for each product/factory
pcap = 1000*rand(F,P) + 500;

% Warehouse capacity between P*400 and P*800 for each
product/warehouse
wcap = P*400*rand(W,1) + P*400;

% Product turnover rate between 1 and 3 for each product
turn = 2*rand(1,P) + 1;

% Product transport cost per distance between 5 and 10 for each
product
tcost = 5*rand(1,P) + 5;

% Product demand by sales outlet between 200 and 500 for each
% product/outlet
d = 300*rand(S,P) + 200;
```

```
distfw = zeros(F,W); % Allocate matrix for factory-warehouse
distances
for ii = 1:F
    for jj = 1:W
        distfw(ii,jj) = abs(xloc(ii) - xloc(F + jj)) +
            abs(yloc(ii) - yloc(F + jj));
    end
end

distsw = zeros(S,W); % Allocate matrix for sales outlet-warehouse
distances
for ii = 1:S
    for jj = 1:W
        distsw(ii,jj) = abs(xloc(F + W + ii) - xloc(F + jj))
            + abs(yloc(F + W + ii) - yloc(F + jj));
    end
end
```

```
obj1 = zeros(P,F,W); % Allocate arrays
obj2 = zeros(S,W);

% Generate the entries of obj1 and obj2.
for ii = 1:P
    for jj = 1:F
        for kk = 1:W
            obj1(ii,jj,kk) = pcost(jj,ii) +
tcost(ii)*distfw(jj,kk);
        end
    end
end

for ii = 1:S
    for jj = 1:W
        obj2(ii,jj) = distsw(ii,jj)*sum(d(ii,:).*tcost);
    end
end

% Combine the entries into one vector.
obj = [obj1(:);obj2(:)]; % obj is the objective function vector
```

Matrix **Aineq** holding the coefficients for the inequality constraints is large and sparse, so use a sparse matrix instead of a dense one.

```
matwid = length(obj);

Aineq = spalloc(P*F + W,matwid,P*F*W + S*W); % Allocate sparse Aeq
bineq = zeros(P*F + W,1); % Allocate bineq as full

% Zero matrices of convenient sizes:
clearer1 = zeros(size(obj1));
clearer12 = clearer1(:);
clearer2 = zeros(size(obj2));
clearer22 = clearer2(:);

% First the production capacity constraints
counter = 1;
for ii = 1:F
    for jj = 1:P
        xtemp = clearer1;
        xtemp(jj,ii,:) = 1; % Sum over warehouses for each product
and factory
        xtemp = sparse([xtemp(:);clearer22]); % Convert to sparse
        Aineq(counter,:) = xtemp'; % Fill in the row
        bineq(counter) = pcap(ii,jj);
        counter = counter + 1;
    end
end
```

```
% Now the warehouse capacity constraints
vj = zeros(S,1); % The multipliers
for jj = 1:S
    vj(jj) = sum(d(jj,:)./turn); % A sum of P elements
end

for ii = 1:W
    xtemp = clearer2;
    xtemp(:,ii) = vj;
    xtemp = sparse([clearer12;xtemp(:)]); % Convert to sparse
    Aineq(counter,:) = xtemp'; % Fill in the row
    bineq(counter) = wcap(ii);
    counter = counter + 1;
end
```

Matrix **Aeq** is for the coefficients of the equality constraints. Also stored as a sparse matrix

```
Aeq = spalloc(P*W + S,matwid,P*W*(F+S) + S*W); % Allocate as sparse
beq = zeros(P*W + S,1); % Allocate vectors as full

counter = 1;
% Demand is satisfied:
for ii = 1:P
    for jj = 1:W
        xtemp = clearer1;
        xtemp(ii,:,jj) = 1;
        xtemp2 = clearer2;
        xtemp2(:,jj) = -d(:,ii);
        xtemp = sparse([xtemp(:);xtemp2(:)]'); % Change to sparse
row
        Aeq(counter,:) = xtemp; % Fill in row
        counter = counter + 1;
    end
end
```

```
% Only one warehouse for each sales outlet:
for ii = 1:S
    xtemp = clearer2;
    xtemp(ii,:) = 1;
    xtemp = sparse([clearer12;xtemp(:)]'); % Change to sparse row
    Aeq(counter,:) = xtemp; % Fill in row
    beq(counter) = 1;
    counter = counter + 1;
end
```

```
intcon = P*F*W+1:length(obj);
lb = zeros(length(obj),1);
ub = Inf(length(obj),1);
ub(P*F*W+1:end) = 1;
[solution,fval,exitflag,output] =
intlinprog(obj,intcon,Aineq,bineq,Aeq,beq,lb,ub);
```