Development of programming models for analysis and solution of logistic problems

Santiago-Omar Caballero-Morales Corresp. 1

1 Postgraduate Department of Logistics and Supply Chain Management, Universidad Popular Autonóma del Estado de Puebla, Puebla, Puebla, Mexico

Corresponding Author: Santiago-Omar Caballero-Morales
Email address: santiagoomar.caballero@upaep.mx

Logistics is the aspect of the supply chain which is responsible of the efficient flow and delivery of goods or services from suppliers to customers. Because a logistic system involves specialized operations such as inventory control, facility location and distribution planning, the logistic professional requires mathematical, technological and managerial skills and tools to design, adapt and improve these operations. On the technological side, main research is focused on modeling and solving logistic problems by means of integer programming and meta-heuristics methods. However, in practice, the application of these methods for specific and large scale problems is restricted by the required knowledge and/or skills (e.g., programming proficiency, background in operations research, platform use). Also, the availability of test instances with specific features is limited. In this context, the present work describes the development of programming models to address the following aspects for research in the logistic and supply chain management fields: (1) development of test instances for routing and facility location problems with real geographical coordinates, (2) implementation of Euclidean, Manhattan and geographical arc length distance metrics for solving routing and facility location problems, (3) simulation of non-deterministic inventory control models, (4) importing/exporting and plotting of input data and solutions for analysis and visualization by third-party platforms, and (5) development of a nearest-neighbor meta-heuristic to provide very suitable solutions for vehicle routing and facility location problems, which are frequently studied for improvement of logistic systems. These developments can be used by the academic researcher or professional to get additional programming skills to solve supply chain problems.

Development of Programming Models for Analysis and Solution of Logistic Problems

Santiago-Omar Caballero-Morales

Universidad Popular Autonoma del Estado de Puebla, A.C., Postgraduate Department of Logistics and Supply Chain Management, Puebla, Mexico

Corresponding author:
Santiago-Omar Caballero-Morales

Email address: santiagoomar.caballero@upaep.mx

ABSTRACT

Logistics is the aspect of the supply chain which is responsible of the efficient flow and delivery of goods or services from suppliers to customers. Because a logistic system involves specialized operations such as inventory control, facility location and distribution planning, the logistic professional requires mathematical, technological and managerial skills and tools to design, adapt and improve these operations. On the technological side, main research is focused on modeling and solving logistic problems by means of integer programming and meta-heuristics methods. However, in practice, the application of these methods for specific and large scale problems is restricted by the required knowledge and/or skills (e.g., programming proficiency, background in operations research, platform use). Also, the availability of test instances with specific features is limited. In this context, the present work describes the development of programming models to address the following aspects for research in the logistic and supply chain management fields: (1) development of test instances for routing and facility location problems with real geographical coordinates, (2) implementation of Euclidean, Manhattan and geographical arc length distance metrics for solving routing and facility location problems, (3) simulation of non-deterministic inventory control models, (4) importing/exporting and plotting of input data and solutions for analysis and visualization by third-party platforms, and (5) development of a nearest-neighbor meta-heuristic to provide very suitable solutions for vehicle routing and facility location problems, which are frequently studied for improvement of logistic systems. These developments can be used by the academic researcher or professional to get additional programming skills to solve supply chain problems.

INTRODUCTION

Research on supply chain management (SCM) and logistics has been performed following quantitative (simulation, mathematical modeling and optimization) and qualitative (case study, interviews, empirical studies) approaches (Sachan A and Datta S, 2005). These approaches have been performed on the different logistic processes of the SC such as inventory control, transportation, production and distribution (Lloyd C et al., 2013; Kuczyńska-Chalada M et al., 2018). Transportation has been reported as the largest research topic in logistics (Daugherty P et al., 2017).

With the rise of Industry 4.0 (which is associated to "Internet of Things" or "SMART systems") logistics faces the challenge to control the entirety of logistic activities of enterprises cooperating in supply and logistic chains (Kuczyńska-Chalada M et al., 2018). In this context, the use of (a) embedded systems and (b) intelligent systems are considered as vital resources to achieve smart and autonomous flow of raw materials, work-in-process, and distribution of end products throughout the SC in accordance to human planning (Lloyd C et al., 2013; Kuczyńska-Chalada M et al., 2018).

The development of intelligent systems is based on quantitative research as it involves optimization, mathematical modeling, simulation and statistical assessment. A key resource for these tasks is the availability of specialized data for analysis and testing. Thus, research on state-of-the-art systems for transportation planning is supported by available databases of test sets (instances) such as TSPLIB (Reinelt G, 1991, 1997) for the Traveling Salesman Problem,

CVRPLIB (Uchoa E et al., 2017; Oliveira D et al., 2019) for the Vehicle Routing Problem, and SJC/p3038 sets (Chaves AA and Nogueira-Lorena LA, 2010; Nogueira-Lorena LA, 2007) for Facility Location Problems.

However, not all databases consider the different aspects of real logistics problems such as specific demand/location patterns and/or distance metrics. Also, in practice, the development and implementation of specific resources and solving methods is restricted by the required technical knowledge or skills associated to programming platforms and mathematical modeling. As presented in (Rao K et al., 1998) the use of software programs, computer programming and spreadsheet modeling, are effective problem-solving tools within logistic education. Hence, universities have revised their programs to provide qualified logistic professionals with these tools (Erturgut R, 2011).

Among the extensive educational resources for the logistic professional, few of them provide detailed examples of implementation. While the methodological steps are frequently described, source data such as software models, programming code, and test sets, are not explicitly available for sharing. Also, license restrictions regarding the implementation software avoids the use of some simulation models for commercial purposes. The high costs of some of these licenses restrict their use by freelance professionals, micro, small and medium enterprises, which have limited economic resources.

In this context, the present work describes the development of open source programming code for the researcher and professional in logistics and supply chain management. Particularly, the following aspects are addressed:

- development of test instances for routing and facility location problems with real geographical coordinates;
- implementation of Euclidean, Manhattan and geographical arc length distance metrics for solving routing and facility location problems (symmetric and asymmetric metrics are considered);
- importing/exporting and plotting of input data and solutions for analysis and visualization by third-party platforms;
- simulation of non-deterministic inventory control models for assessment of supply strategies;
- development of a nearest-neighbor meta-heuristic to provide very suitable solutions for vehicle routing and facility location problems.

As complementary material, two problems are studied and solved with these resources: (a) an integrated inventory – vehicle routing problem, and (b) a multi-facility location problem with inventory management. Implementation of the codes was performed through the open source programming software GNU Octave (Eaton JW et al., 2018). The advantage of this software is that it runs on GNU/Linux, macOS, BSD, and Windows, and it is compatible with many MATLAB scripts. Also, its language syntax makes it easily adaptable to other languages such as C++.

DEVELOPMENT OF TEST INSTANCES

The development of location data is important to test solving methods or algorithms for facility location and vehicle routing problems. To generate location data associated to real geographical coordinates the first step is to understand the coordinate system of the real world. Figure 1 presents the ranges for longitude and latitude coordinates (λ and ϕ respectively) of the world map.

With these ranges the second step consists on generating location data within specific regions. In example, consider the region delimited by $\lambda \in$[-102,-100] and $\phi \in$[20,22]. A set of locations within this region can be generated through two approaches:

- First, by adapting an already existing set of locations (standard test instance). There are available different sets of test instances for research within the distribution field. However,

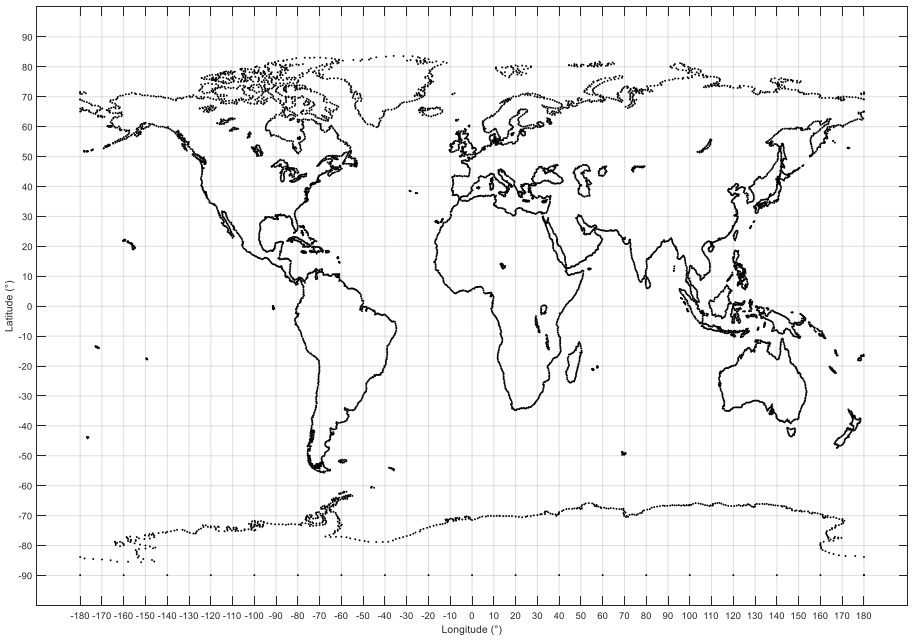


Figure 1. World map displaying the range for longitude (λ) and latitude (ϕ) coordinates in degrees.

many of them are not adjusted to geographic coordinates. This can affect the process to estimate distance metrics in kilometers. Figure 2 presents the visualization of coordinates from the instance *d493* of the database TSPLIB (Reinelt G, 1997). As presented, the range of the x and y axes are different from those presented in Figure 1. These non-geographical coordinates can be converted by using the following equation:

$$v' = \left(\frac{max_{new} - min_{new}}{max_{old} - min_{old}} \right) \times (v - max_{old}) + max_{new}. \quad (1)$$

Where v is the value within the old range and v' is the converted value within the new range. Figure 2 presents the programming code developed to convert the coordinates of *d493.tsp* into geographic coordinates.

- Second, by using a probability function. In this case, random coordinates can be generated by using distributions such as the *uniform* distribution with parameters a and b which represent the minimum and maximum values for the generation range, or the *normal* distribution where a and b represent the mean and standard deviation values within the generation range. Figure 3 presents the code developed to generate N=493 geographic coordinates considering these distributions. Note that a set of specific N coordinates cannot solely be generated by adaptation of existing data. In such case, a mixture of the codes presented in Figure 2 and 3 can provide a required set of N coordinates.

In addition to coordinate data, information regarding demands of locations and capacities of vehicles and warehouses must be generated for routing and coverage problems. By assuming a homogeneous fleet / set of warehouses a unique value is assigned to each of these elements. However, this does not necessarily apply to the demands of the locations to be served. Some instances have considered homogeneous demand (same value for all locations). However, in practice, this is not considered. Figure 4 presents an additional code to generate random demand for each location. Note that, within the demand context, the first location is considered to be the central depot or warehouse, thus, demand is equal to zero.

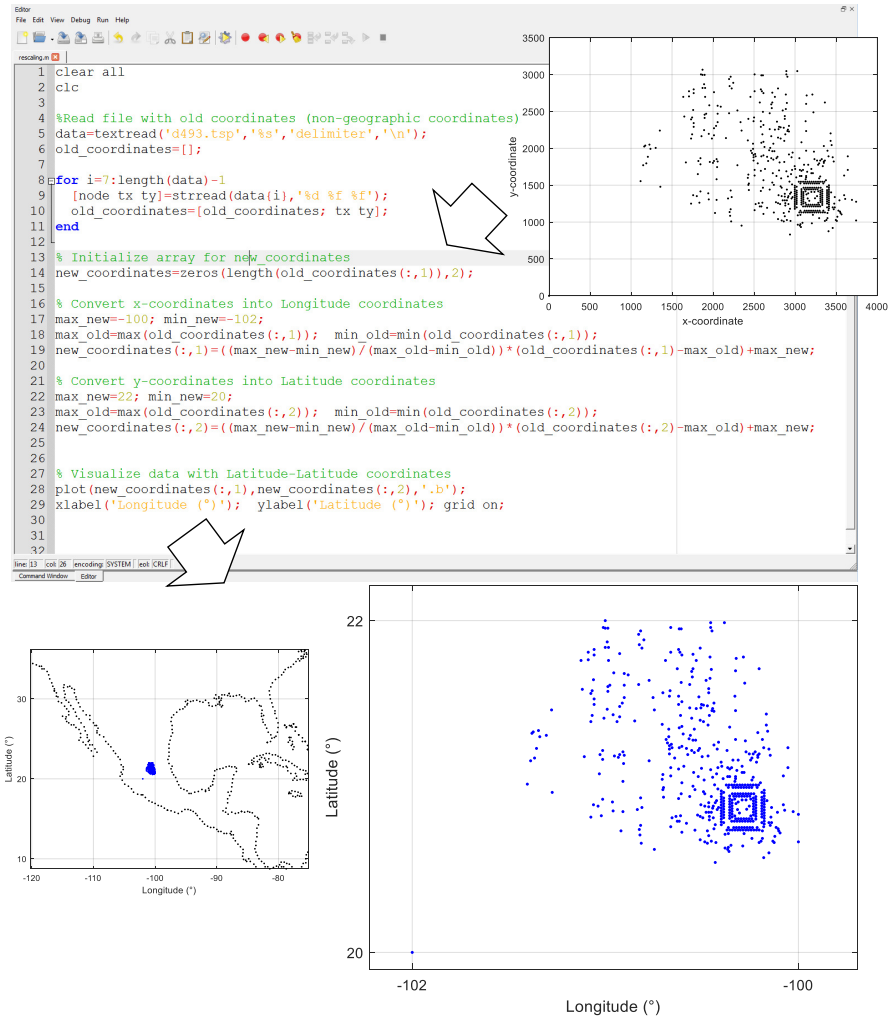


Figure 2. Programming Model 1: Re-scaling and plotting of existing location data.

DISTANCE METRICS

Within distribution problems, a metric to determine the suitability of a route over other routes is required. The most common criteria to optimize distributions problems is to minimize the metric of distance which is positively associated to transportation costs. There is a wide set of distance metrics, being the *Euclidean* distance the most commonly used. Other metrics, such as *Manhattan* distance or *arc length* are more closely associated to real-world contexts (Reinelt G, 1997). Figure 5 shows the details of these distance metrics together with their implementation code. It is important to remember that distance is estimated between two points i and j, thus, the x and y coordinates, or longitude (λ) and latitude (ϕ) coordinates, must be known in advance.

An important resource for any distribution problem is known as the distance matrix A which stores all distances between each location i and j within the distribution network. Thus, this matrix, of dimension $N \times N$ (where N is the number of locations, including the central depot) stores each d_{ij} data where $i, j = 1,\ldots, N$ and $d_{ii} = d_{jj} = 0$ (the distance between each point and itself is zero). The code presented in Figure 5 estimates also the symmetric distance matrix ($d_{ij} = d_{ji}$) for each type of metric.

By considering the converted coordinates into longitude and latitude degrees, the *Euclidean* and *Manhattan* distances provide similar values. In this regard, these values do not represent kilometers. In order to obtain a distance in kilometers, the *arc length* metric is considered. This metric considers the spherical model of the Earth's surface which has a radius of 6371 kilometers.

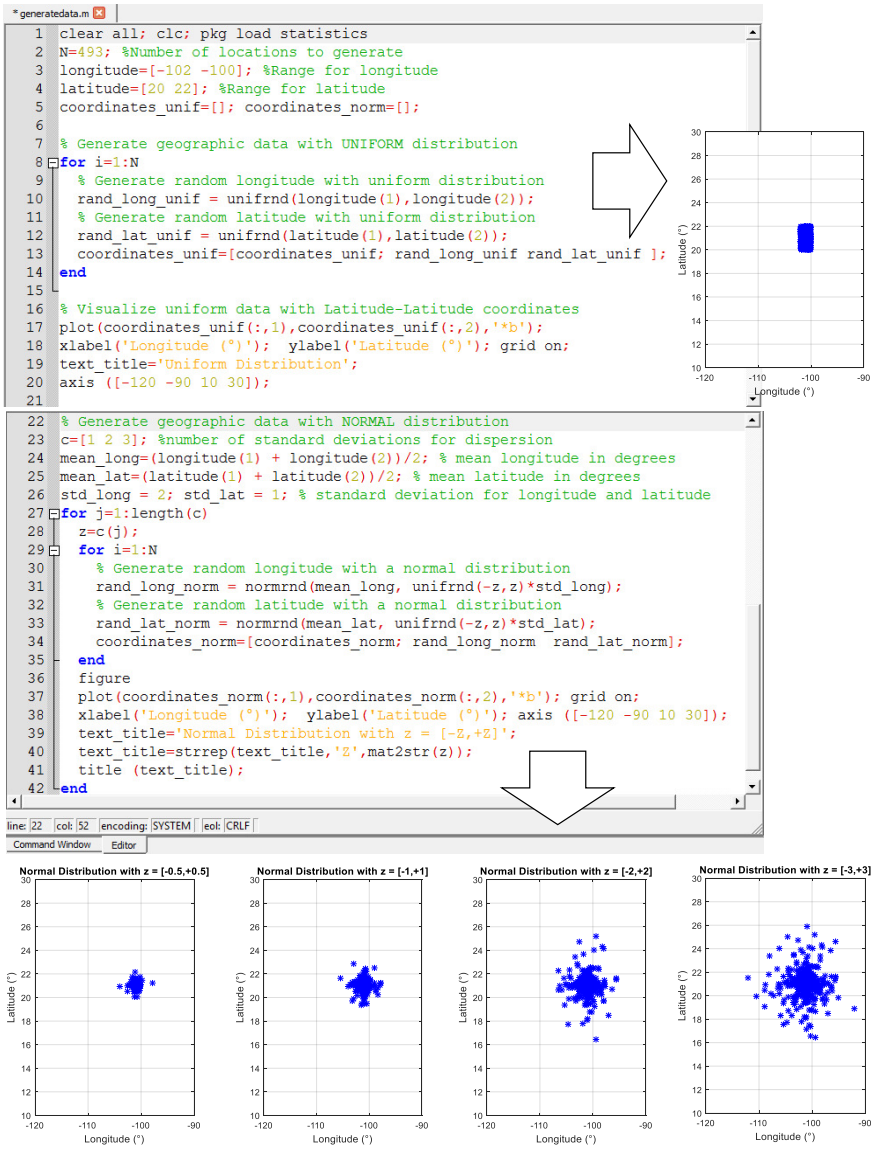


Figure 3. Programming Model 2: Generation of N *uniform* and *normal* random geographic coordinates and plotting.

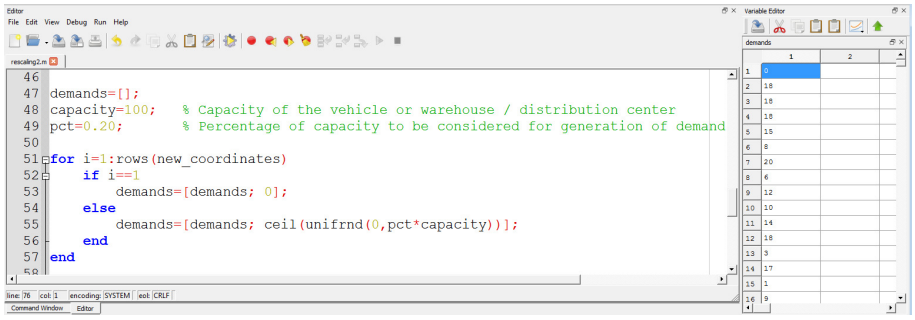


Figure 4. Programming Model 3: Generation of N random demand data.

For this, the coordinates in latitude and longitude degrees are converted into radians by a factor of $\pi/180^o$. This leads to a symmetrical distance matrix in kilometers.

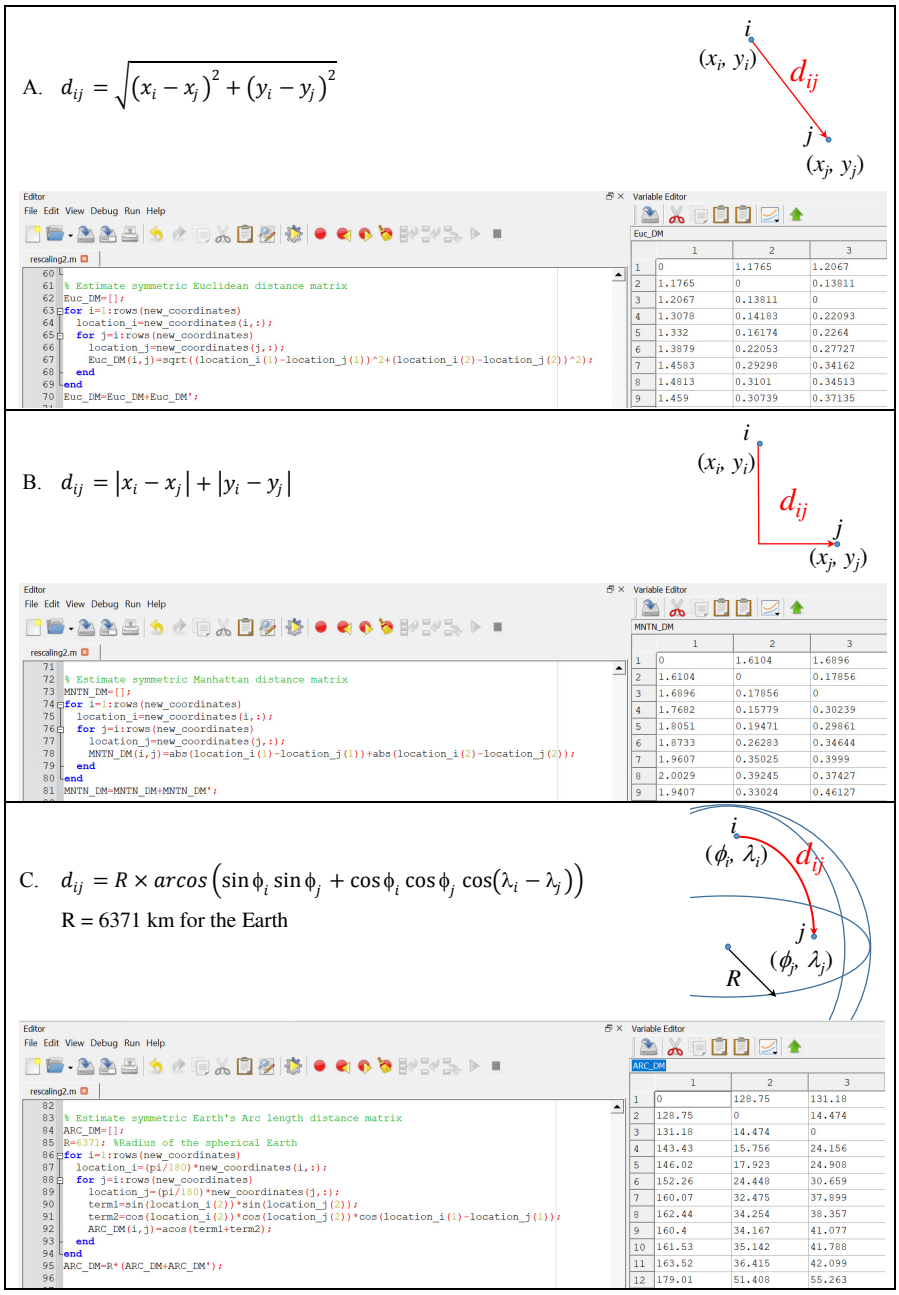


Figure 5. Programming Model 4: (A) Euclidean, (B) Manhattan, and (C) Arc length distances.

It is important to note that an approximation of the *Manhattan* distance can be estimated in terms of the *arc length* or *Euclidean* distance by considering trigonometry and right triangles theory. By knowing the angle θ between the hypotenuse and one of the catheti, and the length or magnitude of the hypotenuse (i.e., *Euclidean* or *arc length* d_{ij}), the length of both catheti can be estimated as:

$$c_1 = d_{ij} \times sin(\theta), \tag{2}$$

$$c_2 = d_{ij} \times cos(\theta). \tag{3}$$

Thus, if the *Euclidean* or *arc length* metric is available, then the *Manhattan* distance can be estimated as $c_1 + c_2$. Different estimates can be obtained based on the assumed value for θ.

Finally, an asymmetric distance matrix ($d_{ij} \neq d_{ji}$) can be constructed by adjusting the code described in Figure 5 to represent $d_{ji} = d_{ij} + unifrnd(0, mean(d_{ij}))$. Note that this operation modifies d_{ji} by adding to d_{ij} a random uniform value within the range from 0 to the mean value of all distances within A (although a different random value can be added).

INVENTORY CONTROL

Inventory management is an important aspect of distribution as proper inventory levels are required to ensure the constant supply of goods. This however must comply with restrictions to avoid unnecessary costs associated to inventory supply processes.

The concept of an "economic order quantity" (EOQ) has been widely used to define the optimal lot size required to minimize operational costs associated to ordering and keeping goods through the supply chain. Defining this lot size is a complex task due to the different variables involved in the inventory supply processes such as costs, delivery times, planning horizon, cycle time, stock-out costs and probabilities, service levels, demand patterns (Thinakaran N et al., 2019).

Thus, different mathematical models have been developed to determine the EOQ considering these multiple variables (Thinakaran N et al., 2019; Braglia M et al., 2019; De SK and Sana SS, 2015; Hovelaque V and Bironneau L, 2015). Two of the most useful models for inventory control are the continuous and periodic review models with non-deterministic demand. These models, also known as the *(Q, R)* (Hariga M, 2009; Adamu I, 2017) and P (Lieberman G and Hillier F, 2000) models respectively, are analyzed in the following sections.

Continuous Review Model

The *(Q, R)* model considers the costs and variables described in Table 1.

Table 1. Parameters and variables of the *(Q, R)* inventory control model.

Parameter or Variable	Description
C_o	Order cost per lot Q.
C_h	Holding cost per unit of product in inventory.
p	Stock-out cost per unit of product.
Q	Economic order quantity (EOQ)
R	Reorder point (level of inventory at which a lot of size Q must be ordered to avoid stock-out).
LT	Lead time
d	Average daily demand of products, or average demand of products on the smallest unit of time.
σ	Standard deviation of the average daily demand of products (it must be estimated on the same unit of time as d).
μ_{LT}	Average demand of products through the LT. It can be estimated as $d \times LT$ if both are under the same unit of time.
σ_{LT}	Standard deviation of products through the LT. It can be estimated as $\sigma \times \sqrt{LT}$ if both are under the same unit of time.
D	Cumulative demand through the planning horizon. Note that if d is a weekly demand, then $D = K \times d$ where K is the number of weeks within the planning horizon.
$L(z)$	Loss function associated to R.
C	Purchase cost per unit of product.

With these parameters and variables, lots of size Q are ordered through a planning horizon to serve a cumulative demand. Figure 6 presents an overview of the supply patterns considered by this model and its mathematical formulation to determine Q, R, and the *Total Inventory Costs* associated to it. As presented, at the beginning of the planning horizon, the inventory level starts at $R+Q$. This is decreased by the daily or weekly demand estimated as d. If historical

demand data is available, their use can provide a better estimate for d. As soon as the inventory level reaches R, the inventory manager must request an order of size Q because there is only stock to supply LT days or weeks. Here, it is important to observe that the inventory must be continuously reviewed or checked to accurately detect the re-order point R and reduce the stock-out risk during the LT.

Because uncertain demand is assumed, the inventory consumption rate is different throughout the planning horizon. Hence, R can be reached at different times. This is the reason to obtain different periods between inventory reviews.

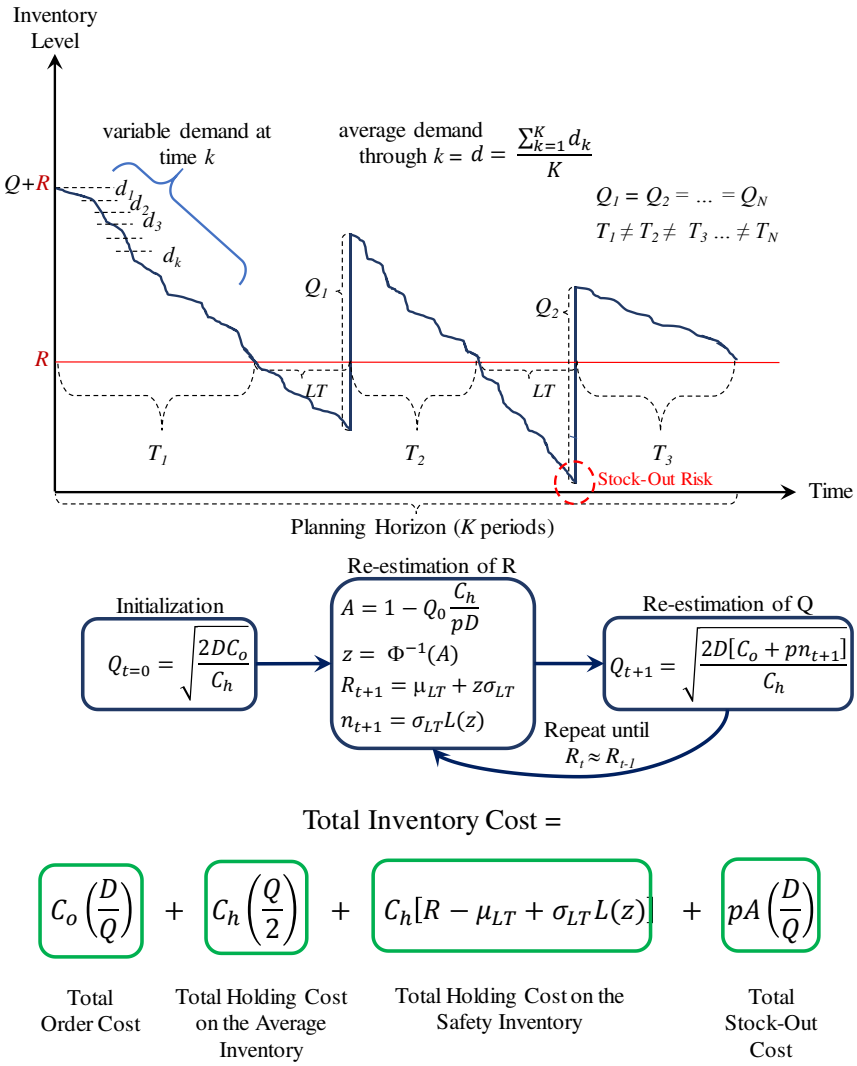


Figure 6. Inventory Supply Pattern under the Continuous Review *(Q, R)* Model.

For this model, Figure 7 presents the code to estimate the lot size Q, the reorder point R, and the *Total Inventory Cost*. How these parameters perform under a dynamic demand scenario is addressed through the visualization (plotting) algorithm also presented in Figure 7. This algorithm generates random demand test data (d_{test}) by the expression:

$$d_{test} = d \pm z\sigma, \tag{4}$$

where z is the number of standard deviations and is randomly generated within a specific range. Figure 8 presents examples considering $z \in$ [-3, +3], $z \in$ [-5, +5], and $z \in$ [-7, +7]. As presented, in all cases through the planning horizon there are no stock-out events. As variability increases

the consumption rate is faster, thus more orders must be requested. Thus, for $z \in$ [-3, +3] six orders were needed while for $z \in$ [-5, +5] seven orders were needed. Finally, for $z \in$ [-7, +7] nine orders were needed to avoid stock-out. This corroborates the validity of this method for inventory control under uncertain demand.

```
clear all; clc; pkg load statistics
d = 10; std = 6; %units per day of average demand and standard deviation
z = 3;           %standard deviations for demand variability
K = 365;         %working days through the planning horizon
D = K*d;         %cumulative demand
%         C      Co    Ch   p
Costs = [250.0 1250.0 25.0 150.0];  % Inventory Associated Costs
LT = 7; T = 21;  %days of lead time and days between reviews
% ============= Determine parameters for (Q,R) model ========================
for i=1:20 %Number of iterations
  if i==1 % Initial Value for Q
    Q = sqrt((2*D*Costs(2))/Costs(3));
  else
    Q = sqrt((2*D*(Costs(2) + Costs(4)*n))/Costs(3));
  end
  A=1-Q*(Costs(3)/(Costs(4)*D));  Z=norminv(A); % Z for Continuous Review Model
  R = d*LT + Z*std*sqrt(LT);
  L = stdnormal_pdf(Z)-Z*(1-stdnormal_cdf(Z)); % Loss function
  n = L*std*sqrt(LT);
end
Q=ceil(Q); R=ceil(R);% round final values towards positive infinity
% ============== Determine Associated Total Inventory Cost ===================
TOC = (Costs(2)*D)/Q; % Total Order Cost
THC = (Costs(3)*Q)/2; % Total Holding Cost
TSS = Costs(3)*(R - d*LT + std*sqrt(LT)*L); % Total Holding Cost of Safety Stock
TSK = Costs(4)*A*(D/Q); % Total Stock-Out Cost
TC = TOC +THC +TSS +TSK ;
% =============   Test (Q,R) Parameters  ====================================
Inventory = Q+R; count_LT=0; inv_consumption = []; % Initialization
for i=1:K
  inv_consumption = [inv_consumption ; Inventory];
  dt=d+unifrnd(-z,z)*std; % Random daily demand within [-z, +z] std
  if dt<0
    dt = 0;
  end
  if Inventory - dt > R
    Inventory =Inventory - dt;
  else
    Inventory =Inventory - dt;
    count_LT=count_LT+1;
  end
  if count_LT == LT+1
    Inventory=Inventory+Q; count_LT=0;
  end
end
% =================        Visualization Code       ==========================
vect_R=ones(length(inv_consumption),1)*R;
plot(inv_consumption); hold on;  plot(vect_R,'-r'); axis([0 K 0 Q+R]);
xlabel('Working Days'); ylabel('Inventory level');
text_title='Continuous Review Strategy with [-z,+z], Q = QX, R = RX, and TC = TCX';
text_title=strrep(text_title,'z',mat2str(z));
text_title=strrep(text_title,'QX',mat2str(Q));
text_title=strrep(text_title,'RX',mat2str(R));
text_title=strrep(text_title,'TCX',mat2str(TC));
title(text_title);
```

Figure 7. Programming Model 5: Continuous Review *(Q, R)* Model and Visualization Code.

Periodic Review Model

The P model considers the costs and variables described in Table 2. Figure 9 presents an overview of the supply patterns considered by this model and its mathematical formulation to determine Q and the *Total Inventory Costs* associated to it. In contrast to the *(Q, R)* model, in the P model the inventory review is performed at fixed intervals of length T and Q is estimated considering the available inventory I at that moment. Thus, different lots of size Q are ordered depending of the available inventory at the end of the review period T. Figure 10 presents the code to estimate the lot size Q, the *Total Inventory Cost*, and the consumption and supply patterns for the P model.

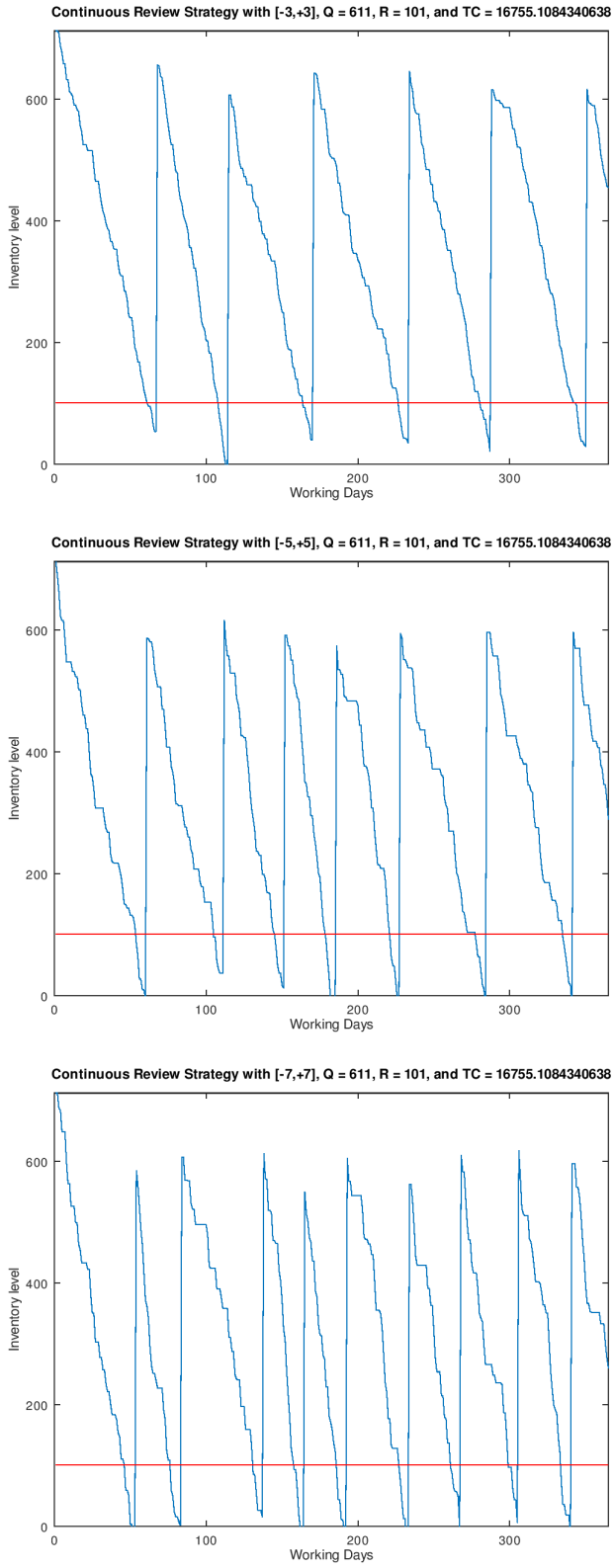


Figure 8. Performance of the Continuous Review *(Q, R)* Strategy with Different Variability Conditions.

Table 2. Main parameters and variables of the *(Q, R)* and *P* models

Variable	Description
C_o	Order cost per lot Q.
C_h	Holding cost per unit of product in inventory.
Q	Economic order quantity (EOQ)
LT	Lead time
T	Time between inventory reviews and $T > LT$
d	Average daily demand of products, or average demand of products on the smallest unit of time.
D	Cumulative demand through the planning horizon. Note that if d is a weekly demand, then $D = K \times d$ where K is the number of weeks within the planning horizon.
σ	Standard deviation of the average daily demand of products (it must be estimated on the same unit of time as d).
z	Number of standard deviations associated to a service level.

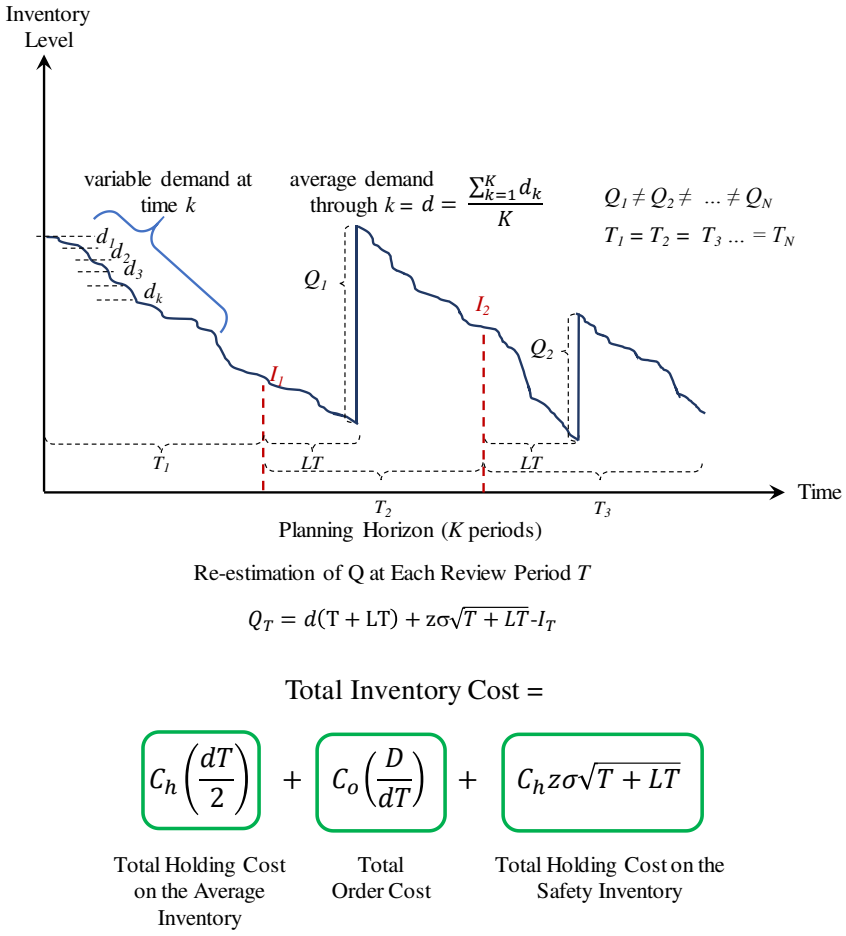


Figure 9. Inventory Supply Pattern under the Periodic Review *(P)* Model.

As in the case of the *(Q, R)* model, Figure 11 presents examples considering $z \in$ [-3, +3], $z \in$ [-5, +5], and $z \in$ [-7, +7] for the variable demand rate. Here, the advantages of the *(Q, R)* model are evident for demand with high variability. At the moment of review (at the end of T), the lot size $Q - I$ is estimated based on the historical data modeled through d and σ. This lot size must be able to cover the demand for the next period $T + LT$. However, this increases

```
clear all; clc; pkg load statistics
d = 10; std = 6; %units per day of average demand and standard deviation
z = 3;           %standard deviations for demand variability
K = 365;         %working days through the planning horizon
D = K*d;         %cumulative demand
SL = 0.985;      %service level of 98.5%
% Inventory associated costs
%         C      Co    Ch   p
Costs = [250.0 1250.0 25.0 150.0];  % Inventory Associated Costs
LT = 7; T = 21;  % days of lead time and days between reviews
Z = norminv(SL); % Z-value for Periodic Review Model
% ============= Determine parameters for P model ========================
I = 0; Q=d*(T+LT)+Z*std*sqrt(T+LT)-I; Inventory=Q; count_T=0; count_LT=0;
% ============== Determine Associated Total Inventory Cost ===================
TOC = (Costs(2)*D)/(d*T); % Total Order Cost
THC = (Costs(3)*(d*T))/2; % Total Holding Cost
TSS = Costs(3)*Z*std*sqrt(T+LT); % Total Holding Cost of Safety Stock
TC = TOC +THC +TSS;
% =============   Test P Parameters  ====================================
inv_consumption = [];
for i=1:K
  inv_consumption = [inv_consumption ; Inventory];
  dt=d+unifrnd(-z,z)*std; % Random daily demand within [-z, +z] std
  if dt<0
    dt = 0;
  end
  Inventory=Inventory-dt; count_T=count_T+1;
  if count_T == T
    I= Inventory; count_T=0;  count_LT=LT+1;
  end
  if count_LT>0
    count_LT=count_LT-1;
    if count_LT == 0
      Inventory = Inventory+(Q-I);
    end
  end
end
% =================        Visualization Code       ==========================
plot(inv_consumption); hold on; axis([0 K 0 Q]);
xlabel('Working Days'); ylabel('Inventory level');
text_title='Periodic Review Strategy with [-z,+z], Q = QX and TC = TCX';
text_title=strrep(text_title,'z',mat2str(z));
text_title=strrep(text_title,'QX',mat2str(Q));
text_title=strrep(text_title,'TCX',mat2str(TC));
title(text_title);
```

Figure 10. Programming Model 6: Periodic Review P Model and Visualization Code.

the stock out risk because ordering is restricted to be performed just at the end of T. Thus, if during that period the demand significantly increases (more than that modeled by σ), inventory can be consumed at a higher rate. For the considered example, the required service level was set to 98.5%. By assuming normality, the z-value associated to this probability is approximately 2.17. Thus, for demand with $z \in$ [-3, +3] standard deviations, the Q estimated by the P model is marginally able to keep the supply without stock-out. As presented in Figure 11, with higher variability there are stock-out events. Thus, it is recommended to re-estimate the Q parameter considering the updated demand patterns during the previous $T + LT$ (i.e., update d and σ). These observations are important while evaluating inventory supply strategies.

SOLVING METHODS FOR ROUTING AND FACILITY LOCATION PROBLEMS

Standard approaches of Operations Research (OR) such as Mixed Integer Linear Programming (MILP) or Dynamic Programming (DP) are often of limited use for large problems due to excessive computation time (Zäpfel G et al., 2010). Thus, meta-heuristics have been developed to provide near-optimal solutions in reasonable time for large production and logistic problems.

As introductory meta-heuristic, this work is considering two integrated local search algorithms with the fundamentals of GRASP (Greedy Randomized Adaptive Search Procedure) and Nearest-Neighbor Search (NNS). A complete review of more complex meta-heuristics and solving methods

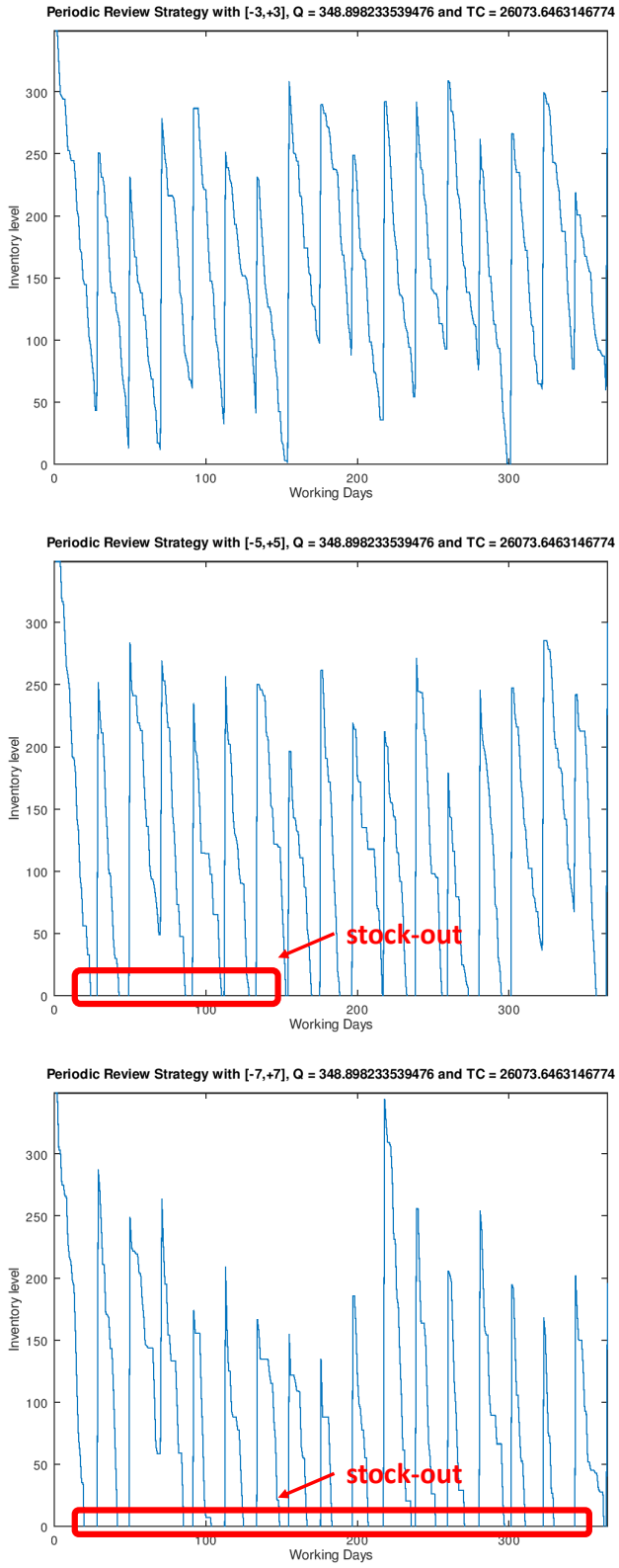


Figure 11. Performance of the Periodic Review P Strategy with Different Variability Conditions.

for different vehicle routing / facility location problems can be found in (Basu S et al., 2015; Prodhon C and Prins C, 2014; Bräysy O and Gendreau M, 2005a,b).

For the development of the integrated local search algorithms it is important to identify the characteristics of the vehicle routing / facility location problems and their solutions. This is discussed in the following sections.

Vehicle Routing Problem (VRP)

In this problem, a vehicle or set of vehicles departs from a single location where a depot or distribution center is established. These vehicles serve a set of locations associated to customers/suppliers to deliver/collect items.

If each vehicle has finite capacity then the vehicle's route can only visit those customer/supplier locations whose requirements do not exceed its capacity. This leads to define multiple routes to serve unique sets of customer/supplier locations where each location can only be visited once. Optimization is focused on determining the required routes and the visiting sequence to minimize traveling distance and/or costs. Figure 12 presents an overview of the capacitated VRP (also known as CVRP).

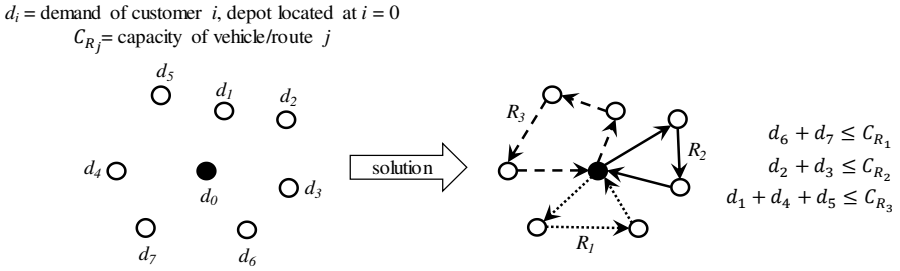


Figure 12. Characteristics of the capacitated vehicle routing problem (CVRP)

For the CVRP, two main features are required to provide a solution: partitioning and sequencing of minimum distance. Partitioning can be applied over a single total route to obtain sub-routes served by vehicles of finite capacity. Sequencing then can be applied over a set of customer locations to determine the most suitable order to reduce traveling time/distance. This can be applied on the single total route and on each sub-route.

In this work, a nearest neighbor search (NNS) approach is considered for the sequencing and partitioning algorithm. Figure 13 presents the sequencing code for the symmetric CVRPLIB instance *X-n219-k73.vrp* considering Euclidean distance. The instance consists of 219 nodes (i.e., 218 customer/demand locations and one central depot location) and homogeneous fleet with capacity of 3.

As presented, the algorithm performs a sequencing through the nearest or closest candidate nodes within a distance threshold given by the minimum distances plus the weighted standard deviation of the distances between all nodes or locations. Once that the total partial route of nearest nodes is obtained, it is partitioned through the function *partitioning2* whose code is presented in Figure 14. The result of the partitioning process leads to 73 sub-routes which is the optimal number as stated in the file name of the instance. Observe that another function, called *random_improvement* (see Figure 15) is performed on the total route and within the function *partitioning2* for each sub-route. This function consists of exchange and flip operators which are applied on sequences of nodes to improve the sequence obtained either by the nearest neighbor approach or by the partitioning process. Finally, Figure 16 presents the plotting code for the total route and the CVRP sub-routes.

An important aspect of any solving method, particularly meta-heuristics, is the assessment of its accuracy. The most frequently used metric for assessment is the % error gap, which is computed as:

$$\%e = \left(\frac{a-b}{b}\right) \times 100.0, \tag{5}$$

```
clear all; clc; pkg load statistics;
coords=[]; demands=[];

%============== Read File of Coordinate and Demand Data  ======================
data=textread('X-n219-k73.vrp','%s','delimiter','\n');
[s_1 s_2 nodes]=strread(data{4},'%s %s %d');
[s_1 s_2 capacity]=strread(data{6},'%s %s %d');
for i=8:8+nodes-1 % Lines of File with Coordinates Data
  [node tx ty]=strread(data{i},'%d %f %f');  coords=[coords; tx ty];
end
for i=8+nodes+1:length(data)-4 % Lines of File with Demand Data
  [node d]=strread(data{i},'%d %f');  demands=[demands; d];
end

% ============= Compute Euclidean Symmetric Distance Matrix ====================
Euc_DM=[]; Euc_DM0=[];
for i=1:length(coords(:,1))
  loc_i=coords(i,:);
  for j=1:length(coords(:,1))
    loc_j=coords(j,:);
    Euc_DM(i,j)=sqrt((loc_i(1)-loc_j(1))^2+(loc_i(2)-loc_j(2))^2);
  end
end

% ============= Compute Distance Statistics for Sequencing =====================
tp_0=sort(reshape(Euc_DM,1,nodes*nodes)); %Re-arrange into vector and sort
tp_1=tp_0(nodes+1:length(tp_0));      %Eliminate distance data of i=j
mn_dst=mean(tp_1); std_dst=std(tp_1); %Mean and standard deviation of Euc_DM
% ==============================================================================

absolute=Inf;
for max_iteras=1:500
  % ======= Nearest-Neighbor NN sequencing to obtain total single route ==========
  % different weights for the standard deviation
  w_std=[0.00001, 0.00005 0.0001 0.0005 0.001 0.005 0.01 0.05];
  cont_lim=1; % counter for selection of weights
  members=2:1:nodes; % vector with all remaining customer locations
  partial_route=[]; all_nodes_inserted=0;
  current_node=1; % The partial route starts at the "current" node "1"
  while all_nodes_inserted==0
    threshold= std_dst*w_std(cont_lim); cont_lim=cont_lim+1;
    if cont_lim>length(w_std) cont_lim=1; end
    distances=[];
    for j=1:length(members) % Compute distances from "current" node to remaining nodes
      distances(j)=Euc_DM(current_node,members(j));
    end
    if isempty(distances)==0 % There are remaining nodes to be inserted if distances is
      % not empty. Candidate remaining nodes are those within a distance threshold
      ind_temp=find(distances<=(min(distances))+threshold);
      % The "closest" node is randomly selected from the candidate remaining nodes
      closest_node=members(ind_temp(ceil(unifrnd(0,length(ind_temp)))));
      ind_ref=find(members==closest_node);
      % (1)"closest" node is removed from the remaining nodes,(2)"current" node is inserted
      % into the "partial" route,(3)"current" node is updated with the "closest" node
      members(ind_ref)=[];partial_route=[partial_route current_node];current_node=closest_node;
    else
      all_nodes_inserted=1; partial_route=[partial_route current_node]; %All nodes inserted
    end
  end
  partial_route(1)=[]; %Node "1" is removed from partial route
  [total_route distance_improved_route]=random_improvement(partial_route, Euc_DM);
  [CVRP_distance CVRP_routes]=partitioning2(total_route,capacity,demands,Euc_DM);

  if CVRP_distance< absolute
    best_absolute=[];
    best_absolute=CVRP_routes; %Updates best absolute CVRP solution
    absolute=CVRP_distance;    %Updates distance value of the best absolute CVRP solution
    fprintf('%d  CVRP distance = %1.4f \n',max_iteras, absolute);
    best_partial_route=[];
    best_partial_route=total_route;
  end
end
```

Figure 13. Programming Model 7: Nearest-neighbor algorithm for node sequencing.

where a is the result obtained with the considered solving method and b is the best-known solution. For the present example, the best-known solution is 117595.0 while the described NNS algorithm achieved a solution of $a = 119162.6799$. Hence, the error gap is estimated as 1.33%, which is very close to the best-known solution. Also, this result was obtained within reasonable time (117.046165 seconds).

Facility Location Problem (FLP)

In this problem, it is required to determine the most suitable location for a facility or set of facilities to serve a set of customers/suppliers. As in the CVRP, if capacity is considered for the facilities, multiple facilities are required to serve unique sets of customer/supplier locations where each location can only be served by a unique facility. If facilities are located at the location of minimum distance to all customers/suppliers, the FLP is known as the capacitated Weber problem (Aras N et al., 2007). However, if facilities are located at the average locations between

```
function [CVRP_distance matrix_CVRP_routes_nodes]=partitioning2(total_route,capacity,demands,Euc_DM)
  remaining_capacity=capacity; matrix_CVRP_routes_nodes=[];
  row=1; % number of sub-routes is initialized as "1"
  column=1; remaining_nodes=length(total_route);
% ================== Partitioning Process ====================
for uu=1:length(total_route)
    node=total_route(uu); d=demands(node);
    if d<=remaining_capacity
      matrix_CVRP_routes_nodes(row,column)=node; remaining_capacity=remaining_capacity-d;column=column+1;
    else
      remaining_capacity=capacity; row = row+1; column=1; % number of sub-routes is increased
      matrix_CVRP_routes_nodes(row,column)=node; remaining_capacity=remaining_capacity-d;column=column+1;
    end
end
% ========= Estimate Total Distance of All Sub-routes = CVRP Distance ==================
CVRP_routes=[]; CVRP_distance=0;
for k=1:length(matrix_CVRP_routes_nodes(:,1))
  sub_route_0=[]; sub_route_0=matrix_CVRP_routes_nodes(k,:); top=find(sub_route_0>0);
  sub_route=[]; sub_route=sub_route_0(1:top(length(top))); CVRP_routes=[CVRP_routes sub_route];
  [sub_route dist_sub_route]=random_improvement(sub_route, Euc_DM);
  for p=1:length(sub_route)
      matrix_CVRP_routes_nodes(k,p)=sub_route(p);
  end
  CVRP_distance=CVRP_distance+dist_sub_route;
end
```

Figure 14. Programming Model 8: Partitioning algorithm for sequence of nodes.

```
function [partial_route ref_distance]=random_improvement(partial_route, Euc_DM)

% reference distance to evaluate improvements
ref_partial_route=[1 partial_route 1]; %inserts node "1" at start/end to compute total distance
ref_distance=0;
for j=1:length(ref_partial_route)-1
  ref_distance=ref_distance+Euc_DM(ref_partial_route(j), ref_partial_route(j+1));
end
if length(partial_route)>1
    ref_distance;
    it= 150; % iterations to improve the partial route
    for iterations=1:it
        % Inversion/Flip Operator
        inv_partial_route=partial_route; inv_distance=0;
        pA=ceil(unifrnd(0.0*length(partial_route),length(partial_route)));
        pB=ceil(unifrnd(0.0*length(partial_route),length(partial_route)));
        while pA==pB pB=ceil(unifrnd(0.0*length(partial_route),length(partial_route))); end
        % Flip segment within positions pA and pB
        if pA<pB
            inv_partial_route(pA:pB)=flip(partial_route(pA:pB));
        else
            inv_partial_route(pB:pA)=flip(partial_route(pB:pA));
        end
        inv_partial_route=[1 inv_partial_route 1]; %inserts node "1" at start/end to compute total distance
        for j=1:length(inv_partial_route)-1
            inv_distance=inv_distance+Euc_DM(inv_partial_route(j),inv_partial_route(j+1));
        end
        % Exchange Operator
        exc_partial_route=partial_route; exc_distance=0;
        exc_partial_route(pA)=partial_route(pB);  exc_partial_route(pB)=partial_route(pA);
        exc_partial_route=[1 exc_partial_route 1];
        for j=1:length(exc_partial_route)-1
            exc_distance=exc_distance+Euc_DM(exc_partial_route(j),exc_partial_route(j+1));
        end
        % Decision process to evaluate improvement
        if inv_distance <= exc_distance
            candidate_route=inv_partial_route; candidate_distance=inv_distance;
        else
            candidate_route=exc_partial_route; candidate_distance=exc_distance;
        end
        if candidate_distance <= ref_distance % new solution is better than current solution
            partial_route=candidate_route; partial_route(1)=[]; partial_route(length(partial_route))=[];
            ref_distance=candidate_distance;
        end
    end
end
```

Figure 15. Programming Model 9: Random algorithm to improve sequencing of nodes.

all customers/suppliers (i.e., at the centroids), the FLP is known as the capacitated centered clustering problem (CCCP) (Negreiros M and Palhano A, 2006). Finally, if facilities are located at already defined median locations, the FLP is known as the capacitated p-median problem (CPMP) (Stefanello F et al., 2015). Figure 17 presents an overview of these variations of the FLP. It is important to observe that the feature of the locations has a direct effect on the solution.

Facility location problems are frequently solved through clustering or classification methods. Within this context, a nearest neighbor search (NNS) approach with an appropriate capacity-restricted assignment algorithm can provide very suitable solutions. Figure 18 presents the main code of the NNS algorithm for the CCCP instance *SJC1.dat* considering Euclidean distance. The instance consists of 100 nodes (i.e., 100 customer/demand locations), 10 centroids (i.e., 10 facilities to be located) and homogeneous capacity of 720.

The main code generates a feasible initial solution through the function *random_assignment* which is described in Figure 19. Randomness is considered when two or more nodes are located at the same distance from a centroid, and on the initial location of the centroids. This adds

```
%Plot routine to draw the partial route, the labels of each node, and the
%CVRP sub-routes
figure
hold on
label_nodes= 1:1:nodes; label_text = cellstr(num2str(label_nodes'));
for i=1:nodes
  plot(coords(i,1),coords(i,2),'*b');
  text(coords(i,1),coords(i,2),label_text(i),'FontSize',10);
end

for i=1:length(best_partial_route)-1
  xx=[coords(best_partial_route(i),1) coords(best_partial_route(i+1),1)];
  yy=[coords(best_partial_route(i),2) coords(best_partial_route(i+1),2)];
  plot(xx,yy,'--k','LineWidth',2);
end

figure
hold on

for k=1:length(best_absolute(:,1))
  sub_route=[]; sub_route=[1 best_absolute(k,:) 1];
  %Eliminate zeros from sub_route
  find_zeros=find(sub_route==0); sub_route(find_zeros)=[];
  for i=1:length(sub_route)-1
    xx=[coords(sub_route(i),1) coords(sub_route(i+1),1)];
    yy=[coords(sub_route(i),2) coords(sub_route(i+1),2)];
    plot(xx,yy,'-b');
  end
end
```

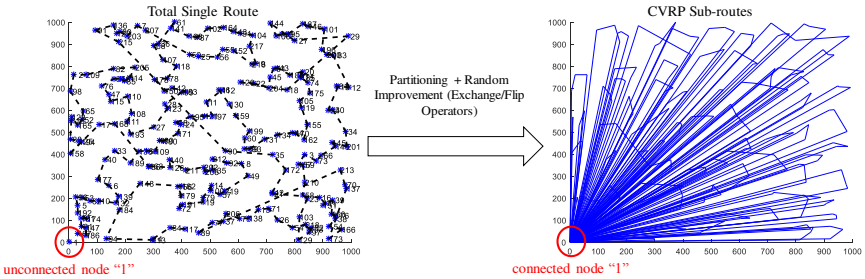


Figure 16. Programming Model 10: Plotting algorithm for visualization of routes.

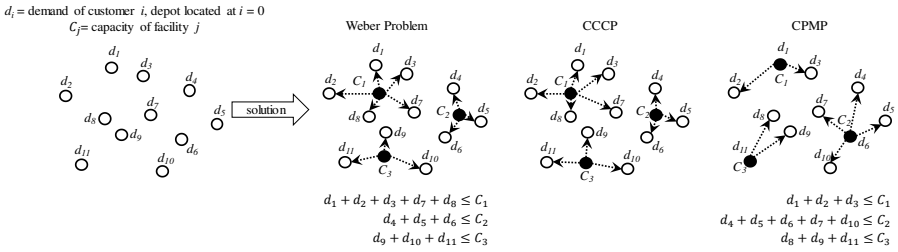


Figure 17. Characteristics of the capacitated facility location problem (CFLP)

flexibility to the assignment task and to the search mechanism of the NNS algorithm.

Then, this initial solution is improved through the function *update_centroids_assignment* which is described in Figure 20. As presented in the main code (see Figure 18) if the solution generated by *update_centroids_assignment* complies with all restrictions and its objective function's value is better than the reference value ($total\ distance < REF$), then it is stored in the arrays *best_assignments* and *best_centroids*.

When solving combinatorial problems, it is always recommended to verify the correctness of the solution. It is also recommended to know the convergence patterns of the solving algorithm. Both aspects provide insights regarding hidden mistakes in the programming code and deficiencies in the search mechanisms of the solving algorithm (e.g., fast or slow convergence to local optima). As presented in the main code (see Figure 18), at each iteration of the NNS the reference value is stored in the array *convergence*. Then, a verification routine is performed on the best solution which was found by the NNS algorithm. If the verification is successful the main code proceeds to plot the best found solution and the *convergence* data. Both plots for the CCCP instance *SJC1.dat* are presented in Figure 21.

```
clear all;  clc; pkg load statistics;

%Load set of coordinates of nodes from test instance file:
data=textread('SJC1.dat','%s','delimiter','\n'); coordinates=[]; demands=[];

for i=2:length(data)
    [coordx coordy capacity dem]=strread(data{i},'%f %f %d %d');
    coordinates=[coordinates;coordx coordy]; demands=[demands;dem];
end

P = 10; % Number of centroids - facilities
N = length(coordinates(:,1)); % Number of clients - nodes

% Generate random solution
[centroids assignments total_distance] = random_assignment(N,P,capacity, demands, coordinates);
REF = total_distance; % Initial reference value for assessment of convergence
fprintf('Iteration 0 with REF = %1.5f \n',REF);

ITERATIONS=50;  convergence=[];

% Local Search Metaheuristic for Improvement
for i=1:ITERATIONS
    [centroids assignments total_distance] = update_centroids_assignment(N,P,capacity, demands, coordinates, assignments);
    if total_distance < REF
        best_assignments=assignments;  best_centroids=centroids;  REF=total_distance;
        fprintf('Iteration %d with REF = %1.5f \n', i, REF);
    else
        if total_distance == Inf
            [centroids assignments total_distance] = random_assignment(N,P,capacity, demands, coordinates);
        end
    end
    convergence=[convergence;REF];
end

% Verification Routine
dems=[];    dists=[];
for i=1:P
    dem_ac=0;   coords=[];
    for j=1:N
        if best_assignments(j)==i
            coords=[coords;coordinates(j,:)]; dem_ac=dem_ac+demands(j);
        end
    end
    dems=[dems;dem_ac]; dist=0; cx=best_centroids(i,:);
    for j=1:length(coords(:,1))
        dist=dist+ sqrt ( (cx(1)-coords(j,1))^2 + (cx(2)-coords(j,2))^2);
    end
    dists=[dists; dist];
end
dists=sum(dists);
if length(find(dems>capacity))>0
    fprintf('Error \n');
else
    if dists == REF fprintf('Full Success \n');  else  fprintf('Partial Success \n');  end
end

figure; hold on;
for i=1:P
    coords=[];
    % Identify nodes assigned to facility i
    for j=1:N
        if best_assignments(j)==i
            coordsx=[best_centroids(i,1) coordinates(j,1)];  coordsy=[best_centroids(i,2) coordinates(j,2)];
            plot(coordsx,coordsy,'-b');
        end
    end
end

title('Assignments of Clients to Facilities','FontSize',12);
xlabel('X-Coordinates');    ylabel('Y-Coordinates');

figure; hold on;
title('Convergence of Local Search Metaheuristic','FontSize',12);
plot(1:1:ITERATIONS, convergence, 'ob');
xlabel('Iterations');   ylabel('Total Distance of Solution');
```

Figure 18. Programming Model 11: Local Search algorithm for CCCP.

Finally, the accuracy of this solution is assessed through Eq. (5). The NNS solution, which is plotted in Figure 21, leads to a total distance value of $a = 18043.76066$ while the best-known solution leads to $b = 17359.75$. Then, the error gap is estimated as 3.94% which is within the limit of 5.0% for the CCCP.

The consideration of randomness within the search mechanism of the NNS algorithm is common to most meta-heuristic solving methods. Convergence is dependent of this aspect, thus, assessment of meta-heuristics is performed considering different CCCP instances and multiple executions. If the coefficient of variability of results through multiple executions is within 0.20 it can be assumed that the meta-heuristic is stable. Figure 22 presents an example of this extended assessment scheme considering instances from the well-known SJC and DONI databases. After 20 executions of the meta-heuristic (as performed in (Chaves AA and Nogueira-Lorena LA, 2010, 2011)), the coefficient of variability (CV), the best, worst and average results are obtained to estimate their associated error gaps from the best-known (BK) solutions of the test instances. As observed, the CV is very small, less than 3.0% for all instances, hence the algorithm is stable. Thus, a very suitable solution can be obtained within a few executions of the algorithm, which for all instances (except *doni2*) is within the error gap of 5.0%.

Other assessment schemes can consider the execution time, time to best solution, and average computational times. This is specific for the assessment of new solving methods. Logistic research

```
function [centroids assignments total_distance] = random_assignment(N,P,capacity, demands, coordinates)
% P = number of facilities,   N = number of nodes
% Generate random solution
x_min=min(coordinates(:,1)); x_max=max(coordinates(:,1));
y_min=min(coordinates(:,2)); y_max=max(coordinates(:,2));

% Random centroids within the range of the nodes' coordinates
centroids=[];
for i=1:P
  centroids=[centroids; unifrnd(x_min,x_max) unifrnd(y_min,y_max)];
end

% Initialization of arrays
distance_matrix = zeros(P,N); % Distance matrix PxN
assignments = zeros(1,N);     % Assignment vector for each node within N
capacities = zeros(1,P);      % Cumulative capacity vector for each facility within P
distances = zeros(1,P);       % Cumulative distance vector for each facility within P

% Compute distance matrix PxN
for i=1:P
    pA=centroids(i,:);
    for j=1:N
        pB=coordinates(j,:); distance_matrix(i,j) = sqrt((pA(1)-pB(1))^2+(pA(2)-pB(2))^2);
    end
end

clients=1:1:N; % List of all nodes to be assigned to each facility within P (unassigned nodes)
% Perform assignment while there are unassigned nodes
while length(clients)>0
    a=min(min(distance_matrix));
    [row,col]=find(distance_matrix==a); % location of the minimum value within the distance matrix
    if length(row)>1
        % random selection if more than one node has the same minimum distance to a facility
        c=ceil(unifrnd(0,length(row)));
    else
        c=1;
    end
    % Identification of the node with the minimum distance to a facility
    facility = row(c);  node = col(c);
    % Perform assignment if the node is within the list of unassigned nodes
    % and there is available capacity within the facility
    if length(find(clients==node))>0 && capacities(facility)+demands(node)<=capacity
        assignments(node)=facility; % Perform assignment of the node to the facility
        capacities(facility)=capacities(facility)+demands(node); % Update capacity of the facility
        d=find(clients==node); clients(d)=[]; % Locate and remove assigned node from the list of unassigned nodes
        distance_matrix(facility,node)=Inf; % Remove the minimum value of the assigned node from the distance matrix
    else
        % If assignment cannot be performed, then remove the minimum value
        % for future assignment to a different facility
        distance_matrix(facility,node)=Inf; % Remove the current minimum value for future assignment
    end
end

% Compute total distance from current centroids to assigned nodes
for i=1:P
    dist=0; coords=[];
    % Identify nodes assigned to facility i
    for j=1:N
        if assignments(j)==i   coords=[coords;coordinates(j,:)]; end
    end
    if length(coords)>0
        cx=centroids(i,:);
        % Compute cumulative distance from centroid to all its assigned nodes
        for j=1:length(coords(:,1))
            cy=coords(j,:);  dist=dist+sqrt((cx(1)-cy(1))^2+(cx(2)-cy(2))^2);
        end
        distances(i)=dist;
    else
        % If no nodes are assigned to the facility, remove facility by making its distance equal to Inf.
        % This also marks this solution as not valid but suitable for improvement.
        distances(i)=Inf;
    end
end
total_distance=sum(distances);
```

Figure 19. Programming Model 12: Nearest-neighbor algorithm for initial assignment of nodes to closer centroids (CCCP instances).

is continuously extended in both important fields: (a) mathematical modeling of problems, and (b) adaptation and development of new solving methods. As presented, this work can provide the basis and resources for these research fields. In the following section, the application of the proposed programming models for two examples is described.

APPLICATION EXAMPLES

Example 1: Capacitated Vehicle Routing

The present example consists on developing a distribution scheme for 550 points within a city. The product to be distributed is insulin, which is important for people with diabetes or pancreatic dysfunction. Because people have different prescriptions for insulin use, and sometimes more than one person within a region needs insulin, the demand patterns may vary significantly from one point to another.

Due to the characteristics of the product, the distribution vehicle must have specific characteristics which involve costs. This restricts the frequency of distribution. Because there are only seven vehicles, the distribution must ensure fast response to the variable demand patterns without exceeding costs associated to inventory supply. In general, Figure 23 presents the methodological steps to solve this distribution problem. Initially, data regarding the capacity of the vehicles in terms of units of product must be estimated. For this case, the unit of product consists of a

```
function [centroids assignments total_distance] = update_centroids_assignment(N,P,capacity, demands, coordinates, assignments)
% P = number of facilities,   N = number of nodes / clients
% Initialization of arrays
distance_matrix = zeros(P,N); % Distance matrix PxN
capacities = zeros(1,P);      % Cumulative capacity vector for each facility within P
distances = zeros(1,P);       % Cumulative distance vector for each facility within P

%Backup of initial assignments
original_assignments = assignments;

%Compute centroids from current assignments
centroids=[];
for i=1:P
    coords=[];
    for j=1:N
        if assignments(j)==i  coords=[coords;coordinates(j,:)];  end
    end
    if length(coords)>0
        if length(coords(:,1))==1
            cx=coords; % If only one node is assigned, then this is the centroid's location
        else
            % If more than one node is assigned, their average coordinates are the centroid's location
            if length(coords(:,1))>1  cx=mean(coords);  end
        end
    else
        % If no nodes are assigned to the facility, compute the centroid's
        % location as the average of all coordinates
        cx=mean(coordinates);
    end
    centroids=[centroids;cx]; % Update centroids
end

% Compute distance matrix with updated centroids
for i=1:P
    pA=centroids(i,:);
    for j=1:N  pB=coordinates(j,:);  distance_matrix(i,j)= sqrt((pA(1)-pB(1))^2+(pA(2)-pB(2))^2); end
end

clients=1:1:N; % List of all nodes to be assigned to each facility within P (unassigned nodes)
% Initialization for update of assignments with updated centroids
assignments = zeros(1,N); alarm=0;

while length(clients)>0 && alarm == 0
    a=min(min(distance_matrix));
    if min(min(distance_matrix))==Inf
        alarm = 1;
    else
        [row,col]=find(distance_matrix==a); % location of the minimum value within the distance matrix
        % random selection if more than one node has the same minimum distance to a facility
        if length(row)>1   c=ceil(unifrnd(0,length(row)));  else c=1; end
        % Identification of the node with the minimum distance to a facility
        facility = row(c);  node = col(c);
        % Perform assignment if the node is within the list of unassigned nodes
        % and there is available capacity within the facility
        if length(find(clients==node))>0 && capacities(facility)+demands(node)<=capacity
            assignments(node)=facility; % Perform assignment of the node to the facility
            capacities(facility)=capacities(facility)+demands(node); % Update capacity of the facility
            % Locate and remove assigned node from the list of unassigned nodes
            d=find(clients==node); clients(d)=[];
            distance_matrix(facility,node)=Inf; % Completely remove assigned node from the distance matrix
        else
            % If assignment cannot be performed, then remove the minimum value
            % for future assignment to a different facility
            distance_matrix(facility,node)=Inf; % Remove the current minimum value for future assignment
        end
    end
end

if alarm==0
    distances=zeros(1,P);
    % Compute total distance from current centroids to assigned nodes
    for i=1:P
        dist=0; coords=[];
        % Identify nodes assigned to facility i
        for j=1:N
            if assignments(j)==i  coords=[coords;coordinates(j,:)];  end
        end
        if length(coords)>0
            cx=[];  cx=centroids(i,:);
            % Compute cumulative distance from centroid to all its assigned nodes
            for j=1:length(coords(:,1))
                cy=[];  cy=coords(j,:);  dist=dist+sqrt((cx(1)-cy(1))^2+(cx(2)-cy(2))^2);
            end
            distances(i)=dist;
        else
            % If no nodes are assigned to the facility, remove facility by making its distance equal to Inf.
            % This also marks this solution as not valid but suitable for improvement.
            distances(i)=Inf;
        end
    end
    total_distance=sum(distances);
else
    total_distance=Inf;
end

if total_distance==Inf
    fprintf('unfeasible solution - initialize with random centroids \n');
    assignments=original_assignments; centroids=[];
    x_min=min(coordinates(:,1)); x_max=max(coordinates(:,1));
    y_min=min(coordinates(:,2)); y_max=max(coordinates(:,2));
    for i=1:P  centroids=[centroids; unifrnd(x_min,x_max) unifrnd(y_min,y_max)];  end
    total_distance=Inf;
end
```

Figure 20. Programming Model 13: Nearest-neighbor algorithm for improvement of the assignment of nodes to closer centroids (CCCP instances).

pre-filled injection pen of 3 ml with 100 UI/ml (300 insulin units per pen) at an approximate cost of $C = 20$ USD. The vehicles' containers for transportation of this product has a capacity of 1.0 m^3. Based on the dimensions of the product's package, estimated as $0.10 \times 0.20 \times 0.085 = 0.0016$ m^3, the capacity is approximated as 1.0 / 0.0016 = 625 units.

Then, data regarding the demand patterns must be estimated to formulate the inventory supply strategy. Figure 24 presents the gathered information from each of the delivery points

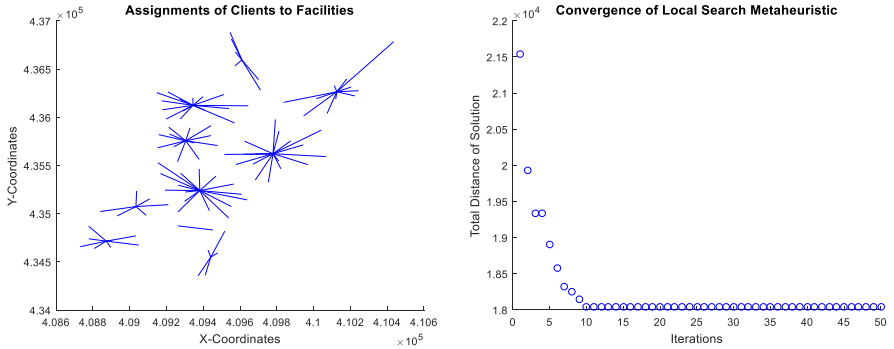


Figure 21. Results of the plotting algorithm: (A) assignments to centroids, (B) convergence of the search algorithm.

Instance	N	P	BK	Executions of the NNS Algorithm										CV	Error Gap (%)		
				1	2	3	4	5	6	7	8	9	10		Average	Best	Worst
SJC1	100	10	17359.75	19108.27	19233.84	19143.14	18031.42	19123.88	19393.45	18290.98	18174.94	18208.74	19200.32	0.02	8.00	3.87	11.72
SJC2	200	15	33181.65	34453.55	34263.04	33790.98	35184.40	34532.35	33966.42	34046.24	35374.06	35105.41	34846.92	0.02	3.81	0.72	7.21
SJC3a	300	25	45366.35	51494.75	47372.43	50184.63	47203.20	48262.37	48935.47	46790.16	48520.69	51850.40	49161.28	0.03	8.50	3.14	14.29
SJC3b	300	30	40695.46	44615.34	45297.04	45737.66	42897.90	43961.01	43714.43	44092.16	44020.33	42566.97	44201.15	0.02	8.14	4.60	13.34
SJC4a	402	30	61944.85	65574.16	66041.85	67444.02	65048.20	66532.38	65065.30	63879.09	65608.26	66293.10	64655.21	0.01	6.09	3.12	8.88
SJC4b	402	40	52214.55	54734.18	56138.30	54767.02	55450.18	56601.05	56984.65	56187.81	55050.67	55194.70	57268.72	0.01	6.96	4.83	9.68
doni1	1000	6	3021.41	3183.97	3184.49	3193.70	3200.12	3096.33	3228.53	3315.32	3254.76	3185.43	3184.49	0.02	6.35	2.45	9.73
doni2	2000	6	6080.70	6450.75	6771.04	6457.37	6520.50	6565.59	6943.25	6812.66	6604.67	6890.82	6462.06	0.02	9.00	6.02	14.19
doni3	3000	8	8446.08	9386.09	9164.45	9046.35	9210.88	9222.20	9284.65	9046.58	9078.15	9521.70	8764.82	0.02	9.05	3.77	12.76
doni4	4000	10	10854.48	11343.30	11554.67	12569.50	11727.37	12145.86	12228.30	11591.12	11749.20	12064.27	11900.86	0.03	9.52	4.50	15.80
doni5	5000	12	11134.94	12134.10	11826.50	12239.01	11795.63	11778.82	12228.90	11756.23	11630.03	11991.76	11688.43	0.02	7.46	4.45	13.58

Instance	N	P	BK	Executions of the NNS Algorithm									
				11	12	13	14	15	16	17	18	19	20
SJC1	100	10	17359.75	18414.13	18771.68	18980.26	18920.01	18238.01	18801.80	19309.07	19110.21	18456.53	18044.67
SJC2	200	15	33181.65	33984.90	34244.03	34194.04	33933.64	33419.28	35572.49	34591.19	34987.62	34192.40	34202.39
SJC3a	300	25	45366.35	49957.54	50486.07	49110.78	49326.85	49819.80	50118.63	49121.94	49122.90	48504.67	49125.72
SJC3b	300	30	40695.46	42757.81	43491.62	43258.93	45979.48	46123.80	44577.86	43476.13	42845.84	43770.52	42757.56
SJC4a	402	30	61944.85	64684.17	66330.04	66942.44	64252.03	65602.74	66930.28	64881.96	66203.34	66084.76	66330.83
SJC4b	402	40	52214.55	56098.36	55717.76	55381.78	55560.09	56688.00	56007.38	55433.25	57244.58	55346.98	55156.16
doni1	1000	6	3021.41	3178.30	3289.38	3280.55	3255.53	3222.71	3285.68	3272.36	3209.67	3095.41	3150.72
doni2	2000	6	6080.70	6716.37	6523.83	6587.83	6514.85	6514.85	6757.51	6769.05	6450.75	6446.69	6792.99
doni3	3000	8	8446.08	9460.70	9121.97	9488.05	8788.67	9107.96	9004.30	9324.03	9316.03	9523.56	9352.43
doni4	4000	10	10854.48	11446.71	11908.09	12536.06	11531.49	12016.75	12263.57	11809.81	11522.08	11916.93	11938.64
doni5	5000	12	11134.94	12647.01	11733.33	11695.76	12072.83	11774.91	11937.90	11931.33	11879.98	12157.47	12410.48

Figure 22. Extended Data for Assessment of NNS Algorithm for CCCP.

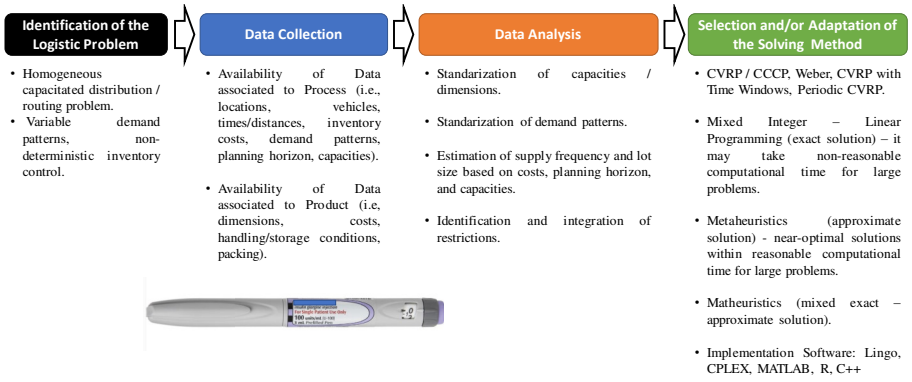


Figure 23. General methodological steps to address logistic problems - Example 1.

regarding demand patterns and locations through a planning horizon of two weeks. The main distribution center is located at λ = -98.245678 and ϕ = 19.056784.

Because most of the demands' CVs are bigger than 0.20, it is considered that variability is significant. Due to this, the non-deterministic periodic review *(P)* model is considered to estimate the net requirements for T = one period of two-weeks (15 days), and LT = one day ($1/15 = 0.07$ of one period of two-weeks). Note that under this assumption, $d = \mu$ from Figure 24 for the *(P)* model, and $z = 2.1700904$ for a service level of 98.5%.

Regarding costs, C_o and C_h are estimated from the point of view of the ordering entity, in

Demand Data

i	μ	σ	CV σ/μ	i	μ	σ	CV σ/μ	i	μ	σ	CV σ/μ	i	μ	σ	CV σ/μ	i	μ	σ	CV σ/μ
1	3	1	0.33	111	3	1	0.33	221	3	1	0.33	331	3	2	0.67	441	3	2	0.67
2	4	1	0.25	112	7	2	0.29	222	3	1	0.33	332	3	2	0.67	442	3	1	0.33
3	3	2	0.67	113	2	1	0.50	223	7	2	0.29	333	3	2	0.67	443	2	1	0.50
4	3	1	0.33	114	4	1	0.25	224	2	1	0.50	334	4	1	0.25	444	4	2	0.50
5	2	1	0.50	115	6	2	0.33	225	3	1	0.33	335	4	2	0.50	445	4	2	0.50
6	2	1	0.50	116	5	1	0.20	226	2	1	0.50	336	4	2	0.50	446	3	2	0.67
7	2	1	0.50	117	3	2	0.67	227	2	1	0.50	337	2	1	0.50	447	4	2	0.50
8	3	1	0.33	118	3	2	0.67	228	3	2	0.67	338	4	2	0.50	448	3	2	0.67
9	4	1	0.25	119	2	1	0.50	229	3	1	0.33	339	4	1	0.25	449	3	1	0.33
10	4	2	0.50	120	3	1	0.33	230	2	1	0.50	340	2	1	0.50	450	2	1	0.50
11	2	1	0.50	121	2	1	0.50	231	4	2	0.50	341	4	2	0.50	451	3	1	0.33
12	5	2	0.40	122	4	2	0.50	232	3	2	0.67	342	3	1	0.33	452	2	1	0.50
13	4	1	0.25	123	4	2	0.50	233	3	2	0.67	343	3	2	0.67	453	4	2	0.50
14	3	1	0.33	124	2	1	0.50	234	6	2	0.33	344	4	1	0.25	454	3	2	0.67
15	3	1	0.33	125	4	2	0.50	235	3	2	0.67	345	2	1	0.50	455	4	2	0.50
16	4	1	0.25	126	7	2	0.29	236	2	1	0.50	346	3	1	0.33	456	3	1	0.33
17	2	1	0.50	127	9	2	0.22	237	3	2	0.67	347	2	1	0.50	457	2	1	0.50
18	3	2	0.67	128	3	1	0.33	238	3	1	0.33	348	3	1	0.33	458	5	2	0.40
19	2	1	0.50	129	3	2	0.67	239	5	2	0.40	349	3	1	0.33	459	4	2	0.50
20	4	2	0.50	130	3	2	0.67	240	2	1	0.50	350	2	1	0.50	460	3	2	0.67
21	6	2	0.33	131	4	2	0.50	241	2	1	0.50	351	4	2	0.50	461	2	1	0.50
22	4	1	0.25	132	2	1	0.50	242	3	1	0.33	352	3	2	0.67	462	3	2	0.67
23	3	1	0.33	133	2	1	0.50	243	2	1	0.50	353	4	2	0.50	463	4	2	0.50
24	5	2	0.40	134	3	1	0.33	244	4	2	0.50	354	6	2	0.33	464	4	2	0.50
25	4	2	0.50	135	5	2	0.40	245	4	2	0.50	355	2	1	0.50	465	3	2	0.67
26	2	1	0.50	136	4	2	0.50	246	3	2	0.67	356	2	1	0.50	466	2	1	0.50
27	3	1	0.33	137	3	2	0.67	247	2	1	0.50	357	2	1	0.50	467	3	2	0.67
28	3	1	0.33	138	5	2	0.40	248	2	1	0.50	358	2	1	0.50	468	3	1	0.33
29	2	1	0.50	139	2	1	0.50	249	4	2	0.50	359	6	2	0.33	469	3	1	0.33
30	3	1	0.33	140	3	2	0.67	250	4	1	0.25	360	3	2	0.67	470	2	1	0.50
31	3	2	0.67	141	4	1	0.25	251	2	1	0.50	361	2	1	0.50	471	4	2	0.50
32	4	1	0.25	142	2	1	0.50	252	3	1	0.33	362	2	1	0.50	472	4	2	0.50
33	4	1	0.25	143	2	1	0.50	253	7	2	0.29	363	4	2	0.50	473	3	2	0.67
34	2	1	0.50	144	2	1	0.50	254	3	2	0.67	364	5	2	0.40	474	5	1	0.20
35	2	1	0.50	145	3	2	0.67	255	3	1	0.33	365	5	1	0.20	475	3	1	0.33
36	4	2	0.50	146	4	2	0.50	256	3	1	0.33	366	2	1	0.50	476	3	1	0.33
37	3	2	0.67	147	4	1	0.25	257	2	1	0.50	367	3	1	0.33	477	2	1	0.50
38	7	2	0.29	148	2	1	0.50	258	4	1	0.25	368	3	1	0.33	478	4	2	0.50
39	2	1	0.50	149	3	1	0.33	259	4	2	0.50	369	2	1	0.50	479	3	2	0.67
40	3	2	0.67	150	2	1	0.50	260	4	2	0.50	370	2	1	0.50	480	3	1	0.33
41	3	1	0.33	151	4	2	0.50	261	4	2	0.50	371	3	2	0.67	481	4	2	0.50
42	3	2	0.67	152	4	2	0.50	262	4	2	0.50	372	3	2	0.67	482	3	2	0.67
43	2	1	0.50	153	3	2	0.67	263	3	1	0.33	373	4	2	0.50	483	3	2	0.67
44	6	1	0.17	154	2	1	0.50	264	5	2	0.40	374	4	2	0.50	484	4	2	0.50
45	2	1	0.50	155	3	2	0.67	265	3	2	0.67	375	3	1	0.33	485	3	2	0.67
46	3	2	0.67	156	3	1	0.33	266	4	2	0.50	376	3	2	0.67	486	4	1	0.25
47	3	2	0.67	157	4	2	0.50	267	4	2	0.50	377	2	1	0.50	487	3	1	0.33
48	3	1	0.33	158	4	1	0.25	268	4	2	0.50	378	4	2	0.50	488	3	1	0.33
49	3	1	0.33	159	2	1	0.50	269	7	2	0.29	379	6	2	0.33	489	2	1	0.50
50	3	2	0.67	160	3	1	0.33	270	2	1	0.50	380	3	2	0.67	490	2	1	0.50
51	3	1	0.33	161	4	2	0.50	271	4	2	0.50	381	2	1	0.50	491	3	1	0.33
52	3	2	0.67	162	4	2	0.50	272	4	1	0.25	382	4	2	0.50	492	4	1	0.25
53	4	1	0.25	163	4	2	0.50	273	3	1	0.33	383	3	1	0.33	493	3	2	0.67
54	4	2	0.50	164	3	2	0.67	274	3	2	0.67	384	5	2	0.40	494	4	2	0.50
55	5	1	0.20	165	7	2	0.29	275	2	1	0.50	385	4	1	0.25	495	2	1	0.50
56	4	1	0.25	166	3	1	0.33	276	3	2	0.67	386	3	1	0.33	496	2	1	0.50
57	3	1	0.33	167	4	1	0.25	277	4	2	0.50	387	3	1	0.33	497	4	2	0.50
58	3	1	0.33	168	4	2	0.50	278	3	1	0.33	388	2	1	0.50	498	5	2	0.40
59	5	2	0.40	169	4	2	0.50	279	4	2	0.50	389	5	2	0.40	499	3	2	0.67
60	4	2	0.50	170	4	2	0.50	280	4	2	0.50	390	3	2	0.67	500	4	2	0.50
61	2	1	0.50	171	4	2	0.50	281	2	1	0.50	391	4	2	0.50	501	2	1	0.50
62	2	1	0.50	172	3	1	0.33	282	5	2	0.40	392	4	2	0.50	502	4	1	0.25
63	4	2	0.50	173	3	2	0.67	283	2	1	0.50	393	4	2	0.50	503	6	2	0.33
64	4	2	0.50	174	3	1	0.33	284	3	2	0.67	394	4	2	0.50	504	2	1	0.50
65	4	1	0.25	175	3	2	0.67	285	5	1	0.20	395	2	1	0.50	505	3	2	0.67
66	2	1	0.50	176	3	1	0.33	286	4	1	0.25	396	3	2	0.67	506	3	2	0.67
67	2	1	0.50	177	4	2	0.50	287	4	1	0.25	397	3	2	0.67	507	2	1	0.50
68	5	1	0.20	178	2	1	0.50	288	3	2	0.67	398	2	1	0.50	508	4	1	0.25
69	2	1	0.50	179	2	1	0.50	289	3	2	0.67	399	4	2	0.50	509	3	2	0.67
70	4	1	0.25	180	3	1	0.33	290	2	1	0.50	400	4	1	0.25	510	4	2	0.50
71	5	2	0.40	181	3	2	0.67	291	5	1	0.20	401	4	2	0.50	511	6	2	0.33
72	4	2	0.50	182	2	1	0.50	292	3	2	0.67	402	4	2	0.50	512	3	1	0.33
73	5	2	0.40	183	4	2	0.50	293	4	2	0.50	403	3	2	0.67	513	2	1	0.50
74	3	1	0.33	184	4	1	0.25	294	3	2	0.67	404	3	1	0.33	514	2	1	0.50
75	3	1	0.33	185	4	2	0.50	295	3	1	0.33	405	2	1	0.50	515	3	2	0.67
76	4	1	0.25	186	4	1	0.25	296	6	2	0.33	406	2	1	0.50	516	4	2	0.50
77	3	2	0.67	187	2	1	0.50	297	2	1	0.50	407	2	1	0.50	517	6	2	0.33
78	2	1	0.50	188	3	1	0.33	298	4	1	0.25	408	3	1	0.33	518	2	1	0.50
79	2	1	0.50	189	2	1	0.50	299	4	1	0.25	409	2	1	0.50	519	4	2	0.50
80	3	1	0.33	190	4	1	0.25	300	2	1	0.50	410	2	1	0.50	520	5	1	0.20
81	3	1	0.33	191	2	1	0.50	301	2	1	0.50	411	3	2	0.67	521	4	2	0.50
82	7	2	0.29	192	4	2	0.50	302	5	2	0.40	412	3	1	0.33	522	4	2	0.50
83	6	2	0.33	193	4	2	0.50	303	3	1	0.33	413	2	1	0.50	523	3	1	0.33
84	3	1	0.33	194	2	1	0.50	304	3	2	0.67	414	4	2	0.50	524	3	1	0.33
85	3	1	0.33	195	4	2	0.50	305	4	1	0.25	415	5	1	0.20	525	4	1	0.25
86	2	1	0.50	196	3	1	0.33	306	4	1	0.25	416	2	1	0.50	526	3	1	0.33
87	2	1	0.50	197	2	1	0.50	307	2	1	0.50	417	7	2	0.29	527	3	1	0.33
88	3	1	0.33	198	2	1	0.50	308	4	2	0.50	418	3	1	0.33	528	4	2	0.50
89	3	1	0.33	199	2	1	0.50	309	5	2	0.40	419	3	2	0.67	529	3	2	0.67
90	4	2	0.50	200	2	1	0.50	310	4	1	0.25	420	3	1	0.33	530	2	1	0.50
91	3	1	0.33	201	4	2	0.50	311	3	2	0.67	421	3	2	0.67	531	3	1	0.33
92	3	1	0.33	202	3	2	0.67	312	2	1	0.50	422	5	1	0.20	532	3	2	0.67
93	2	1	0.50	203	4	2	0.50	313	4	2	0.50	423	2	1	0.50	533	3	1	0.33
94	3	2	0.67	204	3	2	0.67	314	3	2	0.67	424	4	2	0.50	534	3	1	0.33
95	3	2	0.67	205	3	2	0.67	315	4	1	0.25	425	5	2	0.40	535	4	2	0.50
96	3	2	0.67	206	5	1	0.20	316	4	2	0.50	426	3	2	0.67	536	2	1	0.50
97	6	2	0.33	207	4	2	0.50	317	4	1	0.25	427	4	2	0.50	537	3	2	0.67
98	2	1	0.50	208	3	1	0.33	318	4	1	0.25	428	3	2	0.67	538	2	1	0.50
99	3	1	0.33	209	4	2	0.50	319	4	2	0.50	429	2	1	0.50	539	4	2	0.50
100	4	1	0.25	210	4	1	0.25	320	3	1	0.33	430	7	2	0.29	540	4	1	0.25
101	2	1	0.50	211	3	1	0.33	321	2	1	0.50	431	5	2	0.40	541	2	1	0.50
102	3	2	0.67	212	3	2	0.67	322	2	1	0.50	432	2	1	0.50	542	4	2	0.50
103	2	1	0.50	213	3	1	0.33	323	3	1	0.33	433	5	2	0.40	543	6	2	0.33
104	5	2	0.40	214	4	2	0.50	324	3	2	0.67	434	2	1	0.50	544	2	1	0.50
105	4	2	0.50	215	2	1	0.50	325	3	2	0.67	435	3	1	0.33	545	4	2	0.50
106	4	2	0.50	216	3	1	0.33	326	3	2	0.67	436	3	1	0.33	546	3	1	0.33
107	3	2	0.67	217	7	2	0.29	327	4	2	0.50	437	3	1	0.33	547	5	2	0.40
108	3	2	0.67	218	2	1	0.50	328	2	1	0.50	438	4	2	0.50	548	3	2	0.67
109	4	2	0.50	219	3	1	0.33	329	2	1	0.50	439	3	2	0.67	549	2	1	0.50
110	3	2	0.67	220	4	2	0.50	330	4	1	0.25	440	3	1	0.33	550	8	2	0.25

Location Data

i	λ	ϕ	i	λ	ϕ	i	λ	ϕ	i	λ	ϕ	i	λ	ϕ
1	-98.2564	19.0307	111	-98.2423	19.0334	221	-98.2300	19.0568	331	-98.2272	19.0555	441	-98.2279	19.0168
2	-98.2142	19.0379	112	-98.2413	19.0116	222	-98.2304	19.0368	332	-98.2264	19.0247	442	-98.2242	19.0299
3	-98.2390	19.0366	113	-98.2372	19.0324	223	-98.2299	19.0328	333	-98.2273	19.0353	443	-98.2290	19.0077
4	-98.2118	19.0457	114	-98.1962	19.0283	224	-98.2242	19.0392	334	-98.2293	19.0408	444	-98.2338	19.0495
5	-98.2205	19.0363	115	-98.2198	19.0326	225	-98.2321	19.0391	335	-98.2223	19.0252	445	-98.2116	19.0362
6	-98.2249	19.0385	116	-98.2263	19.0367	226	-98.2212	19.0217	336	-98.2326	19.0398	446	-98.2208	19.0240
7	-98.2287	19.0467	117	-98.2111	19.0525	227	-98.2267	19.0305	337	-98.2278	19.0377	447	-98.2233	19.0260
8	-98.2302	19.0230	118	-98.2275	19.0158	228	-98.2295	19.0315	338	-98.2355	19.0134	448	-98.2272	19.0274
9	-98.2466	19.0374	119	-98.2211	19.0362	229	-98.2271	19.0216	339	-98.2268	19.0330	449	-98.2267	19.0412
10	-98.2255	19.0548	120	-98.2330	19.0253	230	-98.2279	19.0135	340	-98.2289	19.0761	450	-98.2265	19.0383
11	-98.2237	19.0365	121	-98.2351	19.0354	231	-98.2305	19.0314	341	-98.2313	19.0377	451	-98.2187	19.0250
12	-98.2248	19.0437	122	-98.2326	19.0359	232	-98.2215	19.0346	342	-98.2385	19.0795	452	-98.2332	19.0532
13	-98.2171	19.0351	123	-98.2283	19.0202	233	-98.2265	19.0125	343	-98.2349	19.0324	453	-98.2367	19.0314
14	-98.2292	19.0576	124	-98.2336	19.0436	234	-98.2353	19.0483	344	-98.2307	19.0391	454	-98.2256	19.0392
15	-98.2253	19.0236	125	-98.2141	19.0338	235	-98.2274	19.0474	345	-98.2617	19.0519	455	-98.2273	19.0372
16	-98.2233	19.0459	126	-98.2262	19.0571	236	-98.2099	19.0307	346	-98.2261	19.0303	456	-98.2252	19.0165
17	-98.2285	19.0321	127	-98.2257	19.0260	237	-98.2283	18.9996	347	-98.2161	19.0395	457	-98.2275	19.0400
18	-98.2086	19.0354	128	-98.2350	19.0529	238	-98.2192	19.0285	348	-98.2250	19.0408	458	-98.2250	19.0364
19	-98.2526	19.0379	129	-98.2263	19.0297	239	-98.2206	19.0314	349	-98.2140	19.0353	459	-98.2240	19.0400
20	-98.2246	19.0295	130	-98.2265	19.0426	240	-98.2225	19.0349	350	-98.2271	19.0383	460	-98.2314	19.0477
21	-98.2182	19.0374	131	-98.2307	19.0232	241	-98.2289	19.0347	351	-98.2270	19.0159	461	-98.2287	19.0145
22	-98.2374	19.0314	132	-98.2259	19.0285	242	-98.2258	19.0481	352	-98.2366	19.0381	462	-98.2230	19.0586
23	-98.2334	18.9998	133	-98.2244	19.0213	243	-98.2352	19.0125	353	-98.2272	19.0436	463	-98.2225	19.0410
24	-98.2246	19.0193	134	-98.2265	19.0200	244	-98.2256	19.0326	354	-98.2230	19.0342	464	-98.2260	19.0408
25	-98.2177	19.0469	135	-98.2268	19.0476	245	-98.2236	19.0210	355	-98.2338	19.0134	465	-98.2330	19.0504
26	-98.2278	19.0329	136	-98.2292	19.0480	246	-98.2162	19.0330	356	-98.2283	19.0587	466	-98.2239	19.0252
27	-98.2145	19.0369	137	-98.2268	19.0306	247	-98.2345	19.0366	357	-98.2279	19.0344	467	-98.2313	19.0443
28	-98.2193	19.0169	138	-98.2242	19.0421	248	-98.2238	19.0630	358	-98.2216	19.0365	468	-98.2261	19.0360
29	-98.2041	19.0561	139	-98.2267	19.0153	249	-98.2287	19.0352	359	-98.2027	19.0396	469	-98.2333	19.0359
30	-98.2300	19.0308	140	-98.2229	19.0492	250	-98.2192	19.0301	360	-98.2254	19.0241	470	-98.2563	19.0483
31	-98.2568	19.0247	141	-98.2368	19.0517	251	-98.2256	19.0391	361	-98.2356	19.0412	471	-98.2309	19.0559
32	-98.2192	19.0517	142	-98.1971	19.0419	252	-98.2253	19.0303	362	-98.2434	19.0659	472	-98.2349	19.0198
33	-98.2281	19.0178	143	-98.2475	19.0497	253	-98.2250	19.0437	363	-98.2248	19.0479	473	-98.2269	19.0456
34	-98.2236	19.0400	144	-98.2302	19.0372	254	-98.2346	19.0416	364	-98.2180	19.0602	474	-98.2299	19.0398
35	-98.2238	19.0435	145	-98.2287	19.0574	255	-98.2261	19.0497	365	-98.2545	19.0414	475	-98.2300	19.0004
36	-98.2235	19.0343	146	-98.2208	19.0452	256	-98.2295	19.0412	366	-98.2266	19.0342	476	-98.2249	19.0268
37	-98.2247	19.0624	147	-98.2294	19.0343	257	-98.2202	19.0289	367	-98.2236	19.0287	477	-98.2103	19.0283
38	-98.2285	19.0591	148	-98.2247	19.0366	258	-98.1985	19.0599	368	-98.2316	19.0424	478	-98.2276	19.0389
39	-98.2224	19.0494	149	-98.2299	19.0118	259	-98.2199	19.0411	369	-98.2263	19.0592	479	-98.2225	19.0384
40	-98.2280	19.0360	150	-98.2240	19.0334	260	-98.2250	19.0633	370	-98.2016	19.0526	480	-98.2171	19.0281
41	-98.2221	19.0387	151	-98.2289	19.0351	261	-98.2281	19.0448	371	-98.2345	19.0396	481	-98.2235	19.0360
42	-98.2434	19.0347	152	-98.2044	19.0545	262	-98.2147	19.0411	372	-98.2362	19.0326	482	-98.2197	19.0295
43	-98.2190	19.0376	153	-98.2166	19.0362	263	-98.2233	19.0254	373	-98.2267	19.0294	483	-98.2284	19.0409
44	-98.2138	19.0440	154	-98.2260	19.0190	264	-98.2258	19.0131	374	-98.2305	19.0535	484	-98.2217	19.0327
45	-98.2369	19.0437	155	-98.2262	19.0239	265	-98.2374	19.0333	375	-98.2224	19.0376	485	-98.2273	19.0470
46	-98.2285	19.0352	156	-98.2282	19.0332	266	-98.2370	19.0558	376	-98.2281	19.0396	486	-98.2325	19.0450
47	-98.2261	19.0318	157	-98.2258	19.0367	267	-98.2169	19.0300	377	-98.2262	19.0051	487	-98.2260	19.0374
48	-98.2298	19.0512	158	-98.2292	19.0051	268	-98.2565	19.0349	378	-98.2173	19.0482	488	-98.2224	19.0459
49	-98.2348	19.0548	159	-98.2358	19.0413	269	-98.2299	19.0436	379	-98.2226	19.0285	489	-98.2385	19.0414
50	-98.2446	19.0198	160	-98.2316	19.0375	270	-98.2361	19.0377	380	-98.2357	19.0380	490	-98.2191	19.0353
51	-98.2237	19.0197	161	-98.2052	19.0575	271	-98.2254	19.0509	381	-98.2322	19.0357	491	-98.2199	19.0408
52	-98.2063	19.0385	162	-98.2226	19.0527	272	-98.2442	19.0361	382	-98.2214	19.0431	492	-98.2084	19.0443
53	-98.2287	19.0365	163	-98.2267	19.0383	273	-98.2183	19.0509	383	-98.2314	19.0381	493	-98.2326	19.0192
54	-98.2193	19.0135	164	-98.2228	19.0339	274	-98.2169	19.0371	384	-98.2318	19.0366	494	-98.2498	19.0395
55	-98.2468	19.0447	165	-98.2250	19.0211	275	-98.2291	19.0336	385	-98.2167	19.0177	495	-98.2132	19.0284
56	-98.2279	19.0380	166	-98.2283	19.0473	276	-98.2217	19.0545	386	-98.2286	19.0458	496	-98.2215	19.0378
57	-98.2310	18.9975	167	-98.2282	19.0469	277	-98.2410	19.0496	387	-98.2243	19.0373	497	-98.2291	19.0328
58	-98.2230	19.0614	168	-98.2119	19.0503	278	-98.2265	19.0369	388	-98.2458	19.0270	498	-98.2148	19.0361
59	-98.2389	19.0463	169	-98.2295	19.0140	279	-98.2048	19.0366	389	-98.2289	19.0392	499	-98.2289	19.0427
60	-98.2266	19.0453	170	-98.2259	19.0240	280	-98.2374	19.0676	390	-98.2395	19.0042	500	-98.2386	19.0550
61	-98.2213	19.0392	171	-98.2322	19.0383	281	-98.2102	19.0139	391	-98.2312	19.0145	501	-98.2328	19.0331
62	-98.2233	19.0713	172	-98.2271	19.0339	282	-98.2275	19.0345	392	-98.2202	19.0141	502	-98.2165	19.0305
63	-98.2188	19.0340	173	-98.2233	19.0165	283	-98.2010	19.0451	393	-98.2314	19.0347	503	-98.2234	19.0305
64	-98.2263	19.0349	174	-98.2294	19.0366	284	-98.2015	19.0282	394	-98.2375	19.0148	504	-98.2282	19.0276
65	-98.2211	19.0518	175	-98.2319	19.0360	285	-98.2275	19.0346	395	-98.2325	19.0455	505	-98.2310	19.0130
66	-98.2180	18.9973	176	-98.2613	19.0557	286	-98.2190	19.0362	396	-98.2262	19.0271	506	-98.2400	19.0382
67	-98.2232	19.0373	177	-98.2274	19.0322	287	-98.2266	19.0153	397	-98.2238	19.0543	507	-98.2397	19.0318
68	-98.2443	19.0358	178	-98.2247	19.0805	288	-98.2333	19.0325	398	-98.2348	19.0362	508	-98.2286	19.0339
69	-98.2191	19.0493	179	-98.2248	19.0590	289	-98.2246	19.0359	399	-98.2168	19.0196	509	-98.2410	19.0624
70	-98.2248	19.0337	180	-98.2175	19.0111	290	-98.2258	19.0195	400	-98.2283	19.0362	510	-98.2031	19.0329
71	-98.2256	19.0437	181	-98.2394	19.0312	291	-98.2346	19.0412	401	-98.2326	19.0298	511	-98.2152	19.0432
72	-98.2329	19.0354	182	-98.2262	19.0366	292	-98.2253	19.0289	402	-98.2302	19.0465	512	-98.2498	19.0420
73	-98.2316	19.0358	183	-98.2271	19.0411	293	-98.2179	19.0121	403	-98.2475	19.0294	513	-98.2254	19.0217
74	-98.2220	19.0414	184	-98.2297	19.0281	294	-98.2274	19.0374	404	-98.2438	19.0311	514	-98.2286	19.0387
75	-98.2340	19.0432	185	-98.2172	19.0391	295	-98.2107	19.0308	405	-98.2115	19.0724	515	-98.2268	19.0409
76	-98.2393	19.0245	186	-98.2315	19.0340	296	-98.2287	19.0361	406	-98.2477	19.0458	516	-98.2270	19.0376
77	-98.2477	19.0408	187	-98.2243	19.0361	297	-98.2335	19.0407	407	-98.2235	19.0496	517	-98.2232	19.0218
78	-98.2216	19.0356	188	-98.2174	19.0321	298	-98.2182	19.0337	408	-98.2178	19.0349	518	-98.2243	19.0295
79	-98.2134	19.0348	189	-98.2306	19.0467	299	-98.2282	19.0260	409	-98.2188	19.0140	519	-98.2171	19.0357
80	-98.2329	19.0461	190	-98.2250	19.0282	300	-98.2320	19.0362	410	-98.2278	19.0339	520	-98.2324	19.0363
81	-98.2333	19.0303	191	-98.2338	19.0382	301	-98.2367	19.0377	411	-98.2270	19.0356	521	-98.2301	19.0307
82	-98.2293	19.0352	192	-98.2287	19.0420	302	-98.2284	19.0291	412	-98.2258	19.0323	522	-98.2204	19.0392
83	-98.2256	19.0522	193	-98.2286	19.0383	303	-98.2260	19.0375	413	-98.2078	19.0473	523	-98.2207	19.0586
84	-98.2344	19.0373	194	-98.2215	19.0415	304	-98.2380	19.0365	414	-98.2345	19.0402	524	-98.2245	19.0346
85	-98.2324	19.0309	195	-98.2404	19.0629	305	-98.2303	19.0340	415	-98.2266	19.0417	525	-98.2099	19.0261
86	-98.2311	19.0378	196	-98.2248	19.0427	306	-98.2317	19.0372	416	-98.2389	19.0308	526	-98.2244	19.0263
87	-98.2458	19.0369	197	-98.2259	19.0436	307	-98.2240	19.0531	417	-98.2278	19.0464	527	-98.2263	19.0445
88	-98.2317	19.0671	198	-98.2237	19.0106	308	-98.2232	19.0351	418	-98.2230	19.0480	528	-98.2279	19.0427
89	-98.2409	19.0393	199	-98.2257	19.0505	309	-98.2348	19.0421	419	-98.2262	19.0215	529	-98.2265	19.0342
90	-98.2255	19.0513	200	-98.2236	19.0335	310	-98.2108	19.0425	420	-98.2187	19.0334	530	-98.2074	19.0419
91	-98.2331	19.0293	201	-98.2236	19.0204	311	-98.2335	19.0203	421	-98.2240	19.0345	531	-98.2265	19.0485
92	-98.2258	19.0362	202	-98.2443	18.9967	312	-98.2235	19.0444	422	-98.1897	19.0251	532	-98.2351	19.0455
93	-98.2307	19.0590	203	-98.2208	19.0588	313	-98.2222	19.0378	423	-98.2410	19.0349	533	-98.2428	19.0414
94	-98.2306	19.0401	204	-98.2251	19.0394	314	-98.2278	19.0469	424	-98.2208	19.0388	534	-98.2231	19.0001
95	-98.2300	19.0389	205	-98.2312	19.0519	315	-98.2383	19.0438	425	-98.2371	19.0294	535	-98.2260	19.0320
96	-98.2254	19.0349	206	-98.2238	19.0313	316	-98.2279	19.0554	426	-98.2214	19.0193	536	-98.2251	19.0450
97	-98.2372	19.0399	207	-98.2236	19.0225	317	-98.2268	19.0319	427	-98.2169	19.0525	537	-98.2218	19.0387
98	-98.2272	19.0379	208	-98.2175	19.0628	318	-98.2258	19.0216	428	-98.2173	19.0496	538	-98.1835	19.0701
99	-98.2241	19.0365	209	-98.2123	19.0352	319	-98.2261	19.0344	429	-98.2256	19.0348	539	-98.2298	19.0472
100	-98.2240	19.0591	210	-98.2359	19.0370	320	-98.2259	19.0376	430	-98.2231	19.0373	540	-98.2256	19.0249
101	-98.2266	19.0492	211	-98.2225	19.0448	321	-98.2277	19.0337	431	-98.2334	19.0232	541	-98.2221	19.0438
102	-98.2256	19.0406	212	-98.2281	19.0476	322	-98.2143	19.0259	432	-98.2322	19.0209	542	-98.2414	19.0405
103	-98.2176	19.0685	213	-98.2092	19.0350	323	-98.2441	19.0438	433	-98.2300	19.0601	543	-98.2241	19.0384
104	-98.2246	19.0346	214	-98.2285	19.0421	324	-98.2298	19.0248	434	-98.2327	19.0418	544	-98.2262	19.0627
105	-98.2172	19.0365	215	-98.2089	19.0359	325	-98.2316	19.0358	435	-98.2297	19.0481	545	-98.2248	19.0314
106	-98.2087	19.0403	216	-98.2142	19.0442	326	-98.2019	19.0338	436	-98.2257	19.0317	546	-98.2139	19.0627
107	-98.2078	19.0360	217	-98.2272	19.0258	327	-98.2248	19.0342	437	-98.2205	19.0240	547	-98.2277	19.0362
108	-98.2349	19.0432	218	-98.2250	19.0327	328	-98.2206	19.0234	438	-98.2174	19.0418	548	-98.2241	19.0271
109	-98.2289	19.0753	219	-98.2302	19.0413	329	-98.2284	19.0472	439	-98.2193	19.0440	549	-98.2380	19.0347
110	-98.2263	19.0199	220	-98.2351	19.0378	330	-98.2295	19.0344	440	-98.2090	19.0439	550	-98.2135	19.0363

Figure 24. Demand of units of product (pre-filled pens) and Location Data (latitude, longitude) for Example 1.

this case, the delivery points which consists of drugstores, small clinics, and patients at home. Nevertheless, ordering costs must cover the operating costs of the supplier which are dependent of the lot size and the service distance. For this case, C_h is estimated as 5.0% of C and the ordering cost is estimated as $C_o = d_s \times \frac{1}{d_e} \times d_j + d_p$, where d_s is the diesel cost per liter, d_e is

the efficiency of the vehicle (kilometers per liter), and d_j is the cumulative distance up to the delivery point j. For a standard vehicle d_e is estimated as 12 kilometers per liter and d_s as 1.10 USD per liter (based on regional costs). Because d_j is directly associated to the traveled distance, its minimization can be achieved through the application of an appropriate distribution planning model as the CVRP (*Programming Models 4(c), 7-10*).

Figure 25 presents the net requirements (lot size Q) per location based on the P model considering a period between reviews of two weeks. These results represent the demand data for the CVRP model. As minimization of the inventory management costs is the main objective of the distribution scheme, the total cost for the P model with the integrated C_o cost is considered as the objective function. Thus, Figure 25 also presents the distribution planning obtained by the CVRP and the associated C_o and total inventory costs for each delivery location.

The sum of all total costs associated to inventory management is estimated as 1016.5939 USD, where the total order cost is estimated as 470.5898 USD. This represents a significant investment even if appropriate inventory control and distribution planning is performed. A key aspect to improve these results is to extend on the availability of other distribution centers. This is to be addressed in the following example.

Example 2: Facility Location

From the previous analysis, the task of finding an alternative location for the distribution center was explored. As presented in Figure 26, the new location estimated at λ =-98.2266 and ϕ =19.0369 by the NNS approach (*Programming Models 11-13*), can lead to a total inventory management cost of approximately 905.1940 USD with a total order cost of 355.1493. This represents a reduction of (1- 905.1940/1016.5939) $\times$ 100.0% = 10.958% and (1- 355.1493/470.5898) $\times$ 100.0% = 24.531% respectively.

Figure 27 presents the results considering two distribution centers, with four and three vehicles respectively. This was achieved by changing the *capacity* variable in *Programming Models 11-13* from a single constant value (homogeneous capacity) to a vector with different capacity values for each distribution center (heterogeneous capacity). Under this scheme, the routes of the first distribution center can lead to a total inventory management cost of approximately 517.4791 USD with a total order cost of 180.7032 USD. On the other hand, the routes of the second distribution center can lead to a total inventory management cost of approximately 350.7018 USD with a total order cost of 137.4330 USD. In total, this strategy leads to a reduction of (1- (517.4791+350.7018)/905.1940) $\times$ 100.0% = 4.09% and (1- (180.7032+137.4330)/355.1493) $\times$ 100.0% = 10.42% respectively on the previous costs.

Figure 28 presents the total inventory management cost for the cases with three, four, five, six and seven distribution centers. It can be observed that there is a limit to increase the number of distribution centers to reduce the total inventory costs. A steady reduction is observed for up to five distribution centers. However, a steady increase is observed for six and seven distribution centers (which implies a vehicle per distribution center). Hence, it is important to consider that there is an optimal or near-optimal number of distribution centers to reduce the total costs associated to distance. Also, the achieved savings must compensate the investment required for this additional infrastructure within a suitable period of time.

RESULTS AND CONCLUSIONS

In this work the development of resources for logistic and inventory management research was presented. These resources are complemented with programming code and implementation examples to motivate their learning and application. Specifically, the following aspects are covered by this material:

- development of instances for vehicle – routing/ facility location instances with near-to-real geographical data;
- development of instances with symmetric and asymmetric distance/cost data considering the main distance metrics available for modeling;
- development of exporting and plotting codes for visualization and processing by third-party platforms;

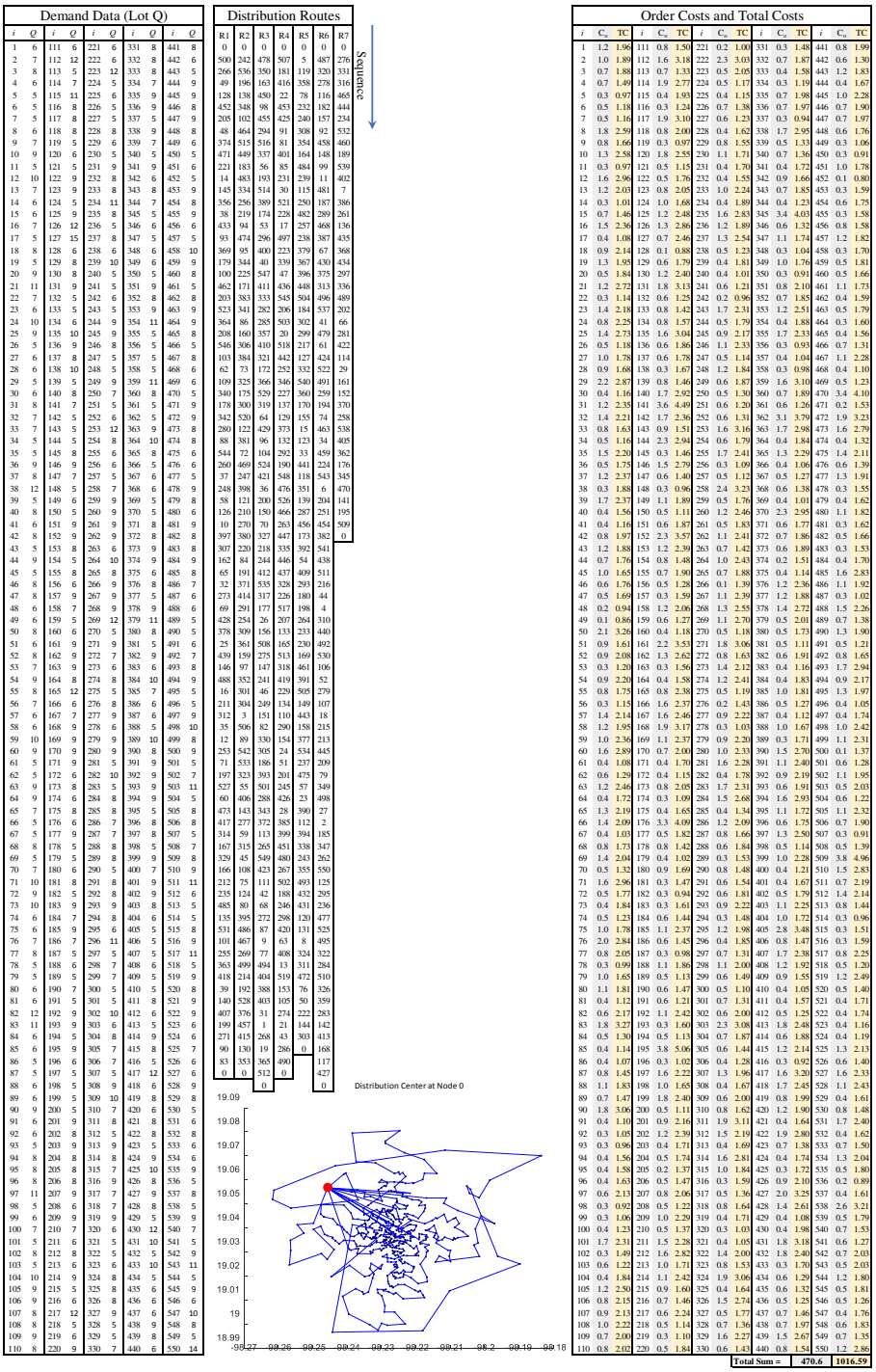


Figure 25. Results of the Inventory and Distribution Models for Example 1: Demand as Lot Quantities (Q) based on the Periodic Review *(P)* Model, C_o and *Total Costs*.

- development of implementation code to evaluate the performance of inventory management techniques with uncertain demand;

- development of a nearest-neighbor search with greedy randomized adaptive search procedure (GRASP) algorithm to (1) provide very suitable solutions to integrated facility location – inventory - vehicle routing problems, and (2) provide insights regarding how these solving

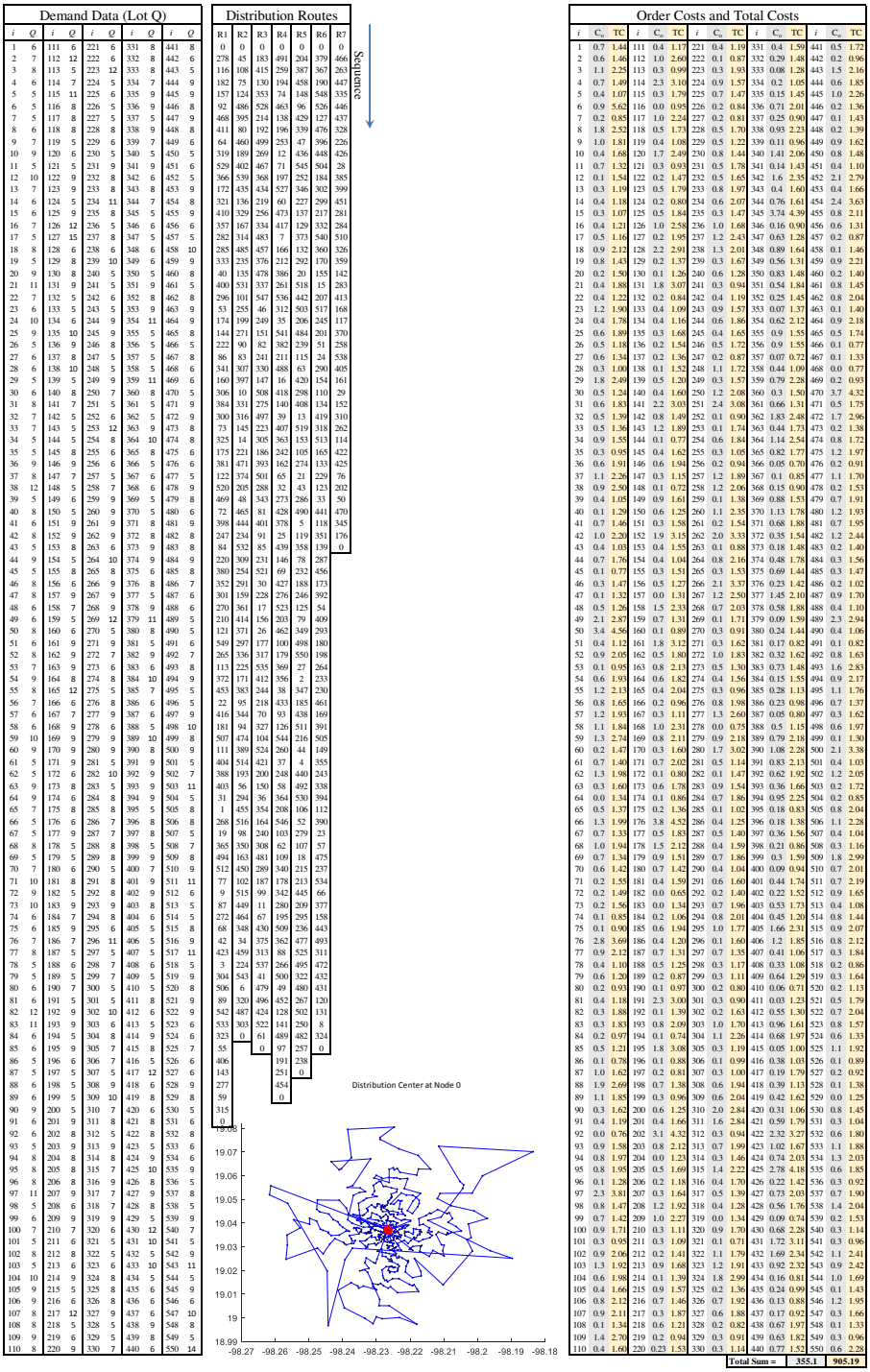
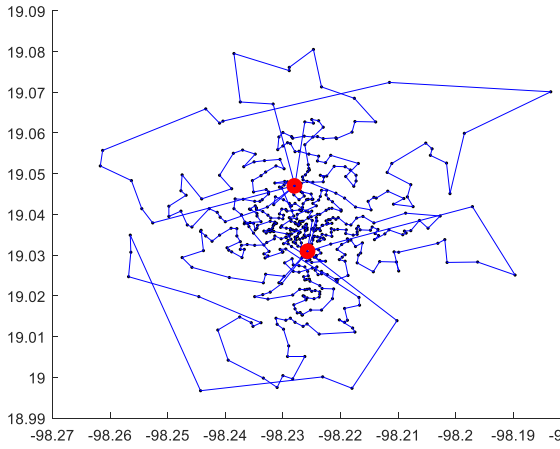


Figure 26. Results of the Inventory and Distribution Models for Example 2: Demand as Lot Quantities (Q) based on the Periodic Review *(P)* Model, C_o and *Total Costs*.

methods work;

- analysis of two examples considering these resources.

It is important to mention that logistic research extends to many fields and problems, and these resources just represent a small fraction of them. Also, the developed codes are subject to improvements. However, as introductory work, it can provide the academic or professional with



	Total Inventory Cost	Total Order Cost C_o
Distribution Center 1 at -98.22579 19.03084	517.4791	180.7032
Distribution Center 2 at -98.22793 19.04709	350.7018	137.4330
Total Costs (USD)	868.1809	318.1362

Figure 27. Results of Inventory Management Costs for Example 2: Strategy with Two Distribution Centers

the key aspects and tools to understand most of the associated works reported in the specialized literature.

STATEMENT

"The author declares that there is no conflict of interest regarding the publication of this paper."

REFERENCES

Adamu I (2017). Reorder quantities for (Q,R) inventory models. *International Mathematical Forum*, 12(11):505–514.

Aras N, Yumusak S, and Altmel K (2007). Solving the capacitated multi-facility weber problem by simulated annealing, threshold accepting and genetic algorithms. In Doerner, K, Gendreau M, Greistorfer P, Gutjahr W, Hartl R, and Reimann M, editors, *Metaheuristics: Progress in Complex Systems Optimization*, pages 491–112.

Basu S, Sharma M, and Ghosh P (2015). Metaheuristic applications on discrete facility location problems: a survey. *OPSEARCH*, 52(3):530—-561.

Braglia M, Castellano D, Marrazzini L, and Song D (2019). A continuous review (Q, r) inventory model for a deteriorating item with random demand and positive lead time. *Computers and Operations Research*, 109:102–121.

Bräysy O and Gendreau M (2005a). Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms. *Transportation Science*, 39(1):104—-118.

Bräysy O and Gendreau M (2005b). Vehicle Routing Problem with Time Windows, Part II: Metaheuristics. *Transportation Science*, 39(1):119—-139.

Chaves AA and Nogueira-Lorena LA (2010). Clustering search algorithm for the capacitated centered clustering problem. *Computers & Operations Research*, 37(3):552–558.

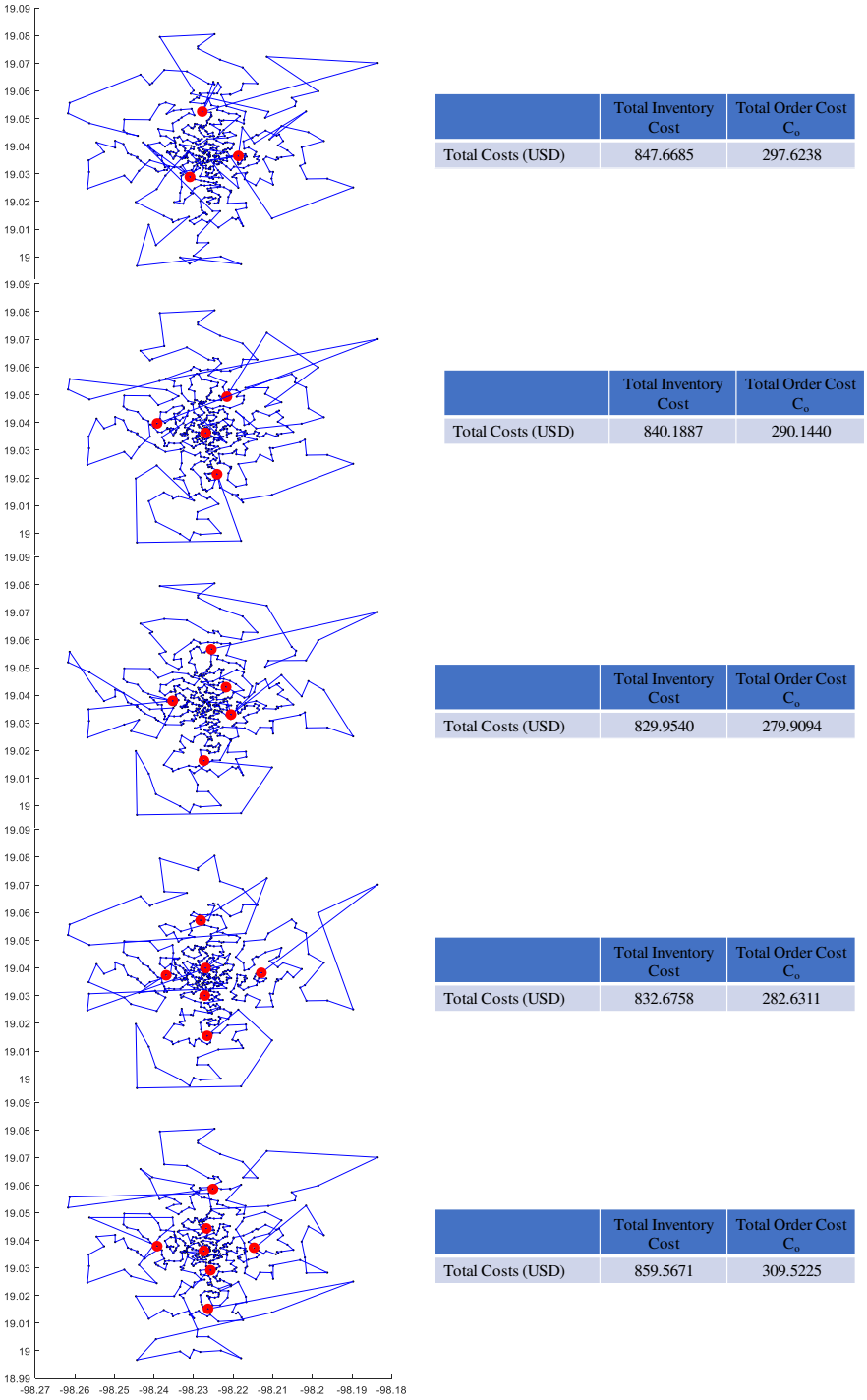
Chaves AA and Nogueira-Lorena LA (2011). Hybrid evolutionary algorithm for the capacitated centered clustering problem. *Expert Systems with Applications*, 38:5013—-5018.

Daugherty P, Bolumole Y, and Schwieterman M (2017). Logistics Research: What a Long, Strange Trip It's Been. *Transportation Journal*, 56(3):213–226.

De SK and Sana SS (2015). Multi-criterion multi-attribute decision-making for an EOQ model in a hesitant fuzzy environment. *Pacific Science Review A: Natural Science and Engineering*, 17:61–68.

Eaton JW, Bateman D, Hauberg S, and Wehbring R (2018). *GNU Octave version 4.4.0 manual: a high-level interactive language for numerical computations.*

Erturgut R (2011). Increasing demand for logistics technician in business world and rising trend of logistics programs in higher vocational schools: Turkey case. *Procedia - Social and Behavioral Sciences*, 15:2776–2780.



	Total Inventory Cost	Total Order Cost C_o
Total Costs (USD)	847.6685	297.6238

	Total Inventory Cost	Total Order Cost C_o
Total Costs (USD)	840.1887	290.1440

	Total Inventory Cost	Total Order Cost C_o
Total Costs (USD)	829.9540	279.9094

	Total Inventory Cost	Total Order Cost C_o
Total Costs (USD)	832.6758	282.6311

	Total Inventory Cost	Total Order Cost C_o
Total Costs (USD)	859.5671	309.5225

Figure 28. Results of Inventory Management Costs for Example 2: Strategy with Three to Seven Distribution Centers

Hariga M (2009). A continuous review (Q,r) model with owned and rented storage facilities. In *Proc. of the IEEE International Conference on Computers & Industrial Engineering (CIE 2009)*, pages 1297–1301.

Hovelaque V and Bironneau L (2015). The carbon-constrained EOQ model with carbon emission dependent demand. *International Journal of Production Economics*, 164:285–291.

Kuczyńska-Chalada M, Furman J, and Poloczek R (2018). The challenges for logistics in the

443 aspect of Iindustry 4.0. *Multidisciplinaty Aspects of Production Engineering*, 1(1):553–559.
444 Lieberman G and Hillier F (2000). *Introduction to Operations Research (7th ed.)*. McGraw-Hill.
445 Lloyd C, Pötsch T, Yi S, Zuñiga R, Safaei M, Issa S, and Rügge I (2013). Resources in logistics
446 - a multidisciplinary challenge. In *Proc. of the 6th International Federation of Automatic*
447 *Control Conference on Management and Control of Production and Logistics (IFAC 2013)*,
448 pages 449–455.
449 Negreiros M and Palhano A (2006). The capacitated centred clustering problem. *Computers &*
450 *Operations Research*, 33(6):1639—-1663.
451 Nogueira-Lorena LA (2007). *Problem Instances*. http://www.lac.inpe.br/l̃orena/instancias.html.
452 Oliveira D, Uchoa E, Pecin D, Pessoa A, Poggi M, Vidal T, and Subramanian A (2019). *CVRPLIB*
453 *Capacitated Vehicle Routing Problem Library*. http://vrp.atd-lab.inf.puc-rio.br/index.php/en/.
454 Prodhon C and Prins C (2014). A survey of recent research on location-routing problems.
455 *European Journal of Operational Research*, 238(1):1–17.
456 Rao K, Stenger A, and Wu HJ (1998). Integrating the use of computers in logistics education.
457 *International Journal of Physical Distribution & Logistics Management*, 28(4):302–319.
458 Reinelt G (1991). TSPLIB - A Traveling Salesman Problem Library. *ORSA Journal on*
459 *Computing*, 3(4):376–384.
460 Reinelt G (1997). *TSPLIB*. http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html.
461 Sachan A and Datta S (2005). Review of supply chain management and logistics research.
462 *International Journal of Physical Distribution & Logistics Management*, 35(9):664–705.
463 Stefanello F, CB-de-Araújo O, and Müller F (2015). Matheuristics for the capacitated p-median
464 problem. *International Transactions in Operational Research*, 22(1):149–167.
465 Thinakaran N, Jayaprakas J, and Elanchezhian C (2019). Survey on Inventory Model of EOQ &
466 EPQ with Partial Backorder Problems. *Materials Today: Proceedings*, 16:629–635.
467 Uchoa E, Pecin D, Pessoa A, Poggi M, Vidal T, and Subramanian A (2017). New benchmark
468 instances for the Capacitated Vehicle Routing Problem. *European Journal of Operational*
469 *Research*, 257(3):845–858.
470 Zäpfel G, Braune R, and Bögl M (2010). *Metaheuristic Search Concepts: A Tutorial with*
471 *Applications to Production and Logistics*. Springer Heidelberg Dordrecht London New York.