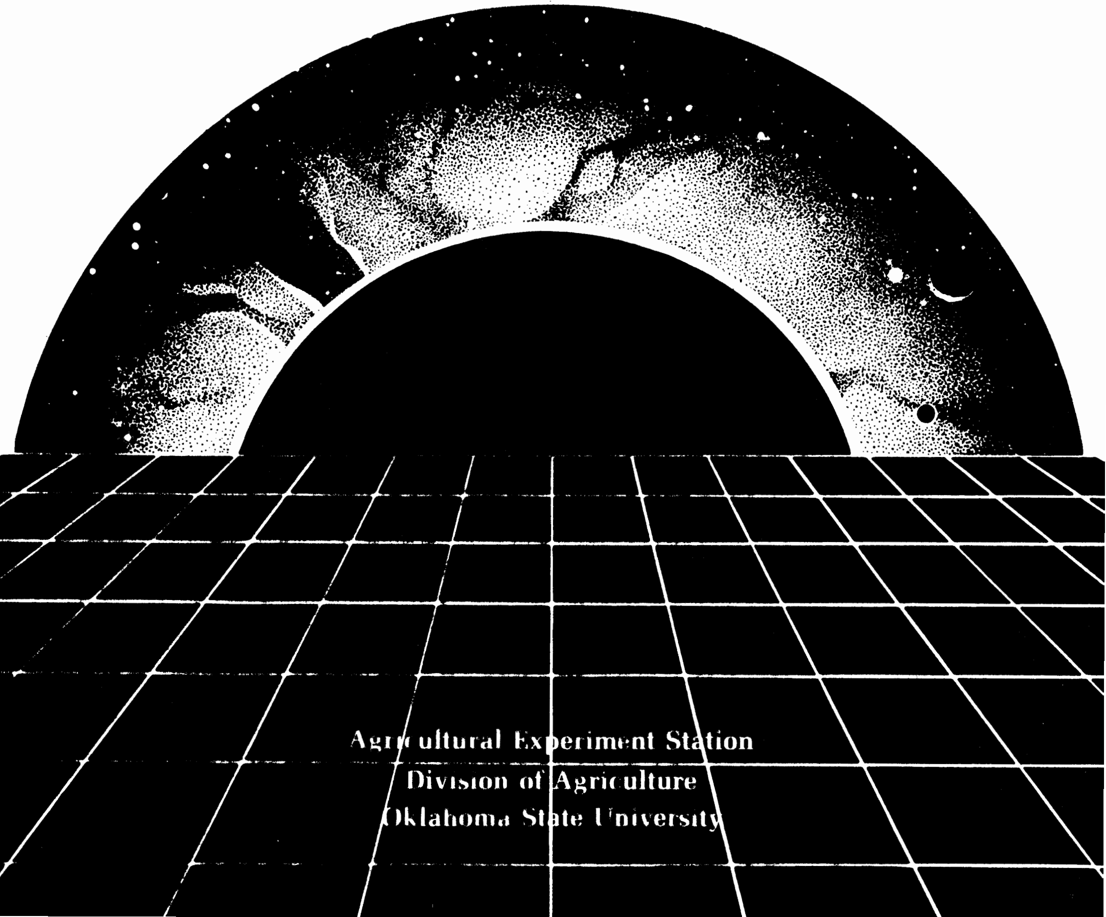
Optimal Plant Location on Planar and Spherical Surfaces

Technical Bulletin T-164

March 1988



Agricultural Experiment Station
Division of Agriculture
Oklahoma State University

OPTIMAL PLANT LOCATION ON PLANAR AND SPHERICAL SURFACES

by

David A. Pyles

This research was conducted under Oklahoma Agricultural Experiment Station Project H-1974.

OPTIMAL PLANT LOCATION ON PLANAR AND SPHERICAL SURFACES

by
David A. Pyles*

Introduction

A common problem in economic location theory is the determination of the best site for a single plant when outputs and inputs of the plant must be shipped to or from certain predetermined locations. The purpose of this paper is to develop some of the mathematics pertaining to such problems. The analysis will treat problems satisfying the following conditions: 1) the location decision is based entirely upon transportation costs, 2) the per-unit costs of shipping each input and each output are constant and known, 3) the quantity to be shipped to or from each of the predetermined locations is fixed and known, 4) every point on the geographical surface contained within the convex hull formed by the predetermined locations is a candidate for plant location, and 5) the relevant transportation distance from plant site to any of the predetermined locations is equal to the shortest geographical distance between the two points. In practical applications, one would seldom expect all of these assumptions to hold, but it is hoped that the assumptions describe a reasonable approximation to a significant class of location problems. Derivations are presented for optimal location on both planar and spherical surfaces, and an algorithm is suggested for solving both sorts of problems as well. Finally, an APPLESOFT BASIC program is provided for solving both spherical and planar location problems.

The Planar Location Problem

When shipments to and from the plant are to be transported short distances, one may often ignore the spherical shape of the earth, and imagine, with a small degree of error, that the shipment space is situated on a plane. Since mathematical evaluation of planes is generally easier than the evaluation of spheres, the assumption of a planar surface may be well justified if the degree of error resulting from such assumption is not excessive. For short distances on the

*Assistant Professor, Department of Agricultural Economics, Oklahoma State University, Stillwater. Research under Hatch 1974 of the Oklahoma Agricultural Experiment Station. Comments of James Trapp and Dave Park are appreciated.

earth's surface, the error of the planar approximation is indeed quite small. For example, if the spherical distance between two points on the surface of the earth is 1000 miles, then the planar distance is approximately 997.35 miles. The degree of error in this case is less than three-tenths of one percent.

Suppose that the plant must ship outputs and inputs to and from a total of n predetermined locations, which shall henceforth be referred to as "input-output points" or "I-O points." Let the rectangular coordinates of the ith I-O point be denoted as (a_i,b_i), and denote the coordinates of the plant as (x,y). Also, let w_i represent the per-mile costs of shipping total quantity to or from the ith I-O point. For example, suppose that the plant will ship input from the kth point and that it costs \$1.50 to ship one unit of input from (a_k,b_k) for one mile. If 500 units are to be shipped from this point, then $w_k = 750.00$. Weights for output points are calculated in exactly the same manner. Of course some points could be both sources of inputs as well as destinations of outputs.

If the objective is to situate the plant such that transportation costs are minimum, then the mathematical problem becomes one of minimizing a weighted sum of Euclidean distances. Specifically, the problem becomes:

$$\text{minimize}(x,y): \quad C(x,y) = \Sigma_{i=1}^{n} \; w_i D^i(x,y)$$

where C(x,y) is total transportation costs, and $D^i(x,y)$ is the Euclidean distance between (a_i,b_i) and (x,y). $D^i(x,y)$ is calculated as:

$$D^i(x,y) = [(x - a_i)^2 + (y - b_i)^2)]^{1/2}$$

It may be concluded a-priori that the optimal plant site must fall within the convex hull formed by the I-O points. The convex hull is the smallest convex set containing all of these points. Intuitively, the hull is formed by pulling a string around all of the I-O points and pulling it tight. The convex hull is then the space contained within the string. At any point outside the convex hull, it is possible to approach the hull so as to reduce the distance to every I-O point; consequently, the optimal plant site cannot occur outside the convex hull.

Assuming that the optimal plant site does not fall exactly on one of the I-O points, the optimizing values of x and y must set the first derivatives of C(x,y) equal to zero. Thus, necessary conditions are:

$$C_x = \Sigma \, w_i D_x^i(x,y) = 0$$

$$C_y = \Sigma \, w_i D_y^i(x,y) = 0$$

where:

$$D_x^i(x,y) = (x - a_i) / D^i(x,y)$$

$$D_y^i(x,y) = (y - b_i) / D^i(x,y)$$

Since C(x,y) is convex[1], a local minimum described by the above must also be a global minimum. Thus, the latter conditions are both necessary and sufficient for a global minimum provided that the optimum does not occur at an I-O point.

A minimizing solution to the above problem is not necessarily unique. Consider any even number of equally weighted I-O points that are situated on a line. In such cases, any point between the two central I-O points represents a minimizing solution. Hence, in this case, there are an infinite number of optima.

Since the partial derivatives of C(x,y) are undefined when (x,y) is equal to any of the (a_i,b_i), the above approach fails when the optimal plant site occurs exactly at an I-O point. However, it will now be shown that a necessary and sufficient condition for a global minimum to occur at (a_k,b_k) is:

$$w_k^2 \geq (\Sigma_{i\neq k}\, w_i a_i^k)^2 + (\Sigma_{i\neq k}\, w_i b_i^k)^2 \qquad (1')$$

where:

$$a_i^k = -D_x^i(a_k,b_k) = (a_i - a_k)/D^i(a_k,b_k)$$

$$b_i^k = -D_y^i(a_k,b_k) = (b_i - b_k)/D^i(a_k,b_k)$$

where $D^i(a_k,b_k)$ is the Euclidean distance between the ith I-O point and the kth I-O point. In a polar coordinate system having (a_k,b_k) as the origin, a_i^k is the cosine and b_i^k is the sine of the angle corresponding to the ith I-O point.

If a local minimum occurs at (a_k,b_k), then C is not reduced by small movements away from this point. Mathematically, this condition may be stated as:

$$dC = w_k D^k(a_k + dx, b_k + dy) + [\Sigma_{i\neq k}\, w_i D_x^i(a_k,b_k)]dx + [\Sigma_{i\neq k}\, w_i D_y^i(a_k,b_k)]dy \geq 0$$

Upon substituting the a_i^k, the b_i^k, and the explicit form of D^k into the above, one

[1]It may be confirmed that $D_{xx}^i \geq 0$, $D_{yy}^i \geq 0$, and that $D_{xx}^i D_{yy}^i - D_{xy}^i D_{yx}^i = 0$ for all x and y; thus, the D^i are convex, and consequently, C is convex.

obtains:

$$dC = w_k(dx^2 + dy^2)^{1/2} - (\Sigma_{i\neq k}\, w_i a_i^k)dx - (\Sigma_{i\neq k}\, w_i b_i^k)dy \geq 0$$

Now, terms can be rearranged to produce:

$$w_k(dx^2 + dy^2)^{1/2} \geq (\Sigma_{i\neq k}\, w_i a_i^k)dx + (\Sigma_{i\neq k}\, w_i b_i^k)dy$$

At this point, we convert to polar coordinates using the substitutions, $dx = r\cos(\theta)$ and $dy = r\sin(\theta)$. The latter expression then simplifies to :

$$w_k \geq (\Sigma_{i\neq k}\, w_i a_i^k)\cos(\theta) + (\Sigma_{i\neq k}\, w_i b_i^k)\sin(\theta)$$

If the relation holds where the right-hand side is maximized with respect to θ, then it is known to hold elsewhere. It may be confirmed that functions of the form, $c_1\cos(\theta) + c_2\sin(\theta)$, can be no greater than $(c_1^2 + c_2^2)^{1/2}$.[2] Hence, the latter relation is known to hold if:

$$w_k \geq [(\Sigma_{i\neq k}\, w_i a_i^k)^2 + (\Sigma_{i\neq k}\, w_i b_i^k)^2]^{1/2}$$

Squaring both sides of this relation will produce (1).

If (1) does hold for a particular I-O point, then it may be concluded that no other (x,y) can yield a lower value for C. This follows since (1) can hold only in the case of a local minimum, and since a local minimum must also be a global minimum for a convex function.

If the latter result holds, then it will not necessarily hold for the largest weight. This may be seen by considering a set of n I-O points and weights where the last point represents the optimal plant site when only the previous n - 1 points are considered. It should be apparent that the optimal plant site for the n points will be this last I-O point regardless of its weight. Thus, (1) could hold even for the smallest weight.

An interesting special case of (1) follows from the fact that $(a_i^k)^2 + (b_i^k)^2 = 1$. Using this result, it may be confirmed that the right-hand side of (1) is equal to:

$$\Sigma_{i\neq k}\, w_i^2 + \Sigma_{j\neq i}\, \Sigma_{i\neq k}\, (a_i^k a_j^k + b_i^k b_j^k) w_i w_j$$

---

[2]Since $\sin^2(\theta) + \cos^2(\theta) = 1$, this problem is equivalent to maximizing $c_1x_1 + c_2x_2$ subject to $x_1^2 + x_2^2 = 1$.

But, it can be shown that $a_i^k a_j^k + b_i^k b_j^k$ is at most equal to one,[3] subsequently:

$$\Sigma_{i \neq k} w_i^2 + \Sigma_{j \neq i} \Sigma_{i \neq k} (a_i^k a_j^k + b_i^k b_j^k) w_i w_j \leq \Sigma_{i \neq k} w_i^2 + \Sigma_{j \neq i} \Sigma_{i \neq k} w_i w_j$$

$$= (\Sigma_{i \neq k} w_i)^2$$

Thus, (1) is known to hold if:

$$w_k^2 \geq (\Sigma_{i \neq k} w_i)^2$$

or:

$$w_k \geq \Sigma_{i \neq k} w_i \tag{2}$$

This result is intuitively apparent since the most that could be gained by moving one mile away form (a_k,b_k) would be $\Sigma_{i \neq k} w_i$, but the loss would be at least as great as w_k.[4] Therefore, if (2) holds, then any movement away from (a_k,b_k) cannot reduce transportation costs.

The Spherical Location Problem

When the radius of the transportation region is large, a planar approximation to the surface of the earth might become unacceptably inaccurate. In these cases, the approach considered here should be taken.

The only difference between the planar and spherical location problems is the calculation of distance. Since our objective is still the minimization of total transportation costs, the objective function remains:

$$\text{minimize } (x,y): \quad C(x,y) = \Sigma_{i=1}^{n} w_i D^i(x,y)$$

However, the distance function now measures spherical rather than Euclidean distance. The explicit representation of D^i is:

$$D^i(x,y) = \cos^{-1}[\sin(b_i)\sin(y) + \cos(b_i)\cos(y)\cos(x - a_i)]$$

[3]This problem is equal to maximizing $x_1y_1 + x_2y_2$ subject to $x_2^2 + x_2^2 = 1$ and $y_2^2 + y_2^2 = 1$.

[4]This situation occurs when the I-O points are situated on a line with (a_k,b_k) being an endpoint.

Here, the (a_i,b_i) are the longitude-latitude coordinates of the I-O points, while (x,y) is the longitude-latitude coordinates of the plant. This formula assumes a sphere of unit radius. Distances on a sphere of radius r are given by rD^i. The formula also assumes that longitude coordinates for the eastern hemisphere and latitude coordinates for the southern hemisphere have been multiplied by -1. Also, note that $\cos^{-1}(.)$ is measured in radians, but the arguments of the other trigonometric function are measured in degrees. A derivation of the formula above is presented in Appendix A.

Assuming that the optimal plant site does not fall on one of the I-O points, the minimizing solution can be found by setting the first derivatives of C(x,y) equal to zero, or:

$$C_x = \Sigma\ w_i\cos(b_i)\cos(y)\sin(x - a_i)/(1 - u_i^2)^{1/2} = 0$$

$$Cy = -\Sigma\ w_i[\sin(b_i)\cos(y) - \cos(b_i)\sin(y)\cos(x - a_i)]/(1 - u_i^2)^{1/2} = 0$$

where:

$$u_i = \cos(D^i) = \sin(b_i)\sin(y) + \cos(b_i)\cos(y)\cos(x - a_i)$$

Unfortunately, C(x,y) is not convex in the spherical case so that there is no apparent guarantee that (x,y) satisfying the above will always locate a global minimum.

The latter approach fails when the minimum occurs at one of the I-O points since the partial derivatives of C(x,y) are undefined at such points. However, it will be shown that the necessary and sufficient condition for a local minimum to occur at (a_k,b_k) is:

$$w_k^2 \geq [\Sigma_{i\neq k}\ w_iD_x^i(a_k,b_k)]^2 / \cos^2(b_k) + [\Sigma_{i\neq k}\ w_iD_y^i(a_k,b_k)]^2 \qquad (3)$$

Again, there is no apparent reason that I-O points satisfying this condition should be globally minimal.

A local minimum occurs at (a_k,b_k) if and only if small departures from this point do not decrease C. Proceeding in similar fashion as in the planar case, this condition may be expressed as:

$$dC = w_kD^k(a_k + dx,b_k + dy) + [\Sigma_{i\neq k}\ w_iD_x^i(a_k,b_k)]dx + [\Sigma_{i\neq k}\ w_iD_y^i(a_k,b_k)]dy \geq 0$$

Now,convert the latter expression to polar coordinates with the substitutions, $dx = r\cos(\theta)$ and $dy = r\sin(\theta)$. Upon rearrangement of terms, the relation may then be written as:

$$w_k D^k[a_k + r\cos(\theta), b_k + r\sin(\theta)]/r \qquad (4)$$

$$\geq -[\Sigma_{i\neq k}\, w_i D_x^i(a_k,b_k)]\cos(\theta) - [\Sigma_{i\neq k}\, w_i D_y^i(a_k,b_k)]\sin(\theta)$$

where the inequality must hold for all θ with sufficiently small r. In particular, the expression must hold in the limit as r approaches zero. Now, observe that:

$$D^k[a_k + r\cos(\theta), b_k + r\sin(\theta)]/r = D^k(r)/r = \cos^{-1}[u_k(r)]/r$$

where:

$$u_k(r) = \sin(b_k)\sin[b_k + r\sin(\theta)] + \cos(b_k)\cos[b_k + r\sin(\theta)]\cos[r\cos(\theta)]$$

Using L'Hospital's rule:

$$L = \lim_{r\to 0} D^k(r)/r = [\lim_{r\to 0} - u_k'/(1 - u_k^2)^{1/2}]/(\lim_{r\to 0} 1)$$

A second application of l'Hospital's rule to the last result yields:

$$L = \lim_{r\to 0} - u_k'/(1 - u_k^2)^{1/2}$$

$$= (\lim_{r\to 0} - u_k''/[\lim_{r\to 0} - u_k' u_k/(1 - u_k^2)^{1/2}]$$

$$= [\lim_{r\to 0} - u_k'']/[\lim_{r\to 0} u_k]L$$

which implies:

$$L^2 = (\lim_{r\to 0} - u_k'')/(\lim_{r\to 0} u_k)$$

or:

$$L = \pm[(\lim_{r\to 0} - u_k'')/(\lim_{r\to 0} u_k)]^{1/2}$$

However, since $D^k(r)$ is always greater than or equal to zero, and since r approaches from the positive direction, the negative alternative for L may be disregarded. It may be confirmed that:

$$\lim_{r\to 0} u_k = 1$$

$$\lim_{r\to 0} -u''_k = \sin^2(\theta) + \cos^2(\theta)\cos^2(b_k)$$

Consequently:

$$L = [\sin^2(\theta) + \cos^2(\theta)\cos^2(b_k)]^{1/2}$$

Therefore, upon taking the limit of (4) as r approaches zero, one obtains:

$$w_k[\sin^2(\theta) + \cos^2(\theta)\cos^2(b_k)]^{1/2} \geq$$

$$- [\Sigma_{i\neq k}\, w_i D^i_x(a_k,b_k)]\cos(\theta) - [\Sigma_{i\neq k}\, w_i D^i_y(a_k,b_k)]\sin(\theta)$$

Actually, a stronger statement may be made; namely, the absolute value of the left-hand side is greater than or equal to the absolute value of the right-hand side. This follows since if we evaluate the relation at $\theta + \pi$ as opposed to θ, then the left hand side remains unchanged while the right-hand side reverses sign. This implies that both sides of the expression may be squared without invalidating the relation; subsequently:

$$w^2_k[\sin^2(\theta) + \cos^2(\theta)\cos^2(b_k)] \geq (\Sigma_{i\neq k}\, w_i D^i_x)^2\cos^2(\theta)$$

$$+ 2(\Sigma_{i\neq k}\, w_i D^i_x)(\Sigma_{i\neq k}\, w_i D^i_y)\sin(\theta)\cos(\theta) + (\Sigma_{i\neq k}\, w_i D^i_y)^2\sin^2(\theta)$$

Both sides of this expression can be divided by $\cos^2(\theta)$, and terms can be rearranged to produce:

$$[w^2_k - (\Sigma_{i\neq k}\, D^i_y)^2]\tan^2(\theta) - 2(\Sigma_{i\neq k}\, w_i D^i_x)(\Sigma_{i\neq k}\, w_i D^i_y)\tan(\theta)$$

$$+ [w^2_k\cos^2(b_k) - (\Sigma_{i\neq k}\, w_i D^i_x)^2] \geq 0$$

Since $\tan(\theta)$ ranges from negative to positive infinity, the latter relation can hold for all θ only if the discriminant of the left-hand side is less than or equal to zero. It may be confirmed that a nonpositive discriminant requires:

$$w^2_k \geq [\Sigma_{i\neq k}\, w_i D^i_x\,(a_k,b_k)]^2/\cos^2(b_k) + [\Sigma_{i\neq k}\, w_i D^i_y(a_k,b_k)]^2$$

which is the same with (3).

With much algebra, it can be shown that:

$$[D^i_x(a_k,b_k)/\cos(b_k)]^2 + [D^i_y(a_k,b_k)]^2 = 1$$

Therefore, by the same reasoning used in the derivation of (2), it can be shown that a special case of (3) occurs when:

$$w_k \geq \Sigma_{i \neq k}\, w_i$$

An Algorithm for Solving Location Problems

The Solution Procedure

The APPLESOFT BASIC program in Appendix B may be used to solve either planar or spherical location problems. For both sorts of problems, the program utilizes the steepest descent method to minimize the transportation cost function. The steepest descent method proves to be a rather efficient algorithm for solving location problems. Here, the steepest descent algorithm is briefly outlined.

To minimize the transportation cost function, C(x,y), the steepest descent algorithm begins at some initial point, (x_0,y_0), and then proceeds iteratively toward the minimum, moving at each iteration in the direction of the steepest instantaneous descent. The direction of steepest instantaneous descent at the ith iteration is given by the negative of the gradient vector evaluated at the current values of x and y. Thus, if (x_{i-1},y_{i-1}) denotes the value of (x,y) going into the ith iteration, then the cost function descends at the greatest instantaneous rate in the direction of the vector:

$$d_i = -\begin{bmatrix} c_x(x_{i-1},y_{i-1}) \\ c_y(x_{i-1},y_{i-1}) \end{bmatrix}$$

x and y are adjusted at each iteration according to the rule:

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} x_{i-1} \\ y_{i-1} \end{bmatrix} + \lambda_i d_i$$

Here, d_i determines the direction of movement, while λ_i determines the step length or extent of adjustment.

Various rules can be used for calculating λ_i. The rule used in the BASIC program attempts to choose λ_i which minimizes C in the direction of d_i. The algorithm sets $\lambda_0 = 1$ and then determines the λ_i for subsequent iterations using the following procedure: First, λ_i is set to $2\lambda_{i-1}$. This value is then repeatedly divided by two until such divisions produce λ_i rendering a lower value of C than that obtained in the previous iteration. This value is then repeatedly divided by two as long as such divisions produce an even lower value of C. In several experiments, this procedure generally produced very rapid convergence.

A primary advantage of the steepest descent algorithm is that it will find stationary minima as well as minima occurring at I-O points. Also, the algorithm inherently tends to avoid saddle points (it "falls out of the saddle"). However, in the spherical optimization problem, it is possible that the algorithm will converge at a local but not global minimum. Consequently, in spherical problems, the algorithm should be executed several times, using different starting values in each trial.

Various stopping rules can be used in the steepest descent algorithm. In the case of the present problem, two sorts of optima must be considered, and therefore, two different stopping criteria must be used. In the BASIC program, it is assumed that a minimum is attained either if the norm of the gradient vector is less than some specified distance from zero, or if the algorithm produces two consecutive (x,y) that are less than some specified distance form an I-O point. The first stopping criterion covers the possibility of an optimum occurring at a stationary point, while the second criterion covers the possibility of an optimum occurring at an I-O point.

Instructions for the BASIC Program

The interactive prompts issued by the program should be largely self explanatory; however, there are a few points needing special attention. These points are discussed here.

The first two inputs to the program are "TOLORENCE DIST. FOR GRADIENT," and "TOLERENCE DIST. FROM I-O PT.". These tolerance levels are used to determine the stopping points in the minimization algorithm. The program will assume that it has attained a minimum if: 1) the norm of the gradient vector is less than the "TOLERENCE DIST. FOR GRADIENT," or 2) for two successive iterations the algorithm produces (x,y) that are less than the "TOLERENCE DIST. FROM I-O PT.". Tolerance levels of .001 will probably be sufficient for both of these inputs.

Next, the program will ask for the "# OF I-O POINTS," and then provide the option of either accessing the data from disk or from the keyboard. If the data are to be taken from disk, but the user is not sure of the number of I-O points that are recorded in the disk file, then for "# OF I-O POINTS," type any number that is known to be less than the number of I-O points in the file. If the user wishes to add more I-O points to the set contained in the file, then type the number of points that there will be after the additions are made. Later, the additions can be entered in the correction routine.

If the data are to be retrieved from disk, then the program will allow the option to output the data to the printer in an echo print. However, this option is not provided when the data are entered at the keyboard. For echo prints in the latter case, the data should be punched in, saved to disk, and then reloaded.

When data are to be keyed in, the program will ask for the coordinates and weight for each I-O point. The user should type the first coordinate, second coordinate, and then the weight, separating each by commas. Press return to elicit the prompt for the next I-O point. In the case of spherical location problems, the coordinates should be measured in terms of degrees and not in terms of degrees, minutes, and seconds. Longitude coordinates for the eastern hemisphere and latitude coordinates for the southern hemisphere should be entered as negative numbers.

After the data are retrieved either from disk or keyboard, the program will provide the option to check and correct the data. The correction routine will list the I-O points and weights on the screen in groups of 15. It will then ask if corrections are to be made to the presented data. To make a correction, type in the location number (a location number is assigned by the program to each I-O point), the first coordinate, the second coordinate, and then the weight, separating each by commas. Press return to see the revised data and to elicit the prompt for the next correction. To delete an I-O point, change its weight to zero in this routine. Also, if the "# OF I-O PTS." supplied earlier is greater than the number of points found in the disk file, then the additional I-O points will have coordinates and weights being equal to zero. These should be supplied with the correct values here.

After corrections are made, the program will provide the option to save the data on disk. If this option is taken, then the data will be stored in a DOS 3.3 sequential file. The order of the data in the file will be: 1) the number of I-O points, 2) the first coordinate, second coordinate, and weight of the first I-O point, 3) the first coordinate, second coordinate, and weight of the second I-O point, and so on.

The program will then allow the option to supply starting values for the plant coordinates. If this option is refused, then the program will use a weighted average of the I-O coordinates for starting values. As mentioned before, spherical problems should probably be ran several times, using difference starting values in each trial. Planar problems must be executed only once, and the option to supply starting coordinates should probably be refused.

The program will print summary statistics for each iteration. These include the current values of x and y, the current norm of the gradient vector (presented under "D"), and the current value of the cost function (presented under "OBJ."). When execution is completed, the program will present the final solution as well as the distance of the plant from each I-O point and the total costs of shipping to and/or from the point.

References

Beightler, Charles S., Don T. Phillips, Douglas J. Wilde. Foundations of Optimization, 2nd ed. Englewood Cliffs, N.J.: Prentice-Hall, 1979.

Losch, August. The Economics of Location. Yale University Press, 1967.

Luenberger, David G. Introduction to Linear and Nonlinear Programming. Reading Massachusetts: Addison-Wesley Publishing Company, 1973.

Rushton, Gerald, Michael F. Goodchild, Lawrence M. Ostresh Jr. Computer Programs for Location-Allocation Problems, Monograph No. 6. Iowa City, Iowa: Department of Geography, University of Iowa, 1973.

Shapiro, Max S. Mathematics Encyclopedia. Garden City, New York: Doubleday and Company, 1977.

Weber, Alfred. Theory of the Location of Industries. Chicago, Illinois: The University of Chicago Press, 1969.

Appendix A

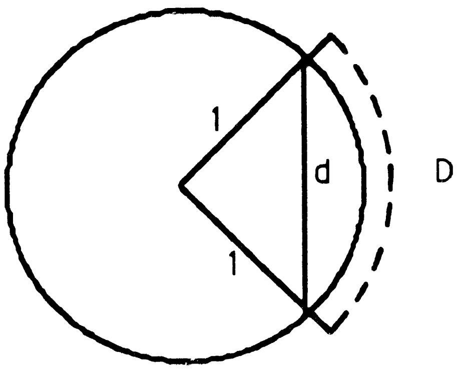
Spherical Distance

If (x_1, y_1) and (x_2, y_2) are longitude-latitude coordinates for two locations on a unit sphere, then the spherical distance between the two locations is given by the formula:

$$D = \cos^{-1}[\sin(y_1)\sin(y_2) + \cos(y_1)\cos(y_2)\cos(x_1 - x_2)]$$

where the inverse consine function is measured in terms of radians, while all other trigonometric functions are for measurements in terms of degrees. In this appendix, we validate the above formula.

If the planar or straight line distance between two points on the sphere is known, then the spherical distance can be calculated using the law of cosines. For example, consider the sphere below:



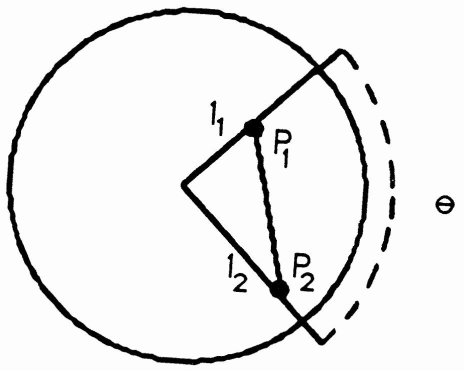
Here, the straight-line distance between the two points on the sphere is denoted by d, whereas the spherical distance is denoted by D. By the law of cosines, it is known that:

$$D = \cos^{-1}(1 - d^2/2) \tag{1}$$

To determine d, drop an imaginary right angle into the sphere with (x_1,y_1) and (x_2,y_2) being connected by the hypotenuse. The verticle leg of the right angle should be perpendicular to the equatorial plane. Obviously, the planar distance between the two points is equal to the distance of the hypotenuse. If the length of the verticle and horizontal legs can be determined, then the length of the hypotenuse can be calculated using Pythagorean's theorem. The length of the verticle leg, v, can be measured as the absolute value of the difference between the distances of the two points from the equatorial plane: hence:

$$v = \left|\sin(y_1) - \sin(y_2)\right|$$

This formula assumes that latitude coordinates as reported for the southern hemisphere have been multiplied by -1. The length of the horizontal leg, h, is the horizontal distance between the two points. To determine this distance, drop plumb lines from (x_1,y_1) and (x_2,y_2) and mark the points where these lines intersect the equatorial plane. If these marks are identified as p_1 and p_2, then the equatorial plane should appear:



where:

$$l_1 = \cos(y_1)$$

$$l_2 = \cos(y_2)$$

$$\theta = \left|x_1 - x_2\right|$$

These equations assume that longitude coordinates in the eastern hemisphere have been multiplied by -1. By the law of cosines:

$$h^2 = l_1^2 + l_2^2 - 2l_1l_2\cos(\theta) = \cos^2(y_1) + \cos^2(y_2) - 2\cos(y_1)\cos(y_2)\cos(x_1 - x_2)$$

Using Pythagorean's theorem, the planar distance between the two points can be calculated as:

$$d^2 = v^2 + h^2 = [\sin^2(y_1) + \sin^2(y_2) - 2\sin(y_1)\sin(y_2)] + [\cos^2(y_1) + \cos^2(y_2) - 2\cos(y_1)\cos(y_2)\cos(x_1 - x_2)]$$

Since $\sin^2(\theta) + \cos^2(\theta) = 1$, the latter becomes:

$$d^2 = 2 - 2[\sin(y_1)\sin(y_2) + \cos(y_1)\cos(y_2)\cos(x_1 - x_2)]$$

The substitution of this relation into (1) yields the spherical distance formula:

$$D = \cos^{-1}[\sin(y_1)\sin(y_2) + \cos(y_1)\cos(y_2)\cos(x_1 - x_2)]$$

For a sphere of radius r, the distance between (x_1,y_1) and (x_2,y_2) would be given by rD.

Appendix B

APPLESOFT BASIC Program for Solving Location Problems

```
10  DATA  0,1,2,3,4,5,6,7,8,9,".","00","-"
20 READ N0,N1,N2,N3,N4,N5,N6,N7,N8,N9,P$,ZP$,D$
30 DIM  A(100),B(100),W(100),SB(100),CB(100)
40 CD$ = CHR$(N4)
50 DR = .0174532
60 HP = 1.5707963
70 RE = 3963.34
80 DEF FN DTR(X) = X*DR
90 DEF FN RTD(X) = X/DR
100 DEF FN ACS(X) = HP - ATN(X/SQR(N1 - X^N2))
110 HOME
120 PRINT "ARE COORDINATES SPHERICAL ? (Y/N): ";
130 GET C1$
140 IF C1$ <> "Y" AND C1$ <> "N" THEN 130
150 PRINT
160 PRINT
170 IF C1$ = "Y" THEN SC = N1
180 INPUT "TOLERENCE DIST. FOR GRADIENT: ? ";T1
190 IF T1 <= N0 THEN 180
200 PRINT
210 INPUT "TOLERENCE DIST. FROM I-O PT.: ? ";T2
220 IF T2 <= N0 THEN 210
230 PRINT
240 INPUT "# OF I-O PTS.: ? ";N
250 IF N <= N0 THEN 240
260 N = INT(N)
270 PRINT
280 PRINT "ARE DATA ON DISK ? (Y/N): ";
290 GET C1$
300 IF C1$ <> "N" AND C1$ <> "Y" THEN 290
310 PRINT
320 PRINT
330 IF C1$ = "N" THEN 670
340 INPUT "FILE NAME: ? ";C1$
350 PRINT
360 PRINT "OUTPUT DATA TO PRINTER ? (Y/N): ";
```

```
370 GET C2$
380 IF C2$ <> "Y" AND C2$ <> "N" THEN 370
390 IF C2$ = "Y" THEN PP = N1
400 PRINT
410 PRINT
420 PRINT "INSERT DISK IN DRIVE 1 AND PRESS RETURN";
430 GET C2$
440 PRINT
450 HOME
460 IF PP THEN PRINT CD$"PR#1"
470 PRINT "LOCATION";SPC(N9);"A";SPC(N9);"B";SPC(N9);"W"
480 GOSUB 3510
490 PRINT CD$"OPEN";C1$,",D1"
500 PRINT CD$"READ";C1$
510 INPUT R1
520 IF R1 > N THEN N = R1
530 FOR I = N1 TO R1
540 INPUT A(I),B(I),W(I)
550 C1$ = STR$(I)
560 C2$ = STR$(A(I))
570 C3$ = STR$(B(I))
580 C4$ = STR$(W(I))
590 PRINT SPC(N8 - LEN(C1$));C1$;SPC(10 - LEN(C2$));C2$;SPC(10 -
    LEN(C3$));C3$;SPC(10 - LEN(C4$));C4$
600 NEXT
610 PRINT CD$"CLOSE"
620 PRINT
630 PRINT
640 PP = N0
650 PRINT CD$"PR#0"
660 GOTO 860
670 HOME
680 PRINT "LOCATION";SPC(N9);"A";SPC(N9);"B";SPC(N9);"W"
690 GOSUB 3510
700 VTAB 23
710 PRINT "INPUT COORDINATES AND WEIGHT FOR EACH LOCATION"
720 PRINT
730 FOR I = N1 TO N
740 C1$ = STR$(I)
750 POKE 34,23
760 PRINT SPC(N8 - LEN(C1$));C1$;"  ";
770 INPUT "(A,B,W) = ";A(I),B(I),W(I)
```

```
780 PRINT
790 C2$ = STR$(A(I))
800 C3$ = STR$(B(I))
810 C4$ = STR$(W(I))
820 PRINT SPC(N8 - LEN(C1$));C1$;SPC(10 - LEN(C2$);C2$;SPC(10 -
    LEN(C3$));C3$;SPC(10 - LEN(C4$));C4$;
830 POKE 34,N2
850 NEXT
860 PRINT
870 PRINT "CHECK DATA ? (Y/N): ";
880 GET C1$
890 IF C1$ <> "Y" AND C1$ <> "N" THEN 880
900 PRINT
910 PRINT
920 IF C1$ = "N" THEN 1240
930 FOR I = N1 TO N STEP 15
940 R1 = I + 14
950 IF R1 > N THEN R1 = N
960 HOME
970 FOR J = I TO R1
980 C1$ = STR$(J)
990 C2$ = STR$(A(J))
1000 C3$ = STR$(B(J))
1010 C4$ = STR$(W(J))
1020 PRINT SPC(N8 - LEN(C1$));C1$;SPC(10 - LEN(C2$));C2$;SPC(10 -
     LEN(C3$));C3$;SPC(10 - LEN(C4$));C4$
1030 NEXT
1040 PRINT
1050 PRINT "CORRECTIONS ? (Y/N): ";
1060 GET C1$
1070 IF C1$ <> "N" AND C1$ <> "Y" THEN 1060
1080 PRINT
1090 PRINT
1100 IF C1$ = "N" THEN 1140
1110 INPUT "(LOCATION,A,B,W) = ";L;A(L),B(L),W(L)
1120 HOME
1130 GOTO 970
1140 NEXT
1150 R1 = N0
1160 FOR I = N1 TO N
1170 IF W(I) = N0 THEN 1220
1180 R1 = R1 + N1
```

```
1190 A(R1) = A(I)
1200 B(R1) = B(I)
1210 W(R1) = W(I)
1220 NEXT
1230 N = R1
1240 TEXT
1250 HOME
1260 PRINT "STORE DATA ON DISK ? (Y/N): ";
1270 GET C1$
1280 IF C1$ <> "Y" AND C1$ <> "N" THEN 1270
1290 PRINT
1300 PRINT
1310 IF C1$ = "N" THEN 1470
1320 INPUT "FILE NAME: ? ";C1$
1330 PRINT
1340 PRINT "INSERT DISK IN DRIVE 1 AND PRESS RETURN";
1350 GET C2$
1360 PRINT
1370 PRINT
1380 PRINT CD$"OPEN";C1$;",D1"
1390 PRINT CD$"WRITE";C1$
1400 PRINT N
1410 FOR I = N1 TO N
1420 PRINT A(I)
1430 PRINT B(I)
1440 PRINT W(I)
1450 NEXT
1460 PRINT CD$"CLOSE"
1470 IF NOT SC THEN 1520
1480 FOR I = N1 TO N
1490 A(I) = FN DTR(A(I))
1500 B(I) = FN DTR(B(I))
1510 NEXT
1520 PRINT "SUGGEST STARTING COORDINATES ? (Y/N): ";
1530 GET C1$
1540 IF C1$ <> "N" AND C1$ <> "Y" THEN 1530
1550 PRINT
1560 PRINT
1570 IF C1$ = "N" THEN 1630
1580 INPUT "(X,Y) = ";X,Y
1590 PRINT
1600 IF SC THEN X = FN DTR(X)
```

```
1610 IF SC THEN Y = FN DTR(Y)
1620 GOTO 1710
1630 R1 = N0
1640 FOR I = N1 TO N
1650 X = X + W(I)*A(I)
1660 Y = Y + W(I)*B(I)
1670 R1 = R1 + W(I)
1680 NEXT
1690 X = X/R1
1700 Y = Y/R1
1710 PRINT "SEND OUTPUT TO PRINTER ? (Y/N): ";
1720 GET C1$
1730 IF C1$ <> "Y" AND C1$ <> "N" THEN 1720
1740 PRINT
1750 IF C1$ = "Y" THEN PP = N1
1760 HOME
1770 IF PP THEN PRINT CD$"PR#1"
1780 PRINT "IT.";SPC(N7);"X";SPC(N7);"Y";SPC(N7);"D";SPC(N8);      "OBJ."
1790 GOSUB 3510
1800 IF SC THEN 2180
1810 F = N0
1820 FX = N0
1830 FY = N0
1840 P2 = N0
1850 FOR I = N1 TO N
1860 R1 = SQR((X - A(I))^N2 + (Y - B(I))^N2)
1870 IF R1 < T2 THEN P2 = I
1880 F = F + W(I)*R1
1890 FX = FX + W(I)*(X - A(I))/R1
1900 FY = FY + W(I)*(Y - B(I))/R1
1910 NEXT
1920 D = SQR(FX^N2 + FY^N2)
1930 GOSUB 3210
1940 IF D < T1 THEN 2610
1950 IF P2 <> P1 OR P1 = N0 THEN 1990
1960 X = A(P2)
1970 Y = B(P2)
1980 GOTO 2610
1990 P1 = P2
2000 IF IT = N0 THEN T = F/(N2*(FX^N2 + FY^N2))
2010 T = T*N2
2020 R1 = F
```

```
2030 R2 = X - T*FX
2040 R3 = Y - T*FY
2050 R4 = N0
2060 FOR I = N1 TO N
2070 R4 = R4 + W(I)*SQR((R2 - A(I))^N2 + (R3 - B(I))^N2)
2080 NEXT
2090 IF R4 < F AND R4 > R1 THEN 2130
2100 R1 = R4
2110 T = T/N2
2120 GOTO 2030
2130 T = T*N2
2140 X = X - T*FX
2150 Y = Y - T*FY
2160 IT = IT + N1
2170 GOTO 1810
2180 FOR I = N1 TO N
2190 SB(I) = SIN(B(I))
2200 CB(I) = COS(B(I))
2210 NEXT
2220 F = N0
2230 FX = N0
2240 FY = N0
2250 P2 = N0
2260 FOR I = N1 TO N
2270 R1 = SB(I)*SIN(Y) + CB(I)*COS(Y)*COS(X - A(I))
2280 R2 = FN ACS(R1)
2290 IF RE*R2 < T2 THEN P2 = I
2300 F = F + W(I)*R2
2310 R2 = -W(I)/SQR(N1 - R1^N2)
2320 FX = FX - R2*CB(I)*COS(Y)*SIN(X - A(I))
2330 FY = FY + R2*(SB(I)*COS(Y) - CB(I)*SIN(Y)*COS(X - A(I)))
2340 NEXT
2350 D = SQR(FX^N2 + FY^N2)
2360 GOSUB 3210
2370 IF D < T1 THEN 2610
2380 IF P2 <> P1 OR P1 = N0 THEN 2420
2390 X = A(P2)
2400 Y = B(P2)
2410 GOTO 2610
2420 P1 = P2
2430 IF IT = N0 THEN T = F/(N2*(FX^N2 + FY^N2))
2440 T = T*N2
```

```
2450 R1 = F
2460 R2 = X - T*FX
2470 R3 = Y - T*FY
2480 R4 = N0
2490 FOR I = N1 TO N
2500 R4 = R4 + W(I)*FN ACS(SB(I)*SIN(R3) + CB(I)*COS(R3)*COS(R2 -
      A(I)))
2510 NEXT
2520 IF R4 < F AND R4 > R1 THEN 2560
2530 R1 = R4
2540 T = T/N2
2550 GOTO 2460
2560 T = T*N2
2570 X = X - T*FX
2580 Y = Y - T*FY
2590 IT = IT + N1
2600 GOTO 2220
2610 R1 = X
2620 R2 = Y
2630 IF NOT SC THEN 2670
2640 R1 = FN RTD(R1)
2650 R2 = FN RTD(R2)
2660 F = RE*F
2670 PRINT
2680 PRINT
2690 PRINT "OPTIMAL SOLUTION"
2700 PRINT "----------------"
2710 PRINT
2720 PRINT "X: ";R1
2730 PRINT "Y: ";R2
2740 PRINT "D: ";D
2750 PRINT "OBJ.: ";F
2760 PRINT
2770 PRINT
2780 PRINT CD$"PR#0"
2790 PRINT "PRESS RETURN FOR MORE";
2800 GET C1$
2810 PRINT
2820 TEXT
2830 HOME
2840 IF PP THEN PRINT CD$"PR#1"
```

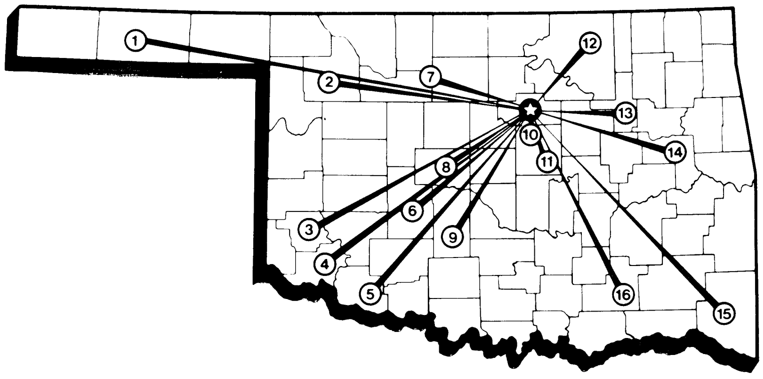
```
2850 PRINT "LOCATION";SPC(N5);"DIST.";SPC(N4);"WEIGHT";SPC(N6);
"COST"
2860 GOSUB 3510
2870 R3 = N0
2880 R4 = N0
2890 FOR I = N1 TO N
2900 IF NOT SC THEN R1 = SQR((X - A(I))^N2 + (Y - B(I))^N2)
2910 IF SC THEN R1 = RE*FN ACS(SB(I)*SIN(Y) + CB(I)*COS(Y)*COS(X     -
     A(I)))
2920 R2 = R1*W(I)
2930 R3 = R3 + R1
2940 R4 = R4 + R2
2950 C1$ = STR$(I)
2960 R = R1
2970 GOSUB 3390
2980 C2$ = C$
2990 R = W(I)
3000 GOSUB 3390
3010 C3$ = C$
3020 R = R2
3030 GOSUB 3390
3040 C4$ = C$
3050 PRINT SPC(N8 - LEN(C1$));C1$;SPC(10 - LEN(C2$));C2$;SPC(10 -
     LEN(C3$));C3$;SPC(10 - LEN(C4$));C4$
3060 NEXT
3070 FOR I = N1 TO 40
3080 PRINT D$;
3090 NEXT
3100 PRINT
3110 R = R3
3120 GOSUB 3390
3130 C1$ = C$
3140 R = R4
3150 GOSUB 3390
3160 C2$ = C$
3170 PRINT SPC(18 - LEN(C1$));C1$;SPC(20 - LEN(C2$));C2$
3180 PRINT CD$"PR#0"
3190 TEXT
3200 END
3210 C1$ = STR$(IT)
3220 R = X
3230 IF SC THEN R = FN RTD(R)
```

```
3240 GOSUB 3390
3250 C2$ = C$
3260 R = Y
3270 IF SC THEN R = FN RTD(R)
3280 GOSUB 3390
3290 C3$ = C$
3300 R = D
3310 GOSUB 3390
3320 C4$ = C$
3330 R = F
3340 IF SC THEN R = RE*R
3350 GOSUB 3390
3360 C5$ = C$
3370 PRINT SPC(N3 - LEN(C1$));C1$;SPC(N8 - LEN(C2$));C2$;SPC(N8 -
     LEN(C3$));C3$;SPC(N8 - LEN(C4$));C4$;SPC(12 - LEN(C5$));C5$
3380 RETURN
3390 IF ABS(R) < .01 THEN R = N0
3400 C$ = STR$(R)
3410 S1 = LEN(C1$)
3420 S2 = N1
3430 IF S2 > S1 THEN 3470
3440 IF MID$(C$,S2,N1) = P$ THEN 3480
3450 S2 = S2 + N1
3460 GOTO 3430
3470 C$ = C$ + P$
3480 C$ = C$ + ZP$
3490 C$ = LEFT$(C$,S2 + N2)
3500 RETURN
3510 FOR I = N1 TO 40
3520 PRINT D$;
3530 NEXT
3540 PRINT
3550 POKE 34,N2
3560 RETURN
```

OKLAHOMA

AGRICULTURAL EXPERIMENT STATION

System Covers the State



✪ Main Station – *Stillwater and Lake Carl Blackwell*

1. Panhandle Research Station – *Goodwell*
2. Southern Great Plains Field Station – *Woodward*
3. Sandyland Research Station – *Mangum*
4. Irrigation Research Station – *Altus*
5. Southwest Agronomy Research Station – *Tipton*
6. Caddo Research Station – *Ft. Cobb*
7. North Central Research Station – *Lahoma*
8. Forage and Livestock Research Laboratory – *El Reno*
9. South Central Research Station – *Chickasha*
10. Agronomy Research Station – *Perkins*
    Fruit Research Station – *Perkins*
11. Pecan Research Station – *Sparks*
12. Pawhuska Research Station – *Pawhuska*
13. Vegetable Research Station – *Bixby*
14. Eastern Research Station – *Haskell*
15. Kiamichi Forestry Research Station – *Idabel*
16. Wes Watkins Agricultural Research and Extension Center – *Lane*

OKLAHOMA STATE UNIVERSITY
CENTENNIAL
1890 • 1990

Reports of Oklahoma Agricultural Station serve people of all ages, race, color, national origin, religion, sex, or handicap. This publication is printed and issued by Oklahoma State University as authorized by the Dean of the Division of Agriculture and has been prepared and distributed at a cost of $434.62 for 366 copies. AI-7427 0388 TD