# Vector Calculus Project

Ethan Agnew, Phi Nguyen, and Will Silloway

## Problem

We were presented with the problem of finding the ideal location for a base of operations for a team of technicians tasked with maintaining multiple banks of computers. Each bank has a different amount of computers, and thus a different amount of times they would have to travel to them to be fixed. It is required to go to Middlebury three times, Tampa four times, to Dallas two times, and Detroit once. Our goal was to find the place that yields the minimum cost for travel over all other locations in the US.

Note: we rounded to two decimal places for most of the calculations in this project and kept longitude positive for simplicity of calculation.

## Creating the Equation

We wanted to find an equation that yields a distance equation (cost included) with weights correlated to how often there is travel to the given destination. Our cost is \$6.74 per mile, and the leading scalar of each respective distance equation is the number of times there is travel to the location, and the following two accounts for the round trip. The 50 and 70 inside the parentheses account for each degree of longitude and latitude in miles.

$$\text{Midd Distance} = 3 * 2 * 6.74\sqrt{(50(x_0 - 73.17))^2 + (70(y_0 - 44.02))^2}$$

$$\text{Tampa Distance} = 4 * 2 * 6.74\sqrt{(50(x_0 - 82.46))^2 + (70(y_0 - 27.95))^2}$$

$$\text{Dallas Distance} = 2 * 2 * 6.74\sqrt{(50(x_0 - 96.80))^2 + (70(y_0 - 32.78))^2}$$

$$\text{Detroit Distance} = 1 * 2 * 6.74\sqrt{(50(x_0 - 83.05))^2 + (70(y_0 - 42.33))^2}$$

$$C(x, y) = 13.48[3\sqrt{(50(x_0 - 73.17))^2 + (70(y_0 - 44.02))^2} + 4\sqrt{(50(x_0 - 82.46))^2 + (70(y_0 - 27.95))^2} + 2\sqrt{(50(x_0 - 96.80))^2 + (70(y_0 - 32.78))^2} + \sqrt{(50(x_0 - 83.05))^2 + (70(y_0 - 42.33))^2}]$$

This is our resulting cost equation.

## Optimizing the Equation

With our cost equation established, the next objective is to determine the specific parameters that yield the function's absolute minimum value. This optimization process involves a few critical steps: first, we will derive the gradient of our cost function. Next, we will identify the stationary points by solving for the coordinates where this gradient is equal to zero, which we have MATLAB do for us. Finally, by evaluating the original function at these coordinates, we can quantify the minimum cost and confirm the optimal state for our model.

```
% Define cost as function of vector v = [x, y]
C = @(v) 13.48 * ( ...
```

```
    3*sqrt((50*(v(1)-73.17))^2 + (70*(v(2)-44.01))^2) + ...
    2*sqrt((50*(v(1)-96.80))^2 + (70*(v(2)-32.78))^2) + ...
    4*sqrt((50*(v(1)-82.46))^2 + (70*(v(2)-27.95))^2) + ...
    1*sqrt((50*(v(1)-83.05))^2 + (70*(v(2)-42.33))^2));

v0 = [82.6, 35.12]; % Initial guess for [x, y]
[v_opt, min_cost] = fminsearch(C, v0); % this line searches for a minimum
with the initial guess

fprintf('Optimal x: %.4f\n', v_opt(1));
```

```
Optimal x: 83.0540
```

```
fprintf('Optimal y: %.4f\n', v_opt(2));
```

```
Optimal y: 31.1198
```

```
fprintf('Minimum Cost: %.4f\n', min_cost);
```

```
Minimum Cost: 83045.5437
```

```

% We want to prove that our convexity argument holds

% Create symbolic variables and expression
syms x y real
Csym = 13.48*( ...
    3*sqrt((50*(x-73.17))^2 + (70*(y-44.01))^2) + ...
    2*sqrt((50*(x-96.80))^2 + (70*(y-32.78))^2) + ...
    4*sqrt((50*(x-82.46))^2 + (70*(y-27.95))^2) + ...
    1*sqrt((50*(x-83.05))^2 + (70*(y-42.33))^2) );

% Gradient (Cx, Cy)
dCdx = diff(Csym, x);
dCdy = diff(Csym, y);

% Convert to numeric function handles for faster evaluation
dCdx_fun = matlabFunction(dCdx, 'Vars', [x y]);
dCdy_fun = matlabFunction(dCdy, 'Vars', [x y]);

% Solve the nonlinear system dCdx = 0, dCdy = 0 using vpasolve (provide
initial guesses)
% Try one initial guess
x0 = 82.6; y0 = 35.12;
sol = vpasolve([dCdx == 0, dCdy == 0],[x y],[x0 y0]);

% Display numeric solution(s)
if ~isempty(sol)
    solx = double(sol.x);
    soly = double(sol.y);
    fprintf('Symbolic root: x = %.6f, y = %.6f\n', solx, soly);
    fprintf('C at root = %.6f\n', double(subs(Csym, {x,y}, {solx, soly})));
```

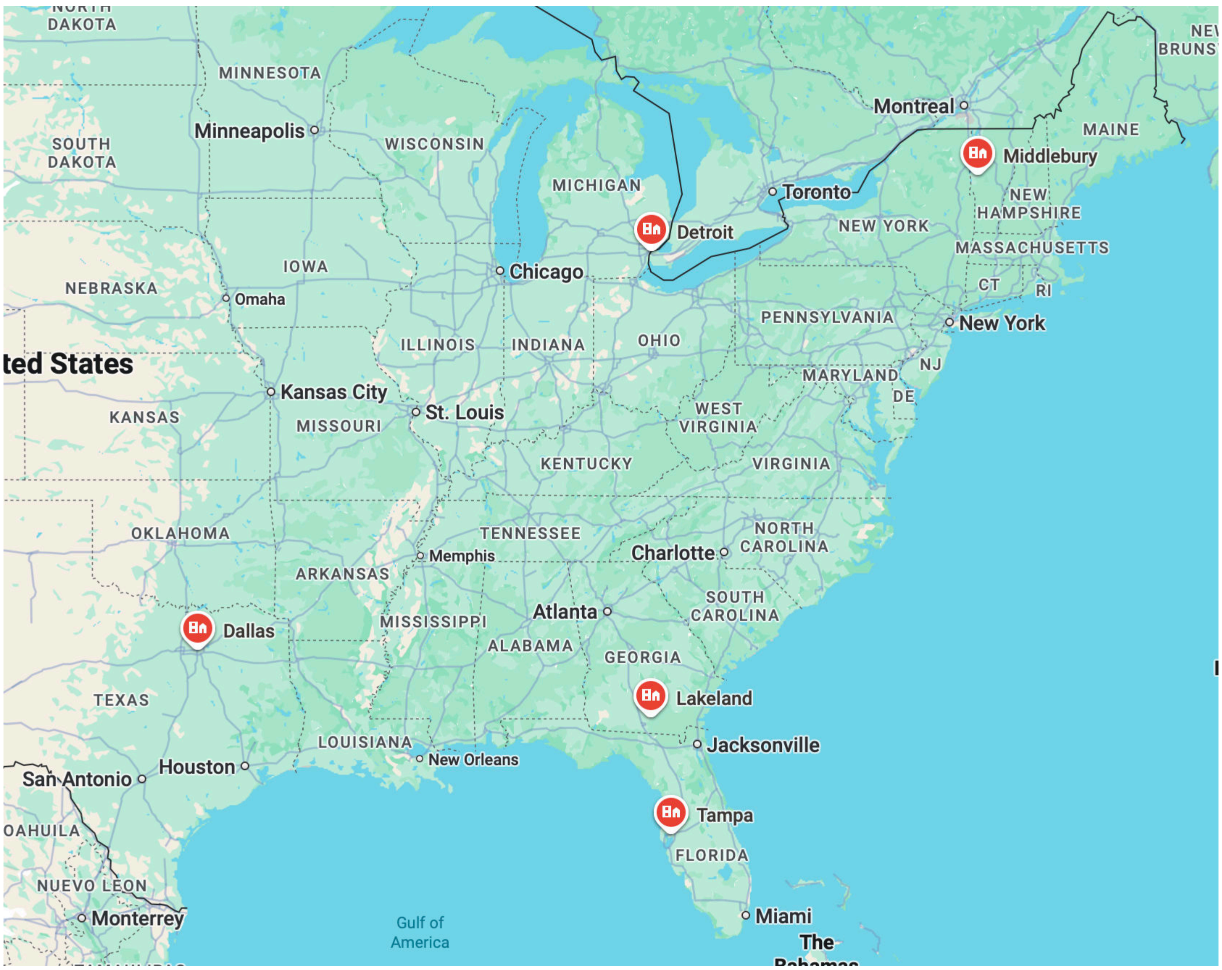
```
else
    disp('No solution found by vpasolve with that initial guess. Try different initial guesses or ranges.');
end
```

```
Symbolic root: x = 83.053962, y = 31.119776
C at root = 83045.543667
```

As one can see, our resulting minimum yields a longitude of 83.05 and a latitude of 31.12 leaving us with our minimum cost of $83,045.54.

## Where is this?

This longitude and latitude lands us near Lakeland, Georgia. This makes sense because we travel to Tampa the highest number of times, which skews our location south, and Middlebury is the second highest occurrence of visits, which skews our location east. Dallas and Detroit bear less weight due to a smaller number of visits. Without even doing the math, one could likely piece together that the optimal location would likely be in the southeast. Using code to further support this, we were able to recreate what Google Maps displays in terms of longitude and latitude.



## What would the costs be if we were located in each original location?

To do this, we used MATLAB to simulate the cost respective to each location.

```
% Compute costs at each location (pass the 2-element vector to C)
C_Middlebury = C(Middlebury);
C_Dallas     = C(Dallas);
C_Tampa      = C(Tampa);
C_Detroit    = C(Detroit);

% Display results
fprintf('Cost at Middlebury: %.2f\n', C_Middlebury);
```

```
Cost at Middlebury: 110691.96
```

```
fprintf('Cost at Dallas: %.2f\n', C_Dallas);
```

```
Cost at Dallas: 113058.83
```

```
fprintf('Cost at Tampa: %.2f\n', C_Tampa);
```

```
Cost at Tampa: 84136.95
```

```
fprintf('Cost at Detroit: %.2f\n', C_Detroit);
```

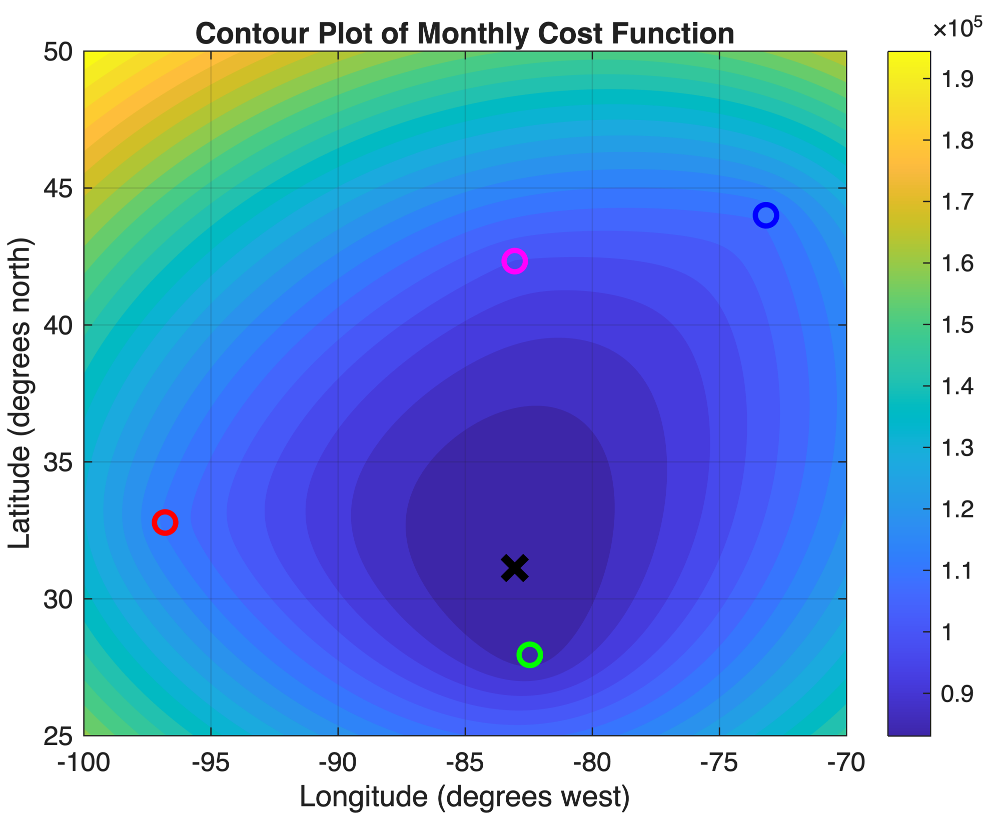
```
Cost at Detroit: 100687.58
```

## Graphing

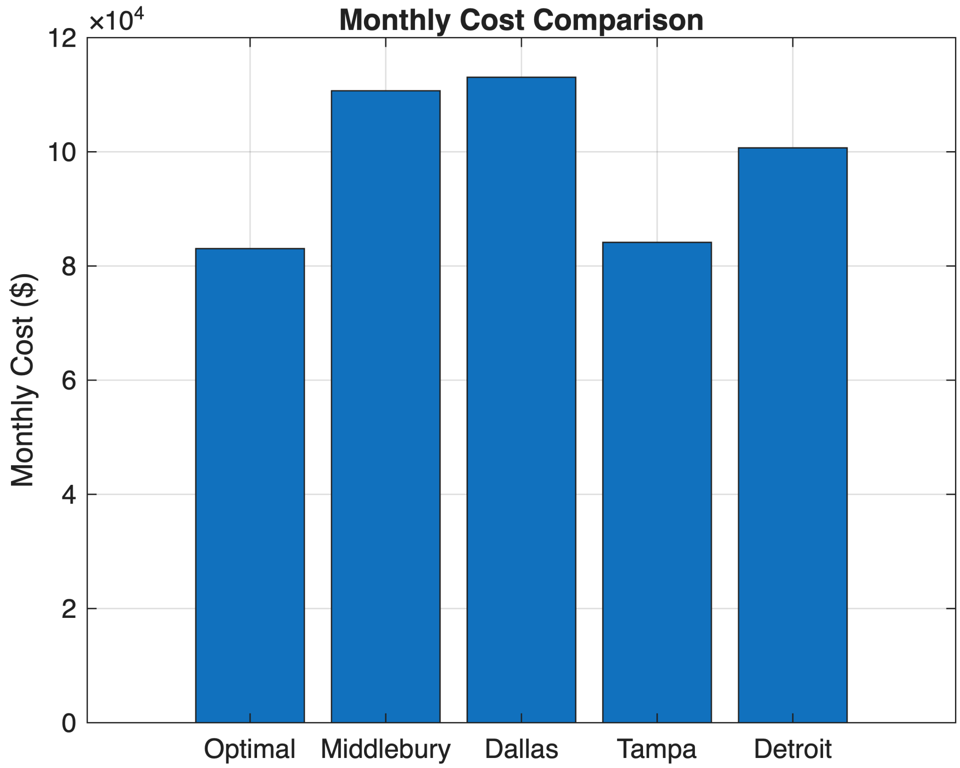
We wanted to see our results and our function in a more visual way. Our first visual was a cost contour plot, which shows level sets of our cost function C(x,y) over x in [70,100] and y in [25,50] — each contour line connects points with equal cost. Darker/lower-color regions are lower cost; lighter/higher-color regions are higher cost. It visualizes how the weighted sum of scaled distances to the four city locations varies across the plane and where the minimum lies. Our second plot, a surface plot, shows how the cost value changes in 3D and where the minimum-cost location lies. In short, we made cool graphs to show how the cost changes as the longitude and latitude change. We also provided a bar chart to compare monthly costs for each location.

```
% Grid for x and y
xv = linspace(70, 100, 201);
yv = linspace(25, 50, 201);
[X, Y] = meshgrid(xv, yv);

% Evaluate C on the grid (vectorize by building 2-by-N inputs)
V = [X(:)'; Y(:)'];              % 2 x N
Z = zeros(1, size(V,2));
for k = 1:size(V,2)
    Z(k) = C(V(:,k)');
end
Z = reshape(Z, size(X));

% Contour plot
```

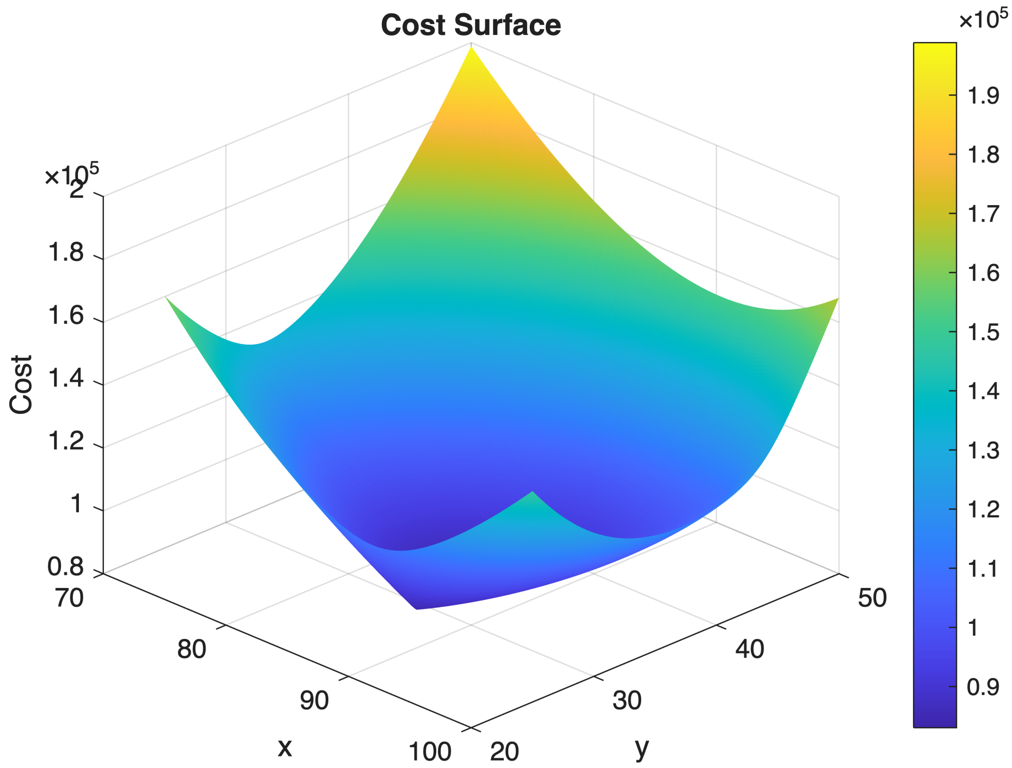
```
% ==========================================================
xv = linspace(-100, -70, 201);
yv = linspace(25, 50, 201);
[LON, LAT] = meshgrid(xv, yv);

% Rebuild Z on the negated longitude grid
V = [LON(:)'; LAT(:)'];
Z = zeros(1, size(V,2));
for k = 1:size(V,2)
    Z(k) = C([-V(1,k), V(2,k)]);  % negate lon back for cost function
end
Z = reshape(Z, size(LON));

figure;
contourf(LON, LAT, Z, 25, 'LineColor', 'none');
colorbar; hold on;
plot(-Middlebury(1), Middlebury(2), 'bo', 'MarkerSize', 8, 'LineWidth', 2);
plot(-Dallas(1),     Dallas(2),     'ro', 'MarkerSize', 8, 'LineWidth', 2);
plot(-Tampa(1),      Tampa(2),      'go', 'MarkerSize', 8, 'LineWidth', 2);
plot(-Detroit(1),    Detroit(2),    'mo', 'MarkerSize', 8, 'LineWidth', 2);
plot(-v_opt(1),      v_opt(2),      'kx', 'MarkerSize', 12, 'LineWidth', 3);
xlabel('Longitude (degrees west)');
ylabel('Latitude (degrees north)');
title('Contour Plot of Monthly Cost Function');
grid on;
```



```
% Bar chart comparison
% ==========================================================
figure;
vals   = [min_cost, C_Middlebury, C_Dallas, C_Tampa, C_Detroit];
labels = {'Optimal','Middlebury','Dallas','Tampa','Detroit'};
bar(vals);
set(gca, 'XTickLabel', labels);
ylabel('Monthly Cost ($)');
title('Monthly Cost Comparison');
grid on;
```




```
% Surface plot
figure;
surf(X, Y, Z, 'EdgeColor', 'none');
view(45, 30);
xlabel('x'); ylabel('y'); zlabel('Cost');
title('Cost Surface');
colorbar;
```




## Conclusion

By modeling the United States as a flat coordinate plane and weighting each destination by both travel frequency and round-trip cost, we built a cost function that measures the total monthly cost of operating the technicians' travel schedule. We then used MATLAB to minimize the given function numerically and found the critical point where the gradient is zero. This gives us the ideal base location, approximately at longitude 83.05 degrees West and latitude 31.12 degrees North, which places the office in southern Georgia. The minimum monthly travel cost at this location is about $83,045.54.

We also compared this minimum cost to the cost of placing the home base directly at each of the four cities. In the 4 cases we see, the optimized location still produced the lowest monthly cost, meaning that none of the existing service cities is the most efficient location. Since our cost function is a positive weighted sum of the distance functions, it is convex, so the critical point we found is a true minimum rather than a maximum or saddle point. Overall, this project demonstrates how vector calculus can be applied in a practical optimization setting to inform real-world business decisions. Based on our model, we recommend placing the technicians' office in Southern Georgia (Lakeland), since this location minimizes the total monthly travel cost while balancing the travel demands of all four computer sites.

## Honor Code

I have neither given nor recieved unauthorized aide on this assignment.