# Optimize the Flight Path of a Pan-US Delivery Drone

## Introduction

You run a pan-US drone delivery service for a popular online retailer. You're given a list of zip codes across the US at which you need to drop off parcels, and want to optimize its journey so it travels the shortest distance.

This application extracts the latitude and longitude of those zip codes from an SQLite database (the database is included with the application, and cross-references US zip codes against their latitude, longitude, city and state). It then performs a traveling salesman optimization and plots the shortest path on a map of the US.

This application uses background plot images, and SQLite integration, two new features introduced in Maple 18.

```
> with(Database[SQLite]) : with(plots) : with(plottools) : with(GraphTheory) :
```

This is the list of US zip codes

```
> zipCodeList := [22035, 94043, 49464, 90210, 72913, 80306, 45874, 59015, 73344, 67543] :
```

## Query the Database for the Latitude and Longitude of the Zip Codes

```
> connection := Open("zipcodes") :
> query := cat("SELECT longitude,latitude FROM USZipCodes WHERE zip=", StringTools:-Join(convert
      ~(zipCodeList, string), " OR zip=")) :
> stmt := Prepare(connection, query) :
> lonlat := parse~(FetchAll(stmt)) :
```

Functions that convert longitude and latitude to Cartesian coordinates

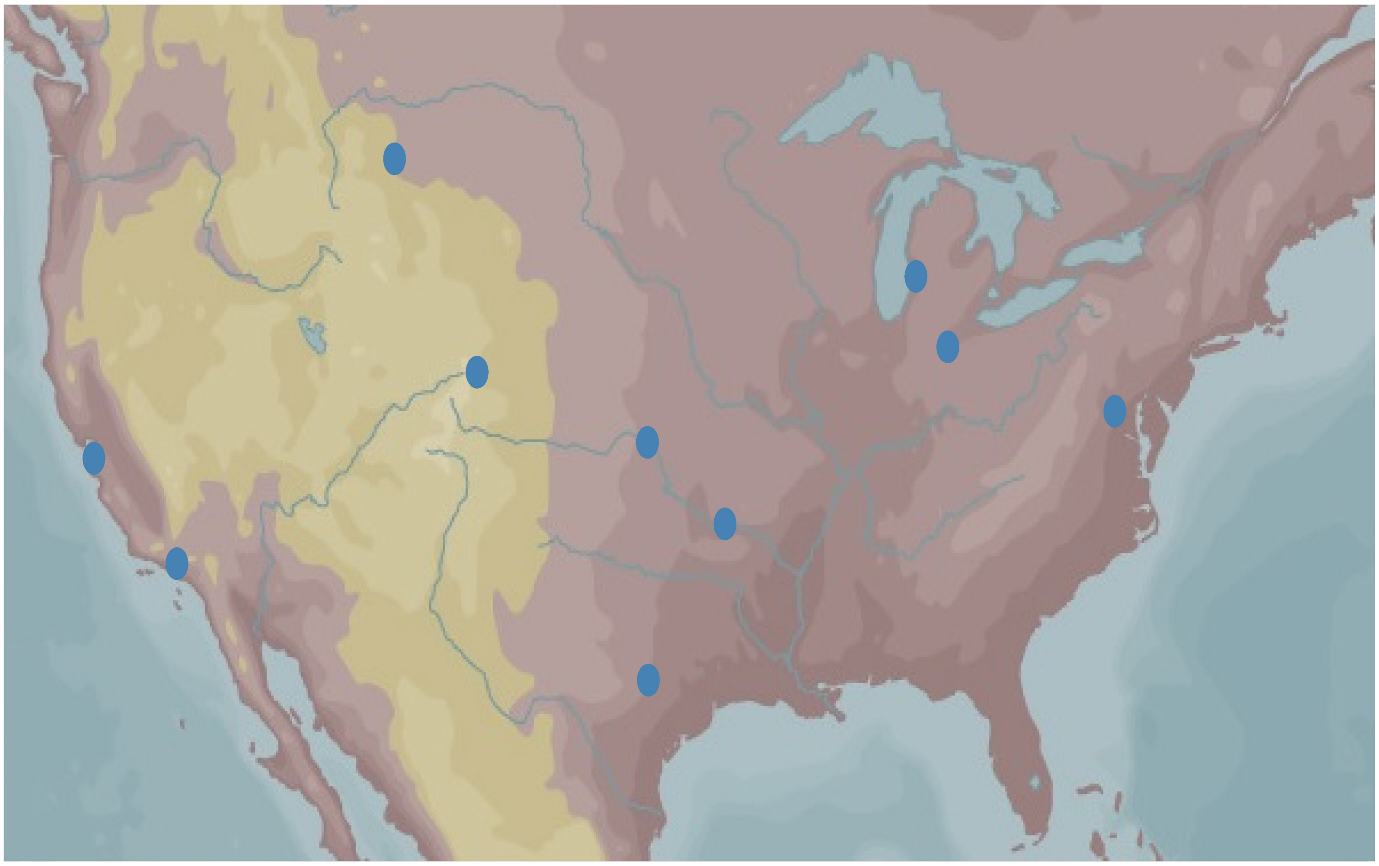
> $x := lon \to lon$ :

$$y := lat \to \mathrm{evalf}\left(45 \cdot \ln\left(\tan\left(\frac{Pi}{4} + \frac{lat \cdot Pi}{2 \cdot 180}\right)\right)\right) :$$

> locMag := seq(disk([x($lonlat_{i,1}$), y($lonlat_{i,2}$)], 0.5, color = "SteelBlue", style = patchnogrid), i = 1 ..op(lonlat)$_1$) :

## Display Location of Zip Code on Map

```
> display(locMag, background = "USMap.jpg", size = [660, 331], view = [-126..-66, y(24)..y(50)],
      axes = none)
```



## ▼ Traveling Salesman Optimization

```
> HaversineDist := proc(lat1, lon1, lat2, lon2)
      local R, dLat, dLon, a, c;
      R := 6373;
      dLat := lat2 − lat1;
      dLon := lon2 − lon1;
      $a := \sin\left(\frac{dLat}{2}\frac{\pi}{180}\right)^2 + \cos\left(lat1\frac{\pi}{180}\right)\cos\left(lat2\frac{\pi}{180}\right)\sin\left(\frac{dLon}{2}\frac{\pi}{180}\right)^2;$
      $c := 2\arctan\left(\sqrt{a}, \sqrt{1-a}\right);$
      return evalf(R c);
  end proc:
```

Create an undirected graph representing the zip code locations

```
> n := LinearAlgebra[RowDimension](lonlat) :
  G := CompleteGraph(n) :
```

Create a matrix of edge weights from the distance between locations

```
> M := Matrix(n, n) :
  for i to n do
    for j to n do
      $M_{i,j}$ := HaversineDist($lonlat_{i,2}$, $lonlat_{i,1}$, $lonlat_{j,2}$, $lonlat_{j,1}$)
    end do
  end do
```

Find the Hamiltonian Cycle

> w, tour := TravelingSalesman( G, M ) :

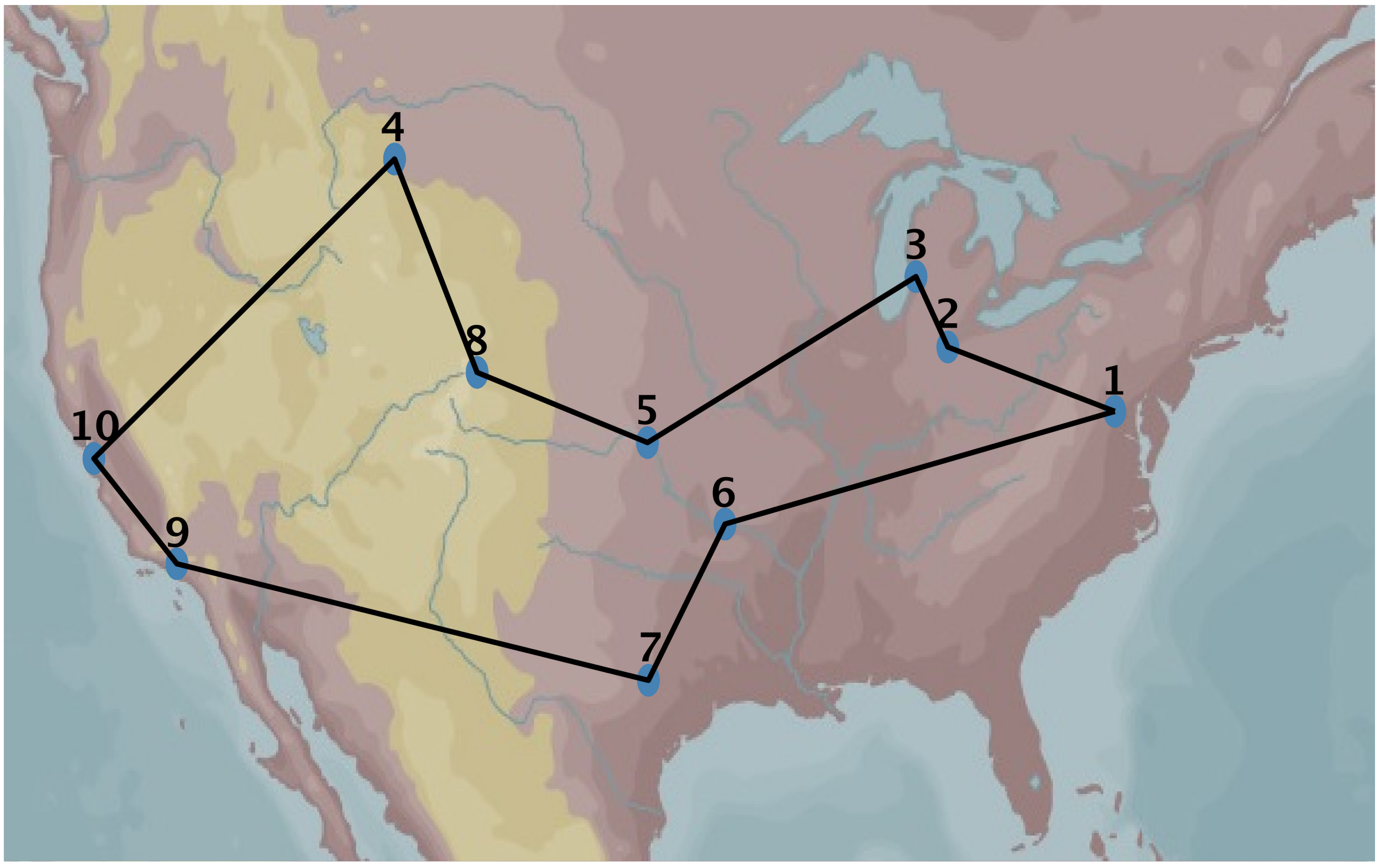
## Map with Optimized Route

> pathOrder := $\text{textplot}\left(\left[\text{seq}\left(\left[x\left(\text{lonlat}_{\text{tour}_i,1}\right), y\left(\text{lonlat}_{\text{tour}_i,2}\right)+1, \text{tour}_i\right], i=1..n\right)\right], \text{font} = [\text{HELVETICA}, \text{BOLD}, 12]\right)$ :

> optRoute := $\left[\text{seq}\left(\left[x\left(\text{lonlat}_{\text{tour}_i,1}\right), y\left(\text{lonlat}_{\text{tour}_i,2}\right)\right], i=1..n\right), \left[x\left(\text{lonlat}_{1,1}\right), y\left(\text{lonlat}_{1,2}\right)\right]\right]$ :

> connectingLines := pointplot( optRoute, connect = true, colour = "black", thickness = 2 ) :

> display( locMag, connectingLines, pathOrder, background = "USMap.jpg", size = [660, 331], view = [ −126 .. −66, y( 24) ..y( 50) ], axes = none)


4
3
2
8
1
5
10
6
9
7