Welcome
Map Hacking

Konstantin Greger
Tableau
@kogreger
kgreger@tableau.com

Sarah Battersby
Tableau Research
@mapsOverlord
sbattersby@tableau.com
Level setting

This is **NOT** a Jedi level session
DO NOT try this at home
Goals

Solve problems we have encountered (or heard from customers)

Fun ways to bend Tableau maps to your will

Accessible solutions

Some tricks *in* Tableau, some *with outside help*

Free and/or open source tools to complement Tableau

Reference materials for most, if not all, of what we demonstrate

  Blog posts, Public workbooks, Github repos with Python scripts, etc.
Taking advantage of new Tableau functionality, in fun, new ways
- Easy great circle routes with KML files
- Distance calculations using R and/or Set Actions

Alternative map types
- Dot density mapping with PostgreSQL

Map projections
- Alternative projections – creative lying about data, and manipulating background images

Augmenting flow maps
- Adding directional arrows
- Animating flows…along great circle routes
More demos than time – so we can’t do step-by-step… but, a few resources…

• [https://github.com/sarahbat/Tableau](https://github.com/sarahbat/Tableau) (geocoding, animating great circle paths)
  • Your caveat on all of these Github code bits – Sarah is not a software engineer ☺
Great circle routes
Straight line vs. great circle path

It’s easy to create a straight line path in Tableau.

But sometime the connection really isn’t a straight line on the map – you need the shortest path (great circle)

How do you deal with that?

Depends on which version of Tableau you are using…
Flights from Seattle – Using line mark type

Line mark type
Flights from Seattle – Using KML lines

Starting in 2018.2, line segments and polygon edges for certain spatial file types are rendered as great elliptic arcs (so close to a great circle that you won’t notice…)
The old way of making great circle arcs

Calculate the vertices (Python, R, Alteryx, etc.) and connect the dots in Tableau

• Chris DeMartini

• Allan Walker

• Gauthier Bonnot
  • [https://www.thedataschool.co.uk/gauthier-bonnot/different-ways-map-paths-tableau/](https://www.thedataschool.co.uk/gauthier-bonnot/different-ways-map-paths-tableau/)
  • [https://public.tableau.com/profile/gauthier2634#!/vizhome/FlyingfromFrance/Dashboard1](https://public.tableau.com/profile/gauthier2634#!/vizhome/FlyingfromFrance/Dashboard1)
Tableau 10.4 – support for linear geometry

No need for long tables of vertices – write each path to a single line

KML, Shapefile, GeoJSON, Hyper, TDE, etc.

Starting in 2018.2, line segments and polygon edges for certain spatial file types are rendered as great elliptic arcs (so close to a great circle that you won’t notice…)

Two point line is all you need!

```xml
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Placemark>
    <name> TC - TC Europe </name>
    <description> NOLA to Berlin! </description>
    <LineString>
      <coordinates>
        -90.0715,29.9511
        12.4050,52.5200
      </coordinates>
    </LineString>
  </Placemark>
</kml>
```
The KML geometry in 2018.1
The same KML geometry in 2018.2+
Easy to make KML in Excel (or Tableau Prep)

- Calculated field to concatenate all of the coordinates and identifiers into a KML placemark
<table>
<thead>
<tr>
<th>SegmentName</th>
<th>KML String</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEA-KOA</td>
<td><code>&lt;Placemark&gt;&lt;name&gt;SEA-KOA&lt;/name&gt;&lt;description&gt;AS 843&lt;/description&gt;&lt;LineString&gt;&lt;coordinates&gt;-156.0460052,19.73880005 -122.3088,47.4502&lt;/coordinates&gt;&lt;/LineString&gt;&lt;/Placemark&gt;</code></td>
</tr>
<tr>
<td>SEA-KOA</td>
<td><code>&lt;Placemark&gt;&lt;name&gt;SEA-KOA&lt;/name&gt;&lt;description&gt;DL 1096&lt;/description&gt;&lt;LineString&gt;&lt;coordinates&gt;-156.0460052,19.73880005 -122.3088,47.4502&lt;/coordinates&gt;&lt;/LineString&gt;&lt;/Placemark&gt;</code></td>
</tr>
<tr>
<td>SEA-OGG</td>
<td><code>&lt;Placemark&gt;&lt;name&gt;SEA-OGG&lt;/name&gt;&lt;description&gt;HA 29&lt;/description&gt;&lt;LineString&gt;&lt;coordinates&gt;-156.4299927,20.89859963 -122.3088,47.4502&lt;/coordinates&gt;&lt;/LineString&gt;&lt;/Placemark&gt;</code></td>
</tr>
<tr>
<td>SEA-OGG</td>
<td><code>&lt;Placemark&gt;&lt;name&gt;SEA-OGG&lt;/name&gt;&lt;description&gt;AS 805&lt;/description&gt;&lt;LineString&gt;&lt;coordinates&gt;-156.4299927,20.89859963 -122.3088,47.4502&lt;/coordinates&gt;&lt;/LineString&gt;&lt;/Placemark&gt;</code></td>
</tr>
<tr>
<td>SEA-OGG</td>
<td><code>&lt;Placemark&gt;&lt;name&gt;SEA-OGG&lt;/name&gt;&lt;description&gt;DL 1397&lt;/description&gt;&lt;LineString&gt;&lt;coordinates&gt;-156.4299927,20.89859963 -122.3088,47.4502&lt;/coordinates&gt;&lt;/LineString&gt;&lt;/Placemark&gt;</code></td>
</tr>
<tr>
<td>SEA-HNL</td>
<td><code>&lt;Placemark&gt;&lt;name&gt;SEA-HNL&lt;/name&gt;&lt;description&gt;AS 831&lt;/description&gt;&lt;LineString&gt;&lt;coordinates&gt;-156.4299927,20.89859963 -122.3088,47.4502&lt;/coordinates&gt;&lt;/LineString&gt;&lt;/Placemark&gt;</code></td>
</tr>
</tbody>
</table>
Flights of the World Part II: How to map great circle routes in the newest Tableau release
How about an animated path?

Allan Walker re-creates WarGames with Tableau and Mapbox

Dot density mapping
Typical way to represent population in Tableau

Choropleth or filled symbol map
Limitation…

For counts (e.g., count of people in a region), the choropleth map only gives one value for each polygon, which can be misleading in terms of true distribution. Consider population distribution by county vs. by tract.

This county (San Bernardino County, CA) has a relatively large population, but it’s mostly in one small corner.
Choropleth not always great for distribution

Dot density maps make it easier to show the smooth, continuous pattern of change with a phenomenon.
But, how do you make one in Tableau?

Not a standard map type in Tableau

Challenge to make dot density map put out by Lilach Manheim (Tableau Zen Master)

PostgreSQL to the rescue!
Steps

Grab some data from the US Census (or wherever you can get appropriate polygons and attributes)

Import the data to PostgreSQL

Custom SQL in Tableau
  For every geographic region…
  Figure out how many points per region…
  Randomly assign points within region…
  Dump that as a table of x,y coordinates to map in Tableau
Custom SQL to generate data in Tableau

Using a parameter to allow automated adjustment of the number of points used
Parameter defines how many \{phenomenon\} each point represents
  e.g., 1 point = 1000 people

The custom SQL doesn’t need to be split into two sections…this was just to clarify the steps

```sql
with dots_temp as (
    select geoid10, dp0010001, geom,
    (st_dump(st_generatepoints(geom, ceiling(dp0010001/ <Parameters.dotCount>)::numeric))).geom as pointDump
    from tract_2010
) 
select geoid10, dp0010001 as Pop,
    st_x(pointDump),
    st_y(pointDump)
from dots_temp
```
Some tips…

Calculate the point placement using the *smallest* reasonable geographic unit that you can.

Reasonable = can be calculated relatively quickly, but also provides enough geographic granularity

For US-scale I use Census Tracts

For state-scale I use Census Blocks
More information?

https://community.tableau.com/people/sarah.battersby.0/blog/2018/03/07/dot-density-maps-in-tableau-postgresql
Map projections
Tableau uses web Mercator base map

All spatial data will be re-projected to web Mercator for display
Sometimes you need something different…

Lambert conformal conic projection
AK and HI moved and scaled to keep in same view
How can you do this?

Short story
  Re-project your data
  Lie about the map projection (if Tableau *thinks* the data is web Mercator it won’t re-project)

But, make sure the coordinates in your new projection fit within the range of web Mercator (+/- 20 million)
Manipulating projections

I use QGIS (http://www.qgis.org – free, open source GIS)

Open spatial file

Save with new projection (and move geography, if needed)

Update projection definition so that it *thinks* it is web Mercator (it really isn’t)…but the data is all just coordinates on a plane…we just need Tableau to take the coordinates from your *real* projection and not alter it for display

Edit .prj file in Shapefile
Where to get the web Mercator .prj info

Epsg.io
Bring the new data into Tableau

This is your re-projected dataset, using the wrong projection definition so that Tableau thinks that it is in web Mercator (and so won’t try to re-project it)
More information?

https://community.tableau.com/people/sarah.battersby.0/blog/2017/05/12/working-with-projected-data-in-tableau-part-i-map-projection-basics

Multi-part series with background on projections, working with polar data, moving geography, etc.
Map Hacking

Konstantin Greger  
Tableau  
@kogreger  
kgreger@tableau.com

Sarah Battersby  
Tableau Research  
@mapsOverlord  
sbattersby@tableau.com