# Spatiotemporal Data

Lecture 20

# Spatiotemporal Data

- One of the many sources of big data is ***spatiotemporal*** datasets
- These datasets are multidimensional:
  1. Space (geographic location, x-y coordinate, etc.)
  2. Time (could be years, days, even microseconds)
- Besides space and time, a spatiotemporal data point isn't very useful without additional ***features***:
  - Name, Age, ID
  - Speed, Weight, Direction

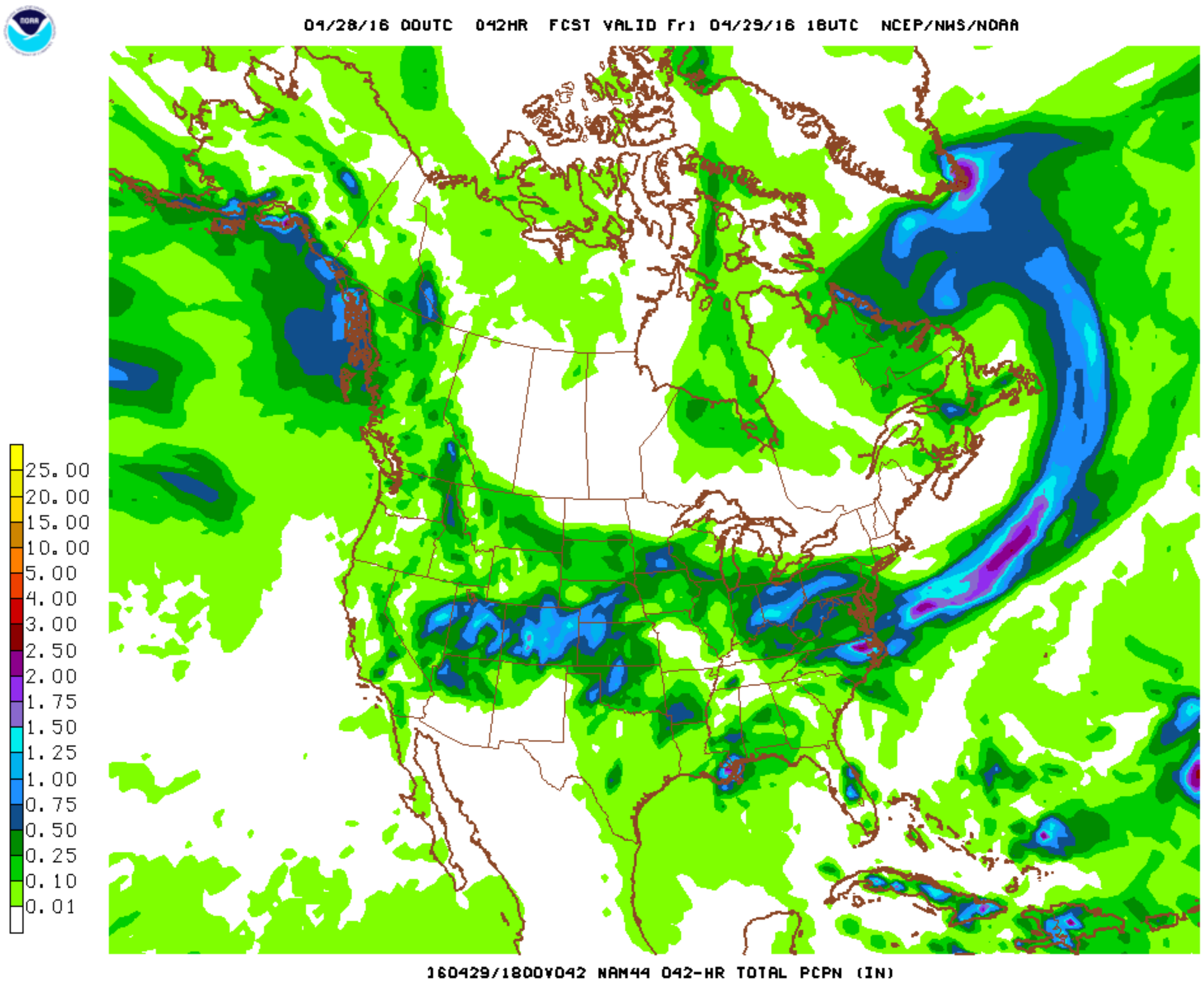
# Spatiotemporal Data Sources

- Geographic information systems
  - Electric usage in a city over time
- Object tracking systems
  - GPS, atomic clocks, speed, direction
- Multiplayer games
  - Player location, attributes
- Networked sensors and radars
  - Temperature sensor with Wi-Fi connectivity
  - Cloud cover or reflectivity readings

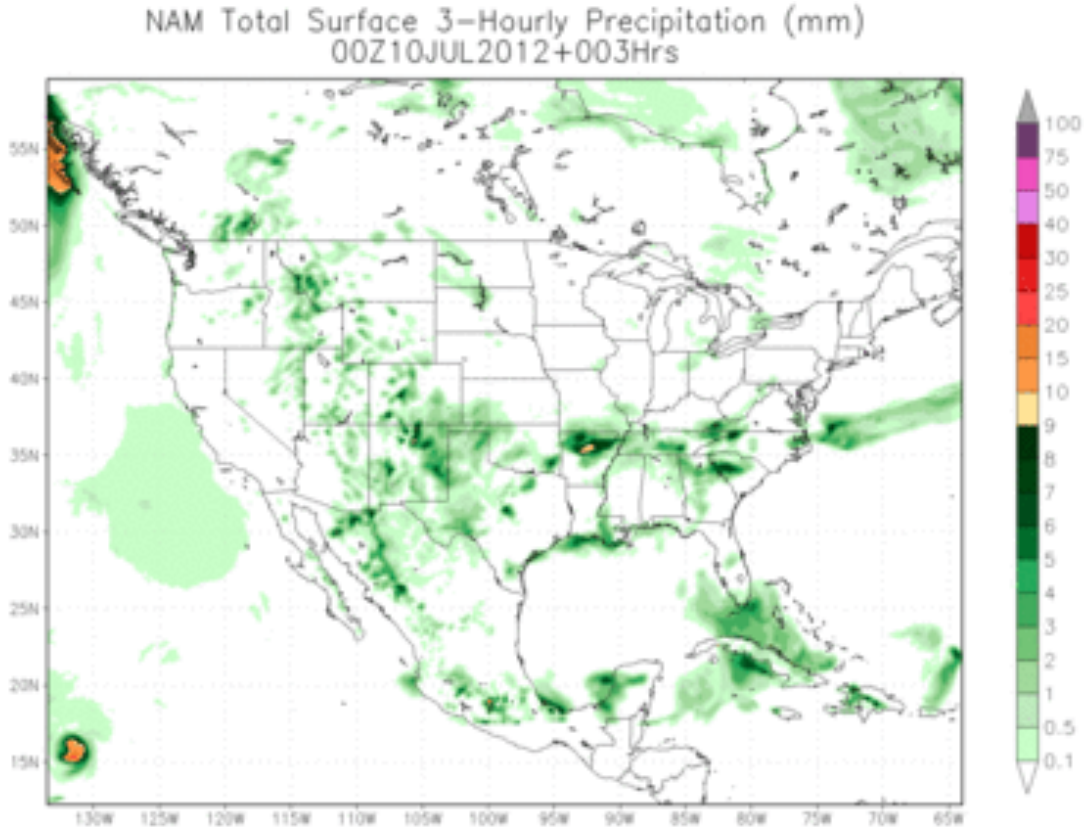
# P3 Motivation: NOAA Dataset

- Sourced from NOAA NAM Project
- Some Dimensions/Features:
  - Geospatial: Latitude, Longitude
  - Time Series: Start Time, End Time
  - Temperature
  - Relative Humidity
  - Wind Speed
  - Snow Depth
- ~2 PB (about 20 billion files)

# NAM Precipitation Snapshot



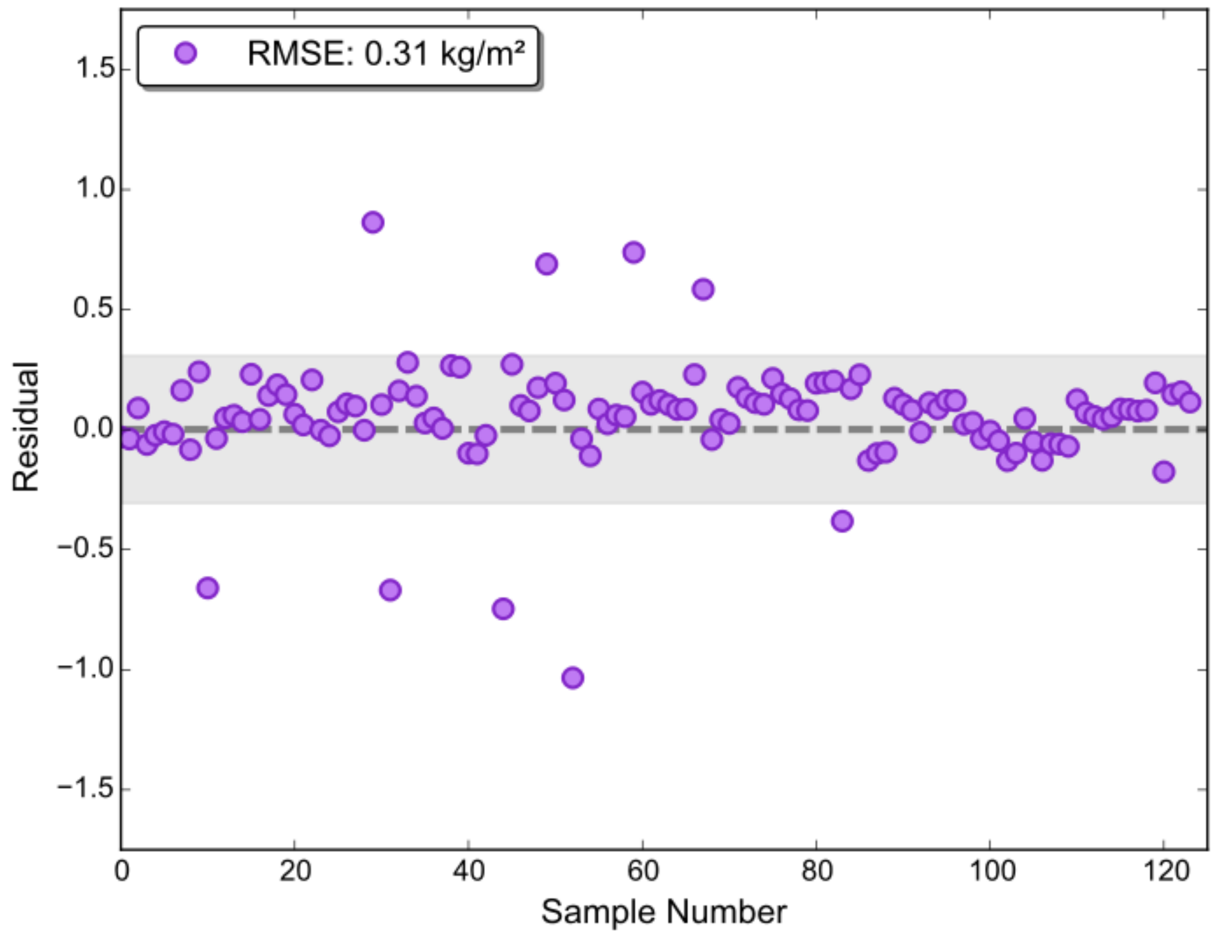
# Animation


NAM Total Surface 3-Hourly Precipitation (mm)
00Z10JUL2012+003Hrs

# NAM Dataset Applications

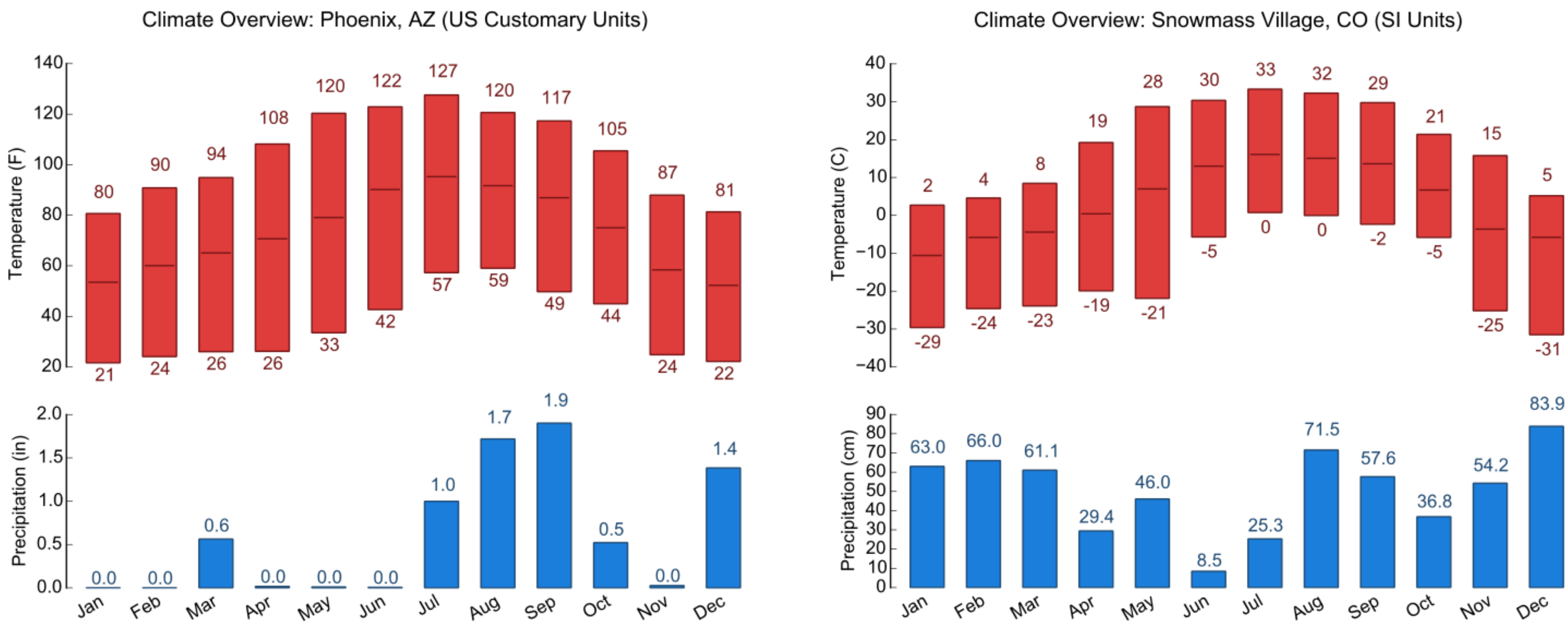
- Predicting future weather events or patterns
  - Machine learning
  - Statistics
- Summarizing Information
  - Visualizations
  - Reports
- Exploring relationships between features
  - How does the temperature influence humidity?
  - How does the location influence precipitation?

# Predicting Rainfall: Wyoming

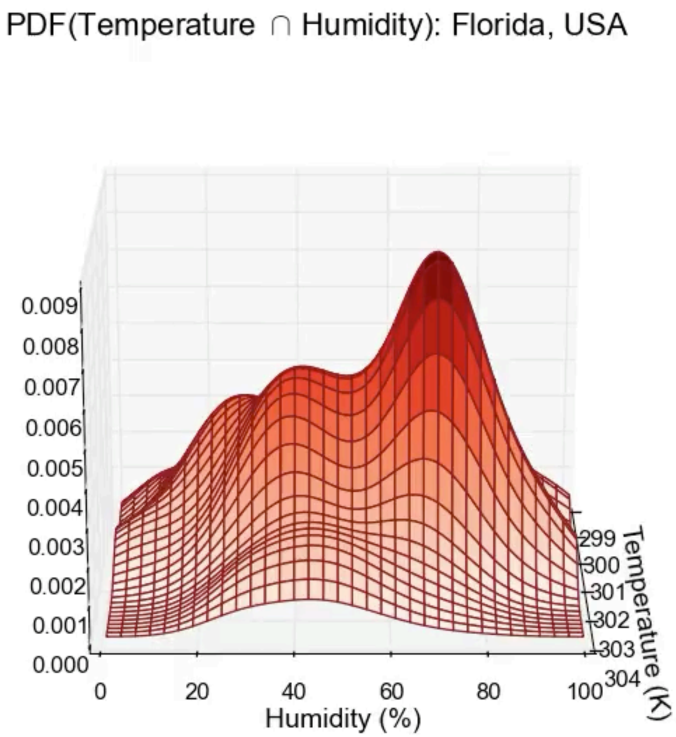


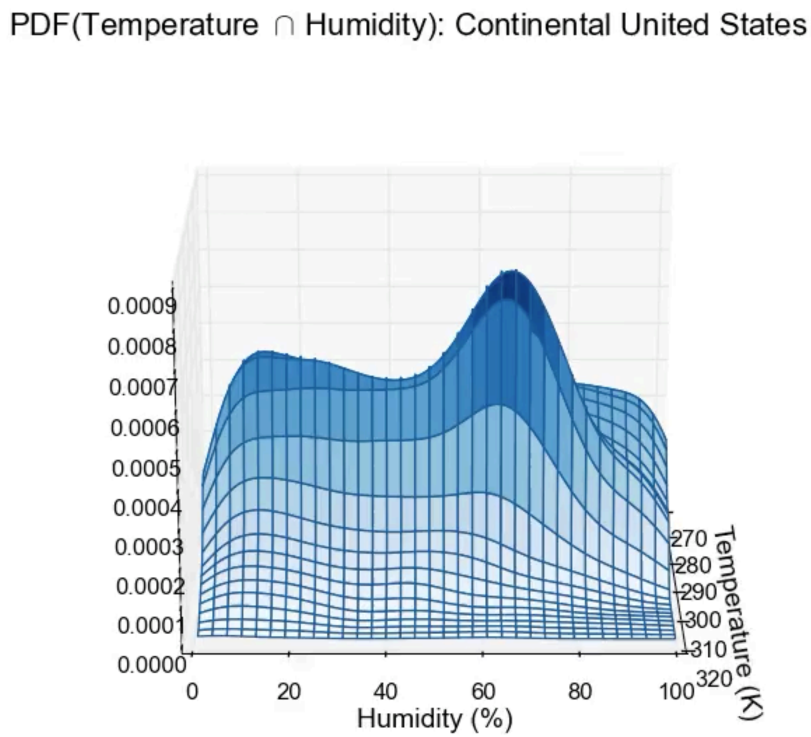
# Contour Visualization

# Climate Chart



# Relationships: Temp & Humidity



# Gathering Insights

- This dataset contains a wealth of information, but extracting insights from the data is challenging
- Multiple dimensions
- Storage requirements: where do we put all of it?
- Querying the data
  - (*knowledge discovery*)

# NAM Dataset Specifics

- The source data is stored in GRIB format
- Can be read and manipulated via the NetCDF library: https://www.unidata.ucar.edu/software/netcdf/
- We'll use a condensed version of the dataset for P3 instead

# Condensed NAM

- Tab-delimited features (no fancy formats)
- We'll consider only 2D feature data
  - No elevation information
- Each .tdv file contains a month of readings from the original source
  - nam_201501.tdv – January 2015

# File Format

- Each line contains a single record
- Each record begins with:
    1. Time of reading (UNIX timestamp)
    2. Geohash location (more on this later!)
- Then, the remaining features (~56)
    - null if no reading existed

# Some Interesting Features

- visibility
- pressure
- vegetation
- lightning
- temperature
- wind speed
- preciptation
- snow cover, snow depth

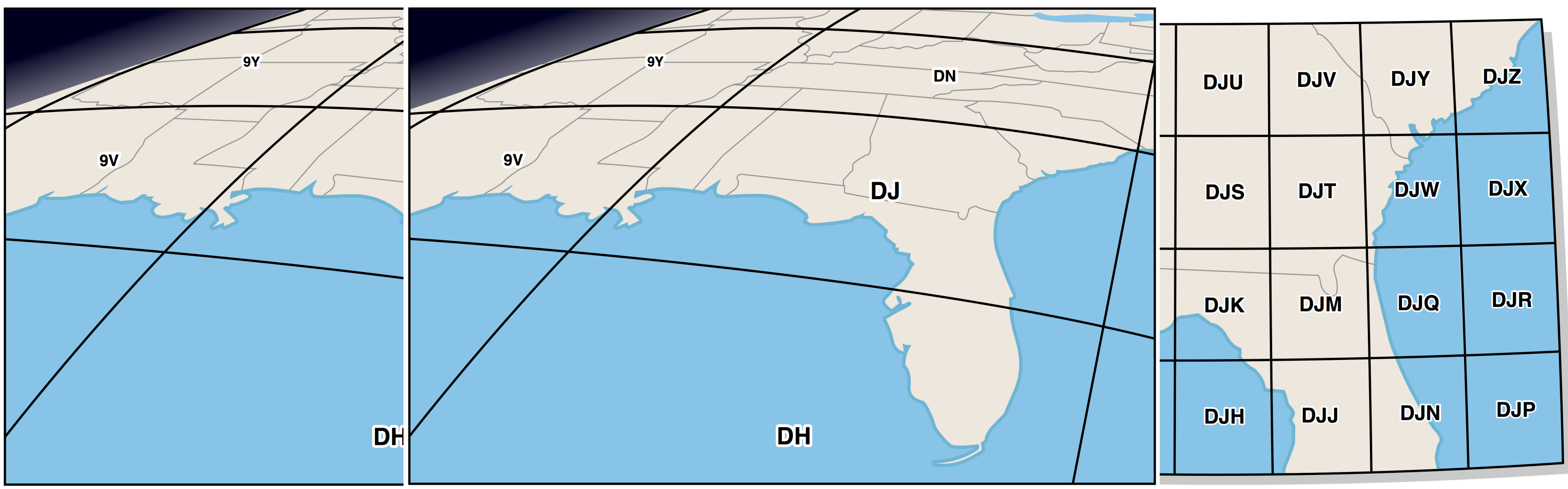
# Spatial Queries

- Querying spatial data is a whole subject in itself
- If I gave you lat-lon pairs in the dataset, you could use those to perform simple spatial queries
  - If lat is >= something && lat <= something else:

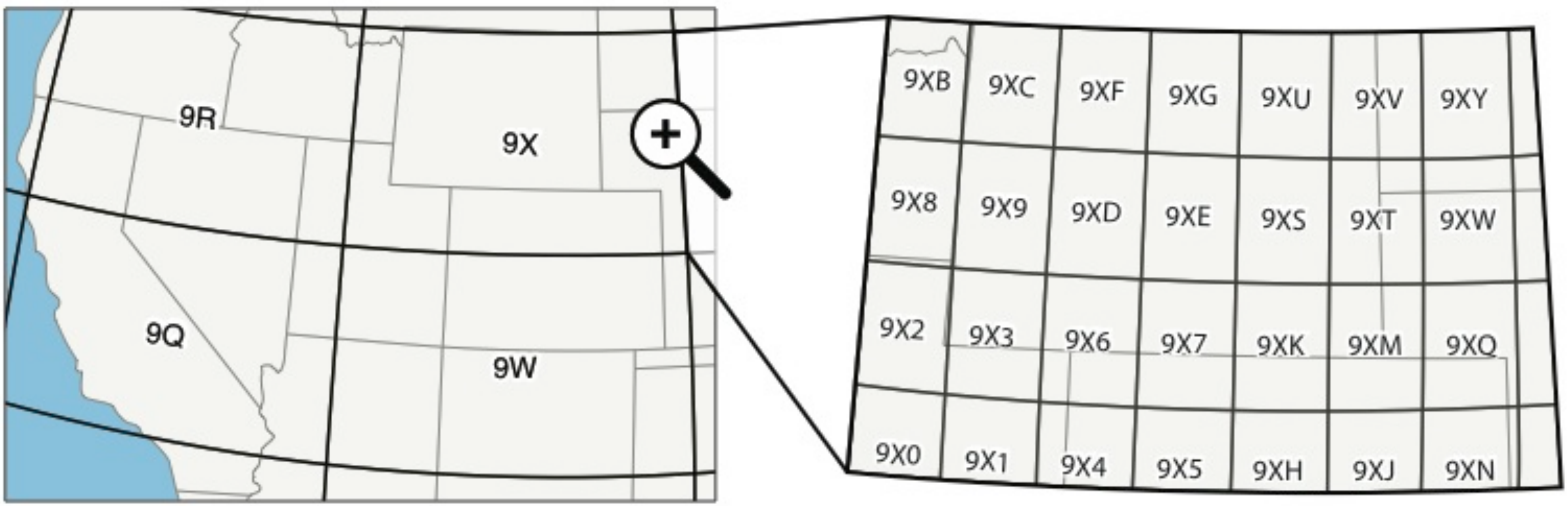
```
blah blah blah();
etc();
```

- A **better** option is to use the Geohash algorithm
  - Maps the earth to base32 strings
  - Defines a spatial **hierarchy** we can exploit

# The Geohash Algorithm (1/2)



# The Geohash Algorithm (2/2)



# Geohash Details

- We use the ***Geohash*** algorithm to represent the spatial location associated with our sensor readings
  - Maps 2D spatial locations to 1D strings
  - Precision is determined by string length
- 9Q8YVT1F ➔ Kudlick Classroom
  - Similar string prefixes refer to similar locations
- Want to support range queries? Just match more or less of the string prefix

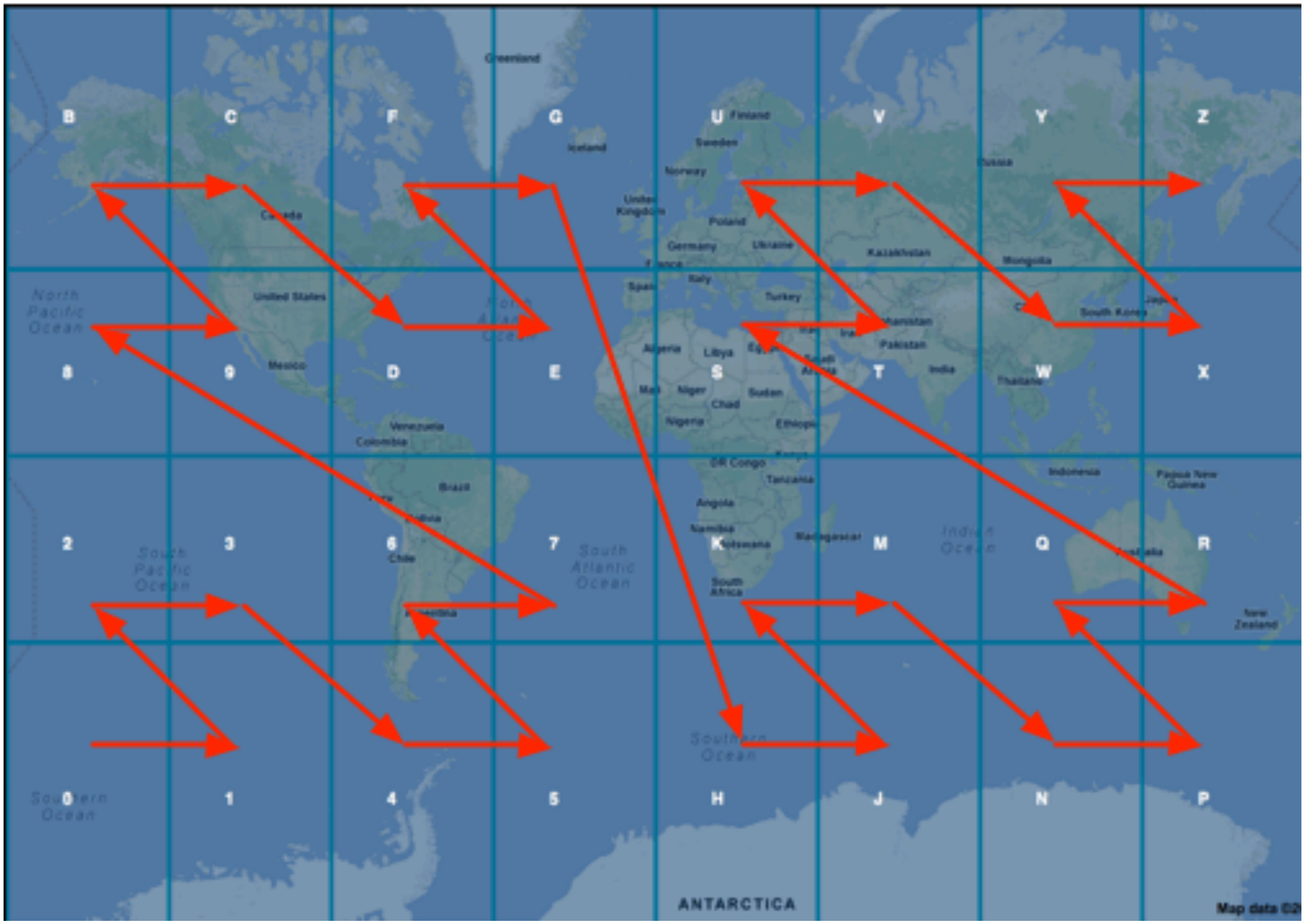
# Geohash Resolutions

- Spatiotemporal data is not always evenly distributed
  - Compare the density of New York City and Glenwood Springs, Colorado
- Hash: 9XJQBF
  - 9XJQ = 20x30 km
  - 9X = 600x1000 km
- However, for our dataset, the data **IS** evenly distributed!
  - Fixed grid

# Geohash Implementation

- Divides the bounding boxes in half with each binary bit added to the string
  - 1 bit = left or right half of the earth
  - 2 bits = top or bottom half of the left/right half
  - And so on…
- Uses 32 alphanumeric characters (Base 32)
  - 32 characters = 5 bits per character (5 divisions)
  - Omits some letters to avoid forming words

# Z-Order Curve



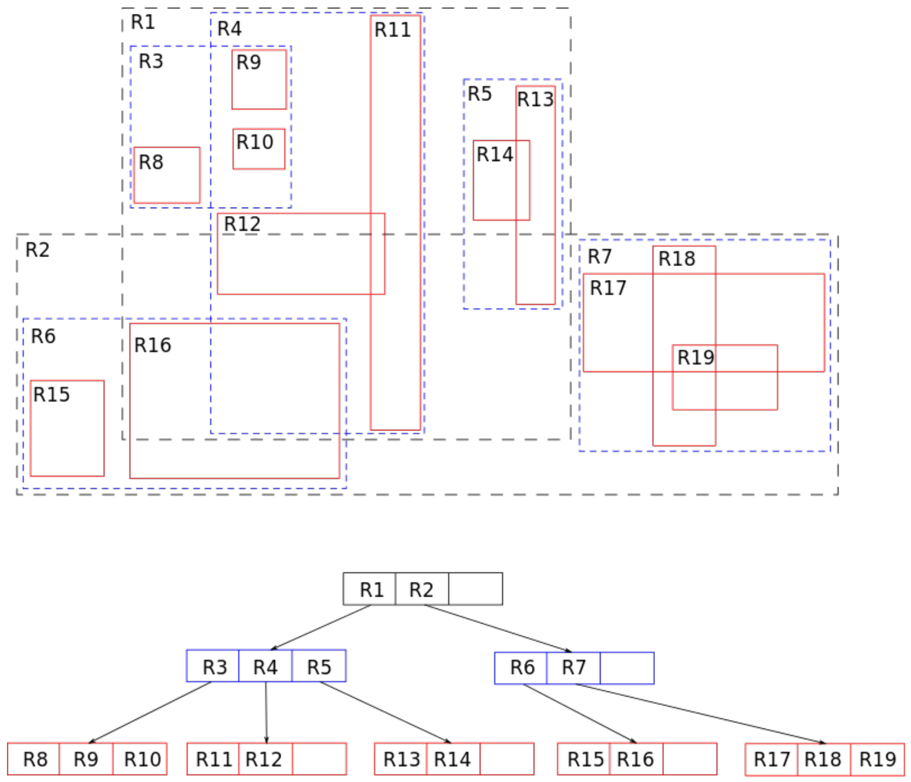

Source: http://www.bigdatamodeling.org/2013/01/intuitive-geohash.html

# Geohash Fun Facts

- Originally designed to allow users to share short URLs that represent locations
- Similar implementations have been used to identify locations for businesses, government
  - Ireland's proof-of-concept ***openpostcode*** can uniquely identify all locations within the UK

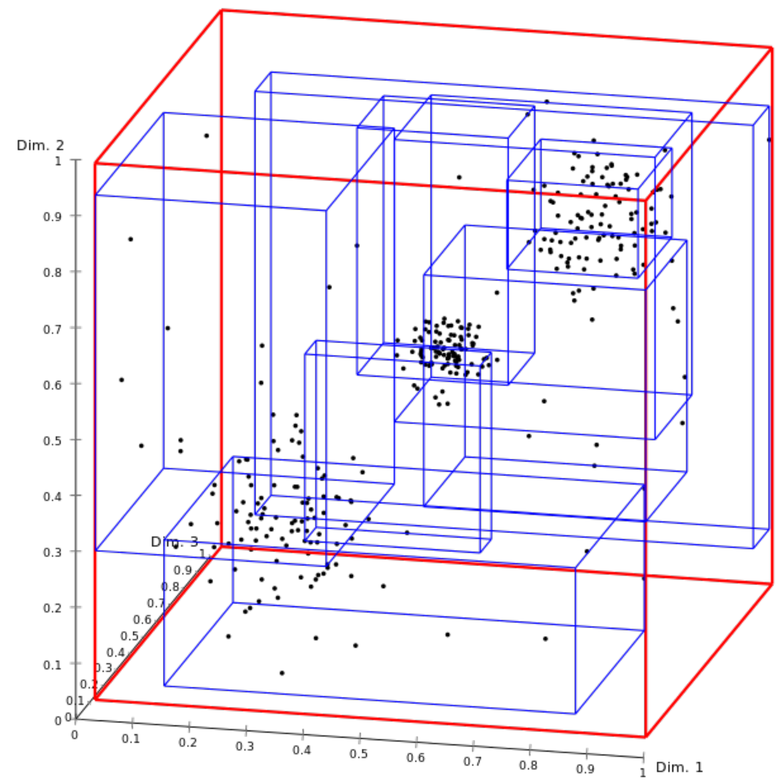
# Spatial Indexing: R-Trees

- **R-Trees** are a widely-used spatial index
- Share many similarities with B-Trees, but support spatial features:
  - Multiple dimensions
  - Intersection, containment queries
  - Nearest neighbor search



# R-Tree Drawbacks

- R-Trees can be overwhelmed by extremely large datasets
- Query performance decreases as the number of leaves in the tree expands
  - Too much precision



# Brute-force Approach

- Submit queries to all nodes, scan over records
  - **This is what we can do in P3…**
- Issues:
  - Wasted processing on nodes with no matches
  - Not scalable
  - High latencies
- A better approach: ***decompose*** the query and distribute it only to relevant nodes… How could we do this?

# One More Thing

- The challenge in P3 is not the questions we ask about the data
- It’s actually being able to get the answers in a reasonable amount of time
- You will likely spend more time thinking about *how* your computations should work than what needs to be done to get the answer… if that makes any sense.