Where Should I Move?

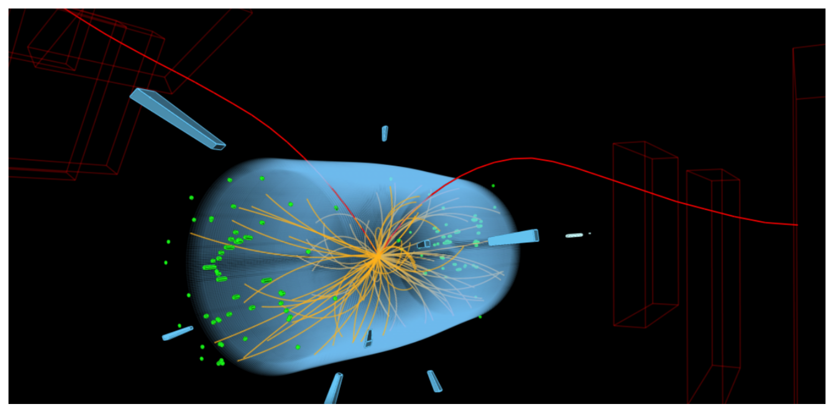
Douglas Kelley
W205-2
Fall 2014 Final Project

Binary vs Text Data

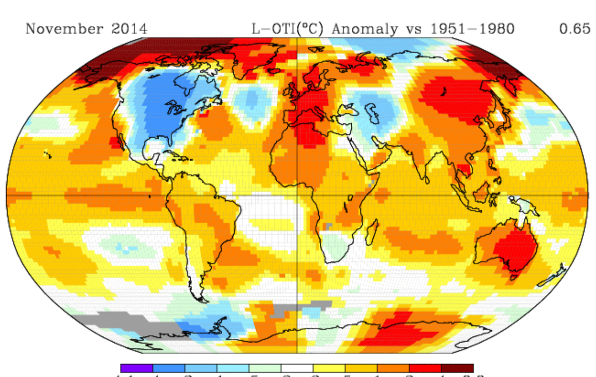
Text data may be ~90% of data generated

Binary data is still important in scientific contexts

http://www.humanconnectomeproject.org/gallery/



http://home.web.cern.ch



http://data.giss.nasa.gov/cgi-bin/gistemp/nmaps.cgi?sat=4&sst=3&type=anoms&mean_gen=11&year1=2014&year2=2014&base1=1951&base2=1980&radius=1200&pol=rob

http://mashable.com/2012/06/22/data-created-every-minute/

Stored in a variety of formats

Analysis generally requires context

Given climate model data, can we identify locations that at some point in the future will have climate similar to a particular location in the past?

Use maximum and minimum temperature and precipitation as proxies for “climate”

Identify “similarity” based on normalized correlation coefficient to capture annual variation

How Is the Data Organized?

- Downsampled climate projections (2006-2100) based on 4 climate scenarios, and historical projections (1950-2005) using the General Circulation Model
- Precipitation, maximum surface air temperature, and minimum surface air temperature calculated monthly
- Averaged spatially over 30 arc-seconds and temporally as a 5 year moving average
- Written to S3 as NetCDF4 files

NetCDF4 = HDF5 = "a filesystem for your data"

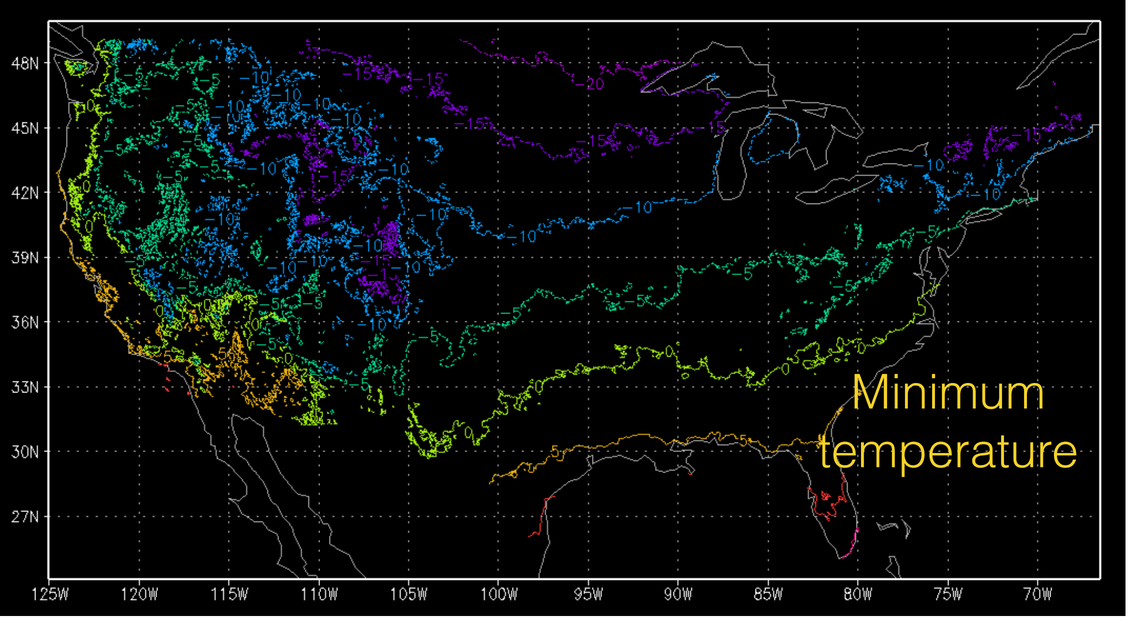
/NEX-quartile

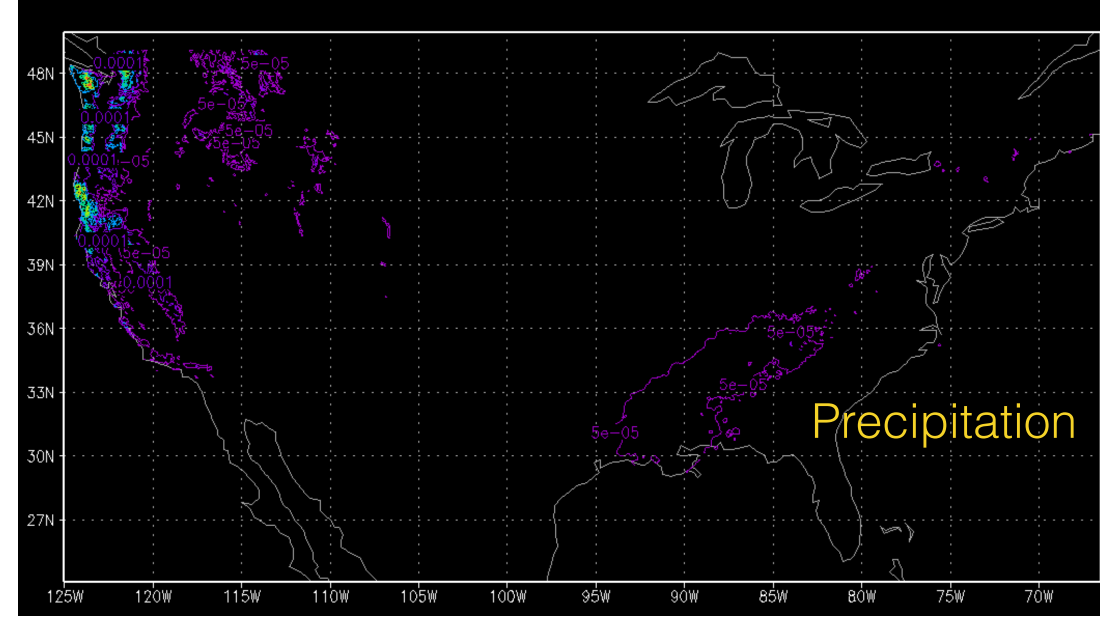
/historical   /rcp26   /rcp45   /rcp60   /rcp85

/mon/atmos

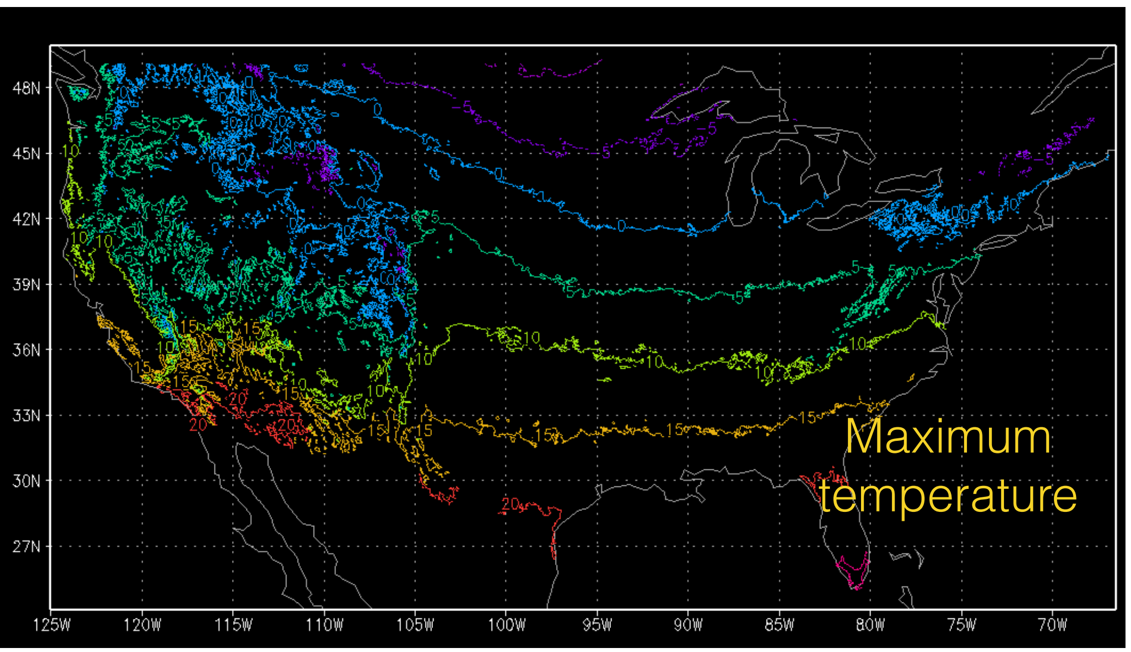
/tasmin   /tasmax   /pr

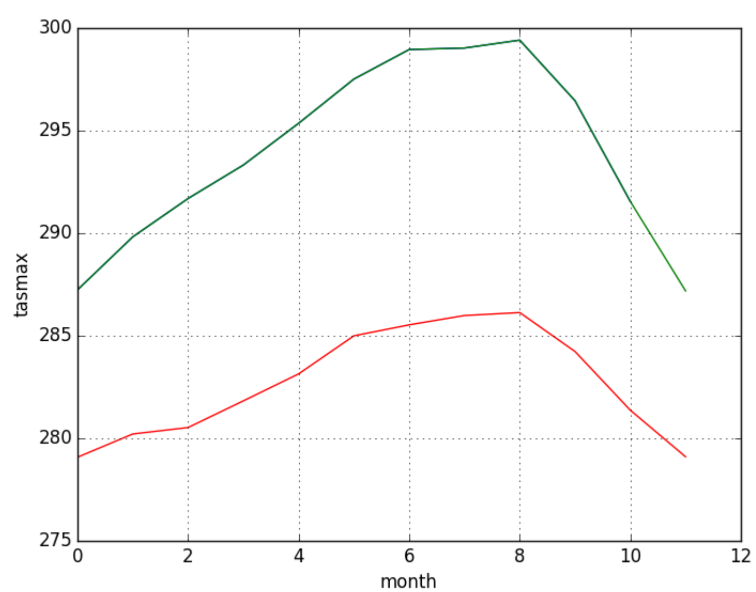
/r1i1p1/v1.0/CONUS/

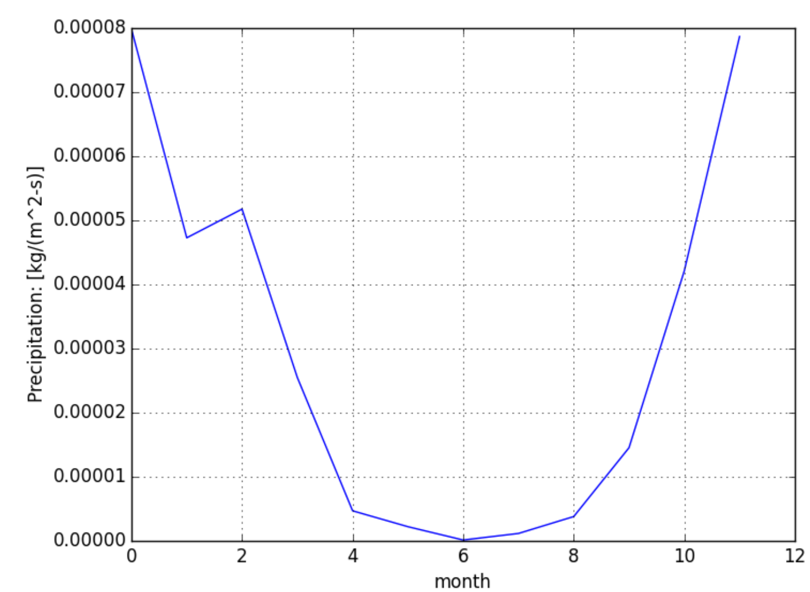




January 2005 data

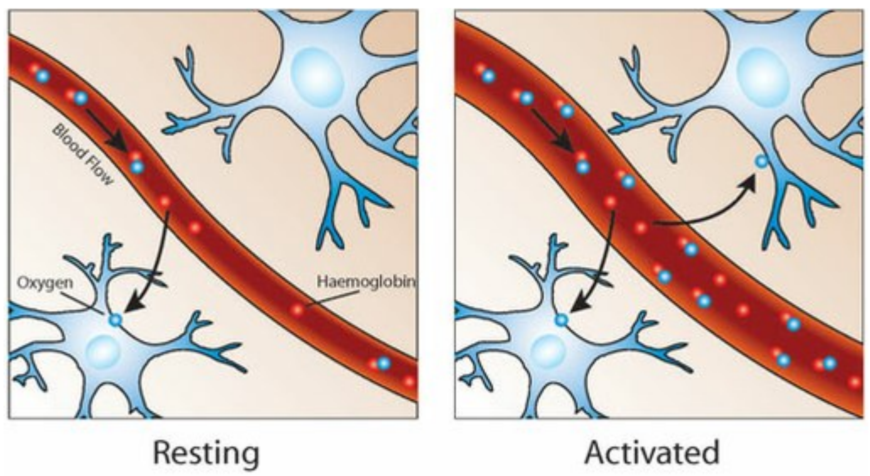




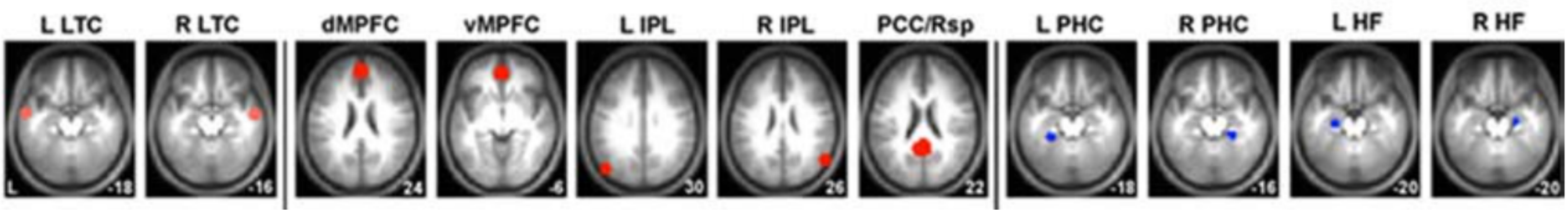


How is this like fMRI?

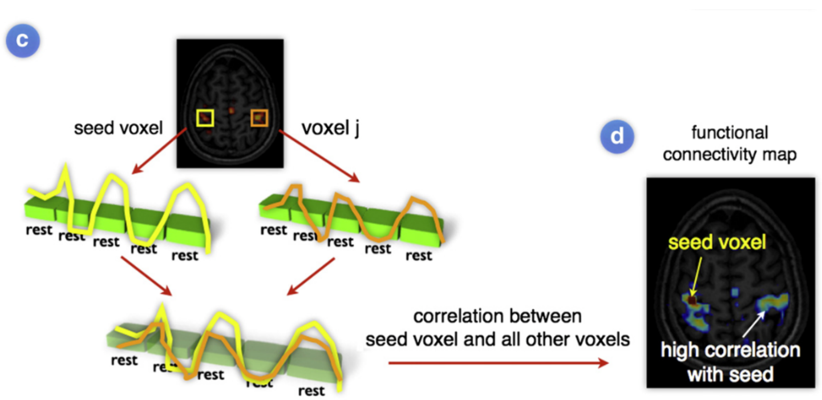
1. The cerebral cortex (grey matter) is organized into a set of patches distinct in function and anatomy.
2. These patches are connected into distinct networks by myelinated fibers (white matter). These networks often include both cortical and subcortical structures (e.g., hippocampus, amygdala).
3. Activation of a network produces a local change in blood flow that dominates any increased metabolic demand (BOLD effect); activation occurs across entire network so connected points show similar time courses



http://www.fmrib.ox.ac.uk/research/education

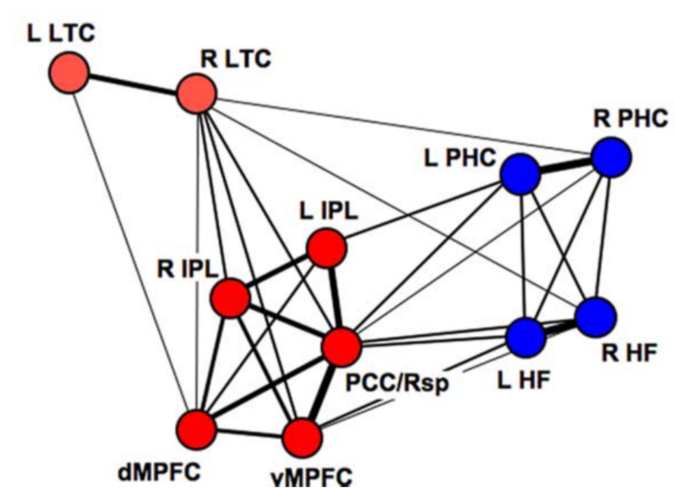


Buckner, et al. http://onlinelibrary.wiley.com/store/10.1196/annals.1440.011/asset/annals.1440.011.pdf?v=1&t=i3tpn53n&s=fdd6d59cd8f2500196dafe46ea19c54dddc03cc0



van den Heuvel and Pol

http://www.sciencedirect.com/science/article/pii/S0924977X10000684



General Strategy

- Pick a seed point and extract a target signature
- Split the data files so that a mapper only deals with one point at a time
- Correlate the point against the target and split out the correlation
- Aggregate across all the points and display a surface plot of the scores to identify the best match

Key Issues

- Hadoop doesn’t like binary data
- BIG files - 2GB each
- Getting one record means going to a specific point in a file

Strategy

- Use h5py — doesn’t try to load the entire file at once
- Use s3fs to mount NEX-DCP30 data as “local” files
- Create a text key to specify what point each mapper must examine
- Files are organized by variable and scenario; only have collisions if two mapper tasks need to access the same file at the same time

Local Execution

- Use mrjob to setup the task
- Download one set of files (3x2GB) as examples
- Store the target in a dictionary (3x12 floats) read from text files
- Identify a point as a slash-delimited string of scenario/variable/year/latitude_index/longitude_index
- Figure out which file to open
- Calculate the scaled correlation coefficient
- Use the point take as the key and the correlation coefficient as the value

python genTags.py 2024 1664 300 10 1 | python where2Move.py

4 seconds to evaluate 3 tags on 2.9GHz i5

Takes longer to generate tags

Bootstrap

Need to install a few things:

- HDF5 libraries
- python 2.7
- cython
- mrjob
- h5py
- s3fs-fuse

Then we mount the NEX DCP30 datafiles
Add some swap space
And create the target signature files

bootstrap.sh

Mapper

- mrjob doesn't seem to work remotely, so split off the mapper and use the EMR web interface to set up the job

- Convert the map task to pull input from stdin

- Handle the target signatures by checking for keys in the target dictionary and loading them from a file if they're not loaded in the mapper instance

- Handle invalid data as well

files s3://kelley-w205/Project/corrMapper.py -mapper corrMapper.py -reducer NONE -input s3://kelley-w205/Project/Input/tags64.txt -output s3://kelley-w205/Project/Output/8dec2014_5/

Execution vs Dataset Size

Medium cluster

- 50 minutes to boot
- 25 minutes for a single point
- 92 minutes for 4 points
- 181 minutes for 25 points

Large cluster

- 31 minutes to boot
- 122 minutes to process 64 points

XLarge cluster

- 29 minutes to boot
- 73 minutes to process 128 points

XLarge cluster with 4 cores

💥😖 Need to handle file collisions better

Aggregation and Visualization

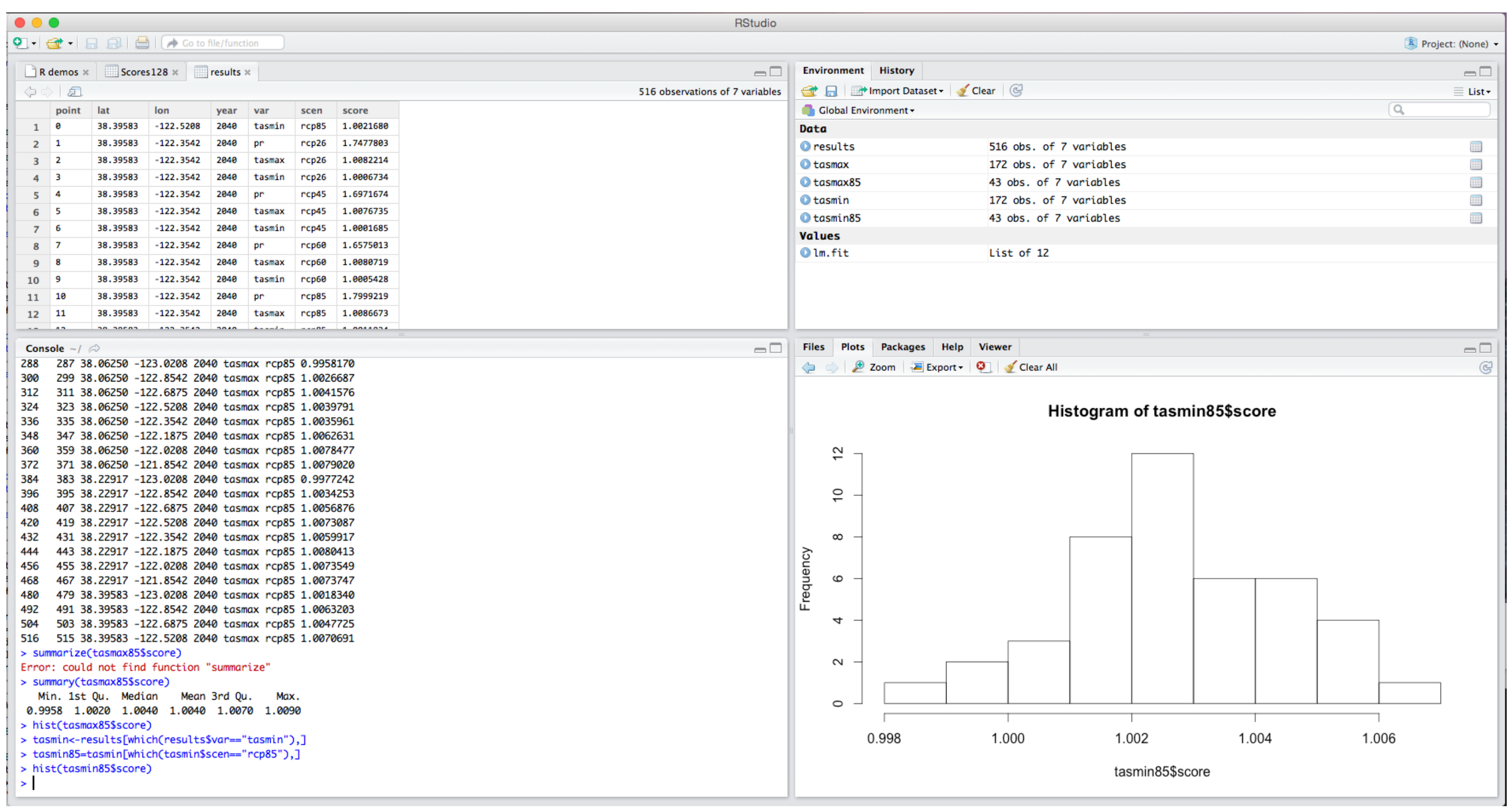
Catenate part000* files to a single text file

Parse tags and convert to CSV

Load into R and analyze

Use maps library to display onto map

Given enough data, load into NetCDF4 file and use GRaDS



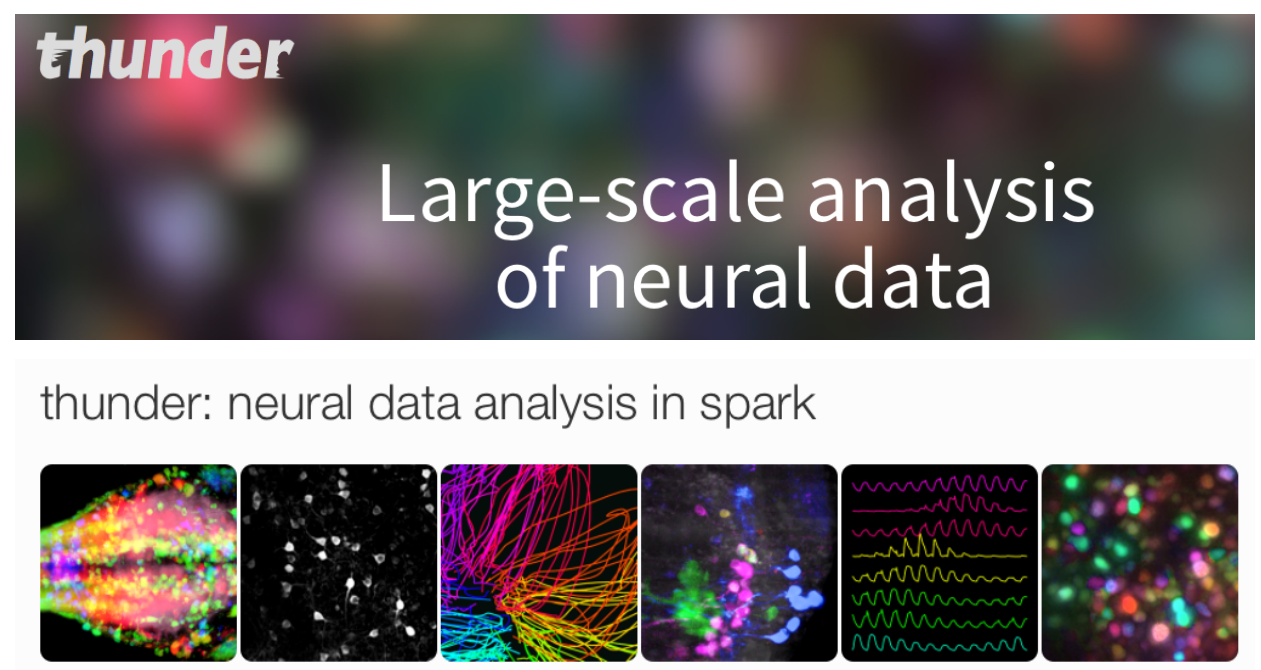
Conclusions and Next Steps

This is really slow — need a much bigger cluster and a more intelligent way of handling collisions between the datafiles

Sloth could be due to s3fs

More native implementation in Java might be faster

Spark library — thunder — designed for analyzing electrical recording data



http://thefreemanlab.com/thunder/docs/