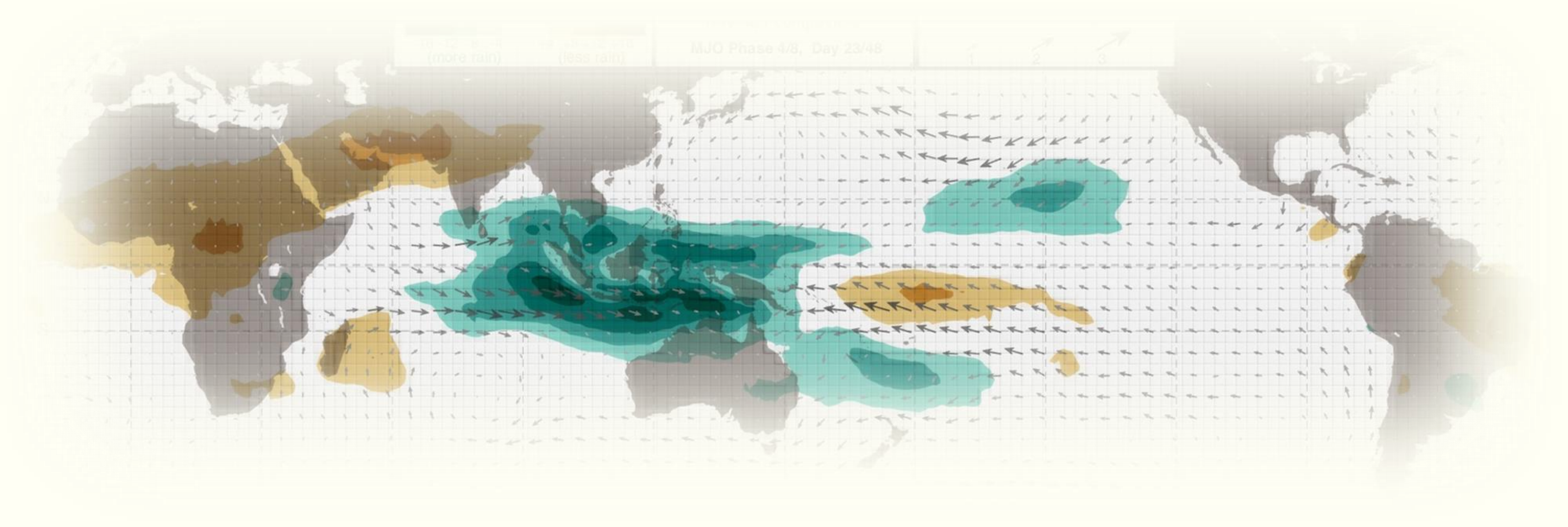
A high-resolution, high-frame rate, clarity-optimized geophysical data visualization of Australian monsoon rainfall and wind

B. Jason West
Ph.D. Candidate
Atmospheric and Oceanic Sciences
University of Colorado Boulder



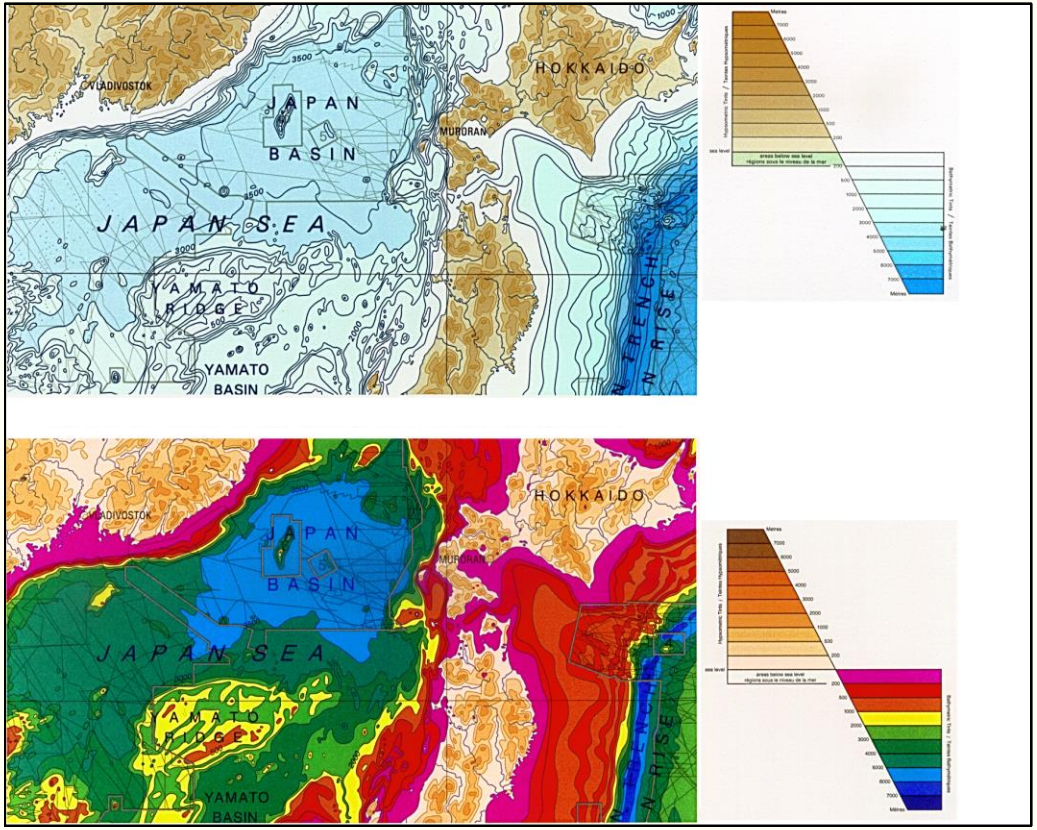
Background

- When I started my “day job” as a grad. student researching the Asian monsoon, I set a goal for myself to publish a dataset on NOAA’s Science on a Sphere (SOS).
- The idea of creating content for these platforms has driven my visualization work for the past several years.
- I will briefly discuss two design considerations that drove my work:

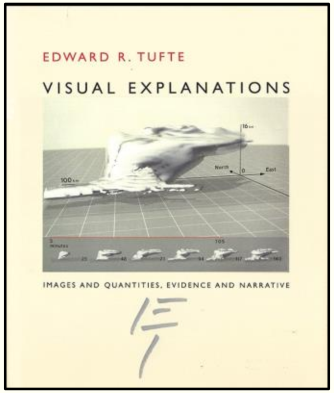
http://www.christopherstevens.cc/blog/2012/12/18/science-on-the-sphere

1. **Clarity-optimization**. The design of the animation should convey the maximum information with the least amount of mental effort required of the viewer.
2. **Maximization of computing resources,** so that the animation can be generated on a laptop, but still be high-resolution and high-frame-rate (smooth images and image sequences are easier on the eyes)**.**

Colors can vividly tell the story of geophysical data…



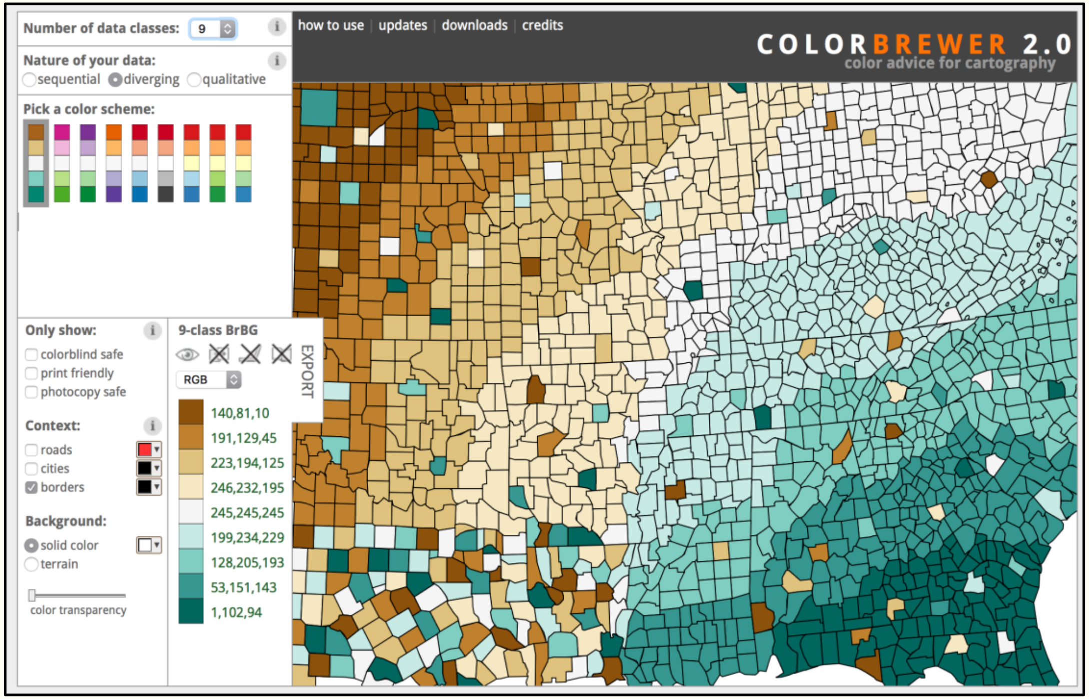
From *Tufte* [1997]



…or lead to confusion.

How to communicate effectively with color without being a color expert

- There are excellent tools available, such as Color Brewer (at right).
- Several varieties of color schemes are available; RGB, HEX, and CMYK values can be transferred directly into the visualization software of choice.
- Colors have been optimized for the human visual perception system, as opposed to arbitrary physical-based color scales.
- Colors can be filtered for colorblind-safe-only versions.



Number of data classes: 9

Nature of your data: ○ sequential ● diverging ○ qualitative

Pick a color scheme:

Only show:
□ colorblind safe
□ print friendly
□ photocopy safe

Context:
□ roads
□ cities
☑ borders

Background:
● solid color
○ terrain

color transparency

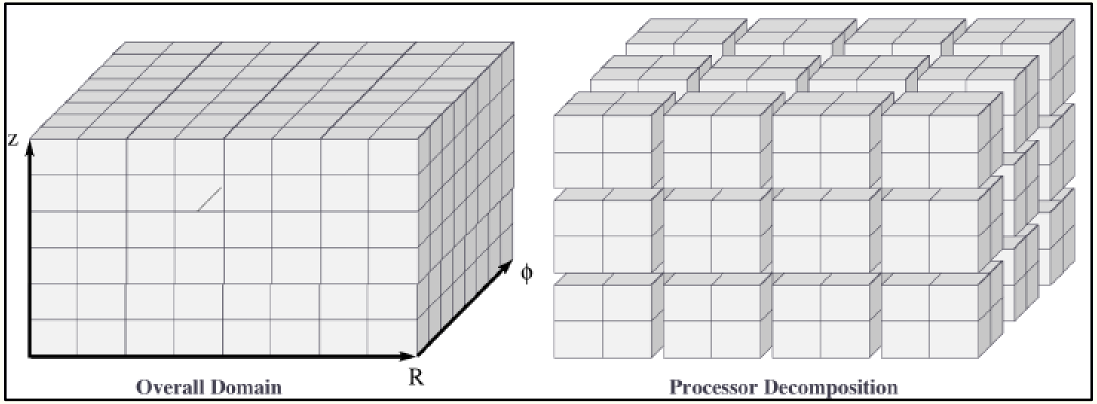
9-class BrBG

RGB

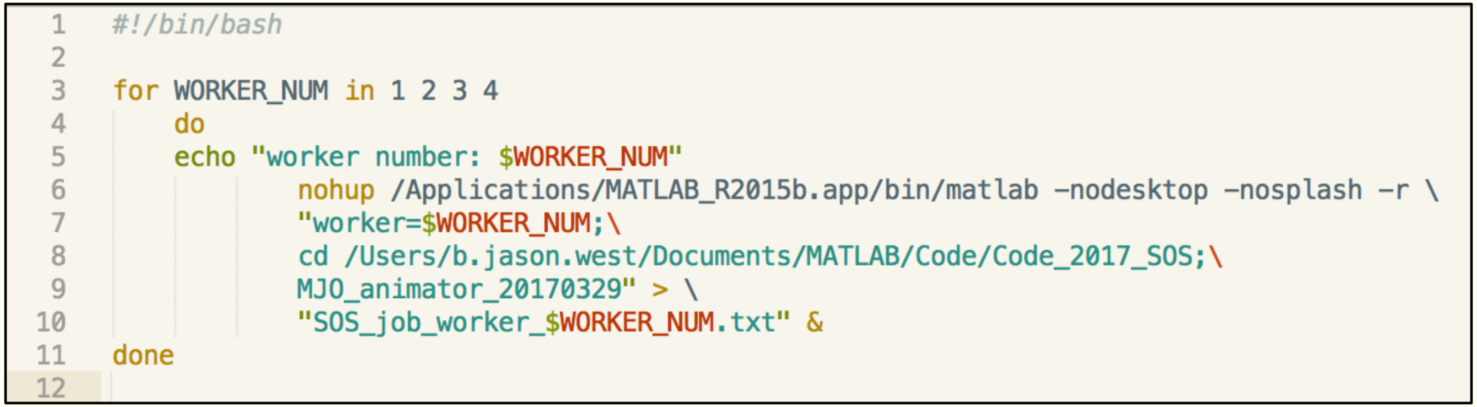
🟫	140,81,10
🟫	191,129,45
🟨	223,194,125
🟨	246,232,195
⬜	245,245,245
🟩	199,234,229
🟩	128,205,193
🟩	53,151,143
🟩	1,102,94

ColorBrewer2.org

"Hacked" parallel computing on a laptop



- A simple, but effective, way to do domain decomposition with Matlab or similar software package is to run multiple instances of the code, each using its own core and memory for a different chunk of the problem domain.
- This is easily accomplished through shell scripting in under a dozen lines, resulting in a ~4x rendering speed increase on a quad-core laptop:



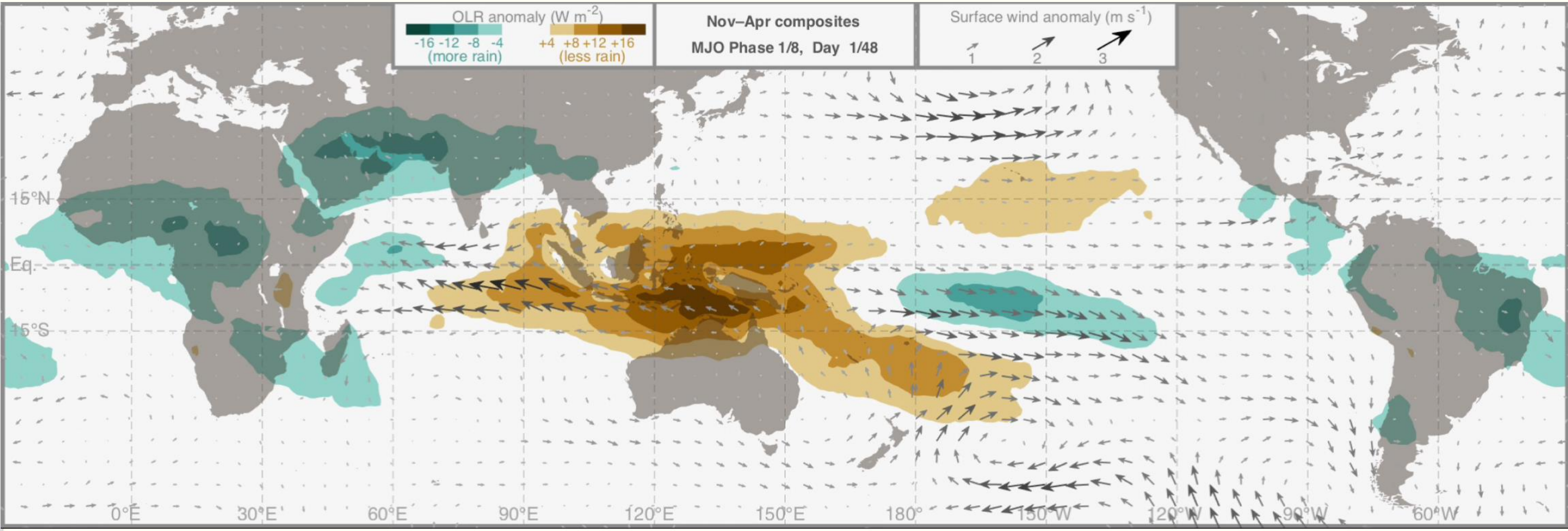
```bash
#!/bin/bash

for WORKER_NUM in 1 2 3 4
    do
    echo "worker number: $WORKER_NUM"
            nohup /Applications/MATLAB_R2015b.app/bin/matlab -nodesktop -nosplash -r \
            "worker=$WORKER_NUM;\
            cd /Users/b.jason.west/Documents/MATLAB/Code/Code_2017_SOS;\
            MJO_animator_20170329" > \
            "SOS_job_worker_$WORKER_NUM.txt" &
done

```

The visualization

*Note: I did not include the animation here due to data size constraints. Please contact me at b.jason.west@colorado.edu if you're interested in seeing the animation.



Quick stats:

- **Resolution**: 2046 x 683 pixels (1.4 MP; btw. 720p and 1080p)*.
- **Frame rate**: 30 fps (576 frames).
- **Rendering time**: 2 hours (parallel); 8 hours (serial).

*The SOS version is 4096 x 2048 (~8.4MP; ~4K res.) and was rendered in 20 minutes.

Data are from NOAA, NASA, and the Australian Bureau of Meteorology.

Thanks for your attention!

Helpful resources:

- EdwardTufte.com
- ColorBrewer2.org
- CU Research Computing workshops

Contact: b.jason.west@colorado.edu