TEXAS TECH UNIVERSITY

Real-Time and Deep Dive Analyses with lmatools

Eric Bruning
TTU Department of Geosciences
Atmospheric Science Group

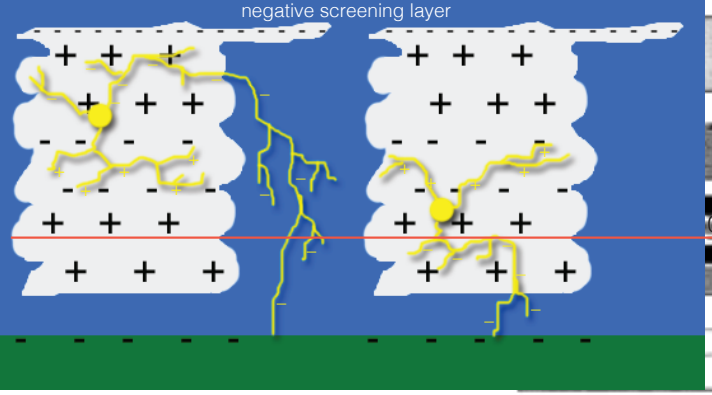
Contributors: Vanna Chmielewski,
Vicente Salinas, Phillip Ware, Matthew Brothers

GLM Science Meeting, Huntsville, AL
27-29 September, 2016

WHY DEEP DIVE? HOW DO LMA DATA FIT?

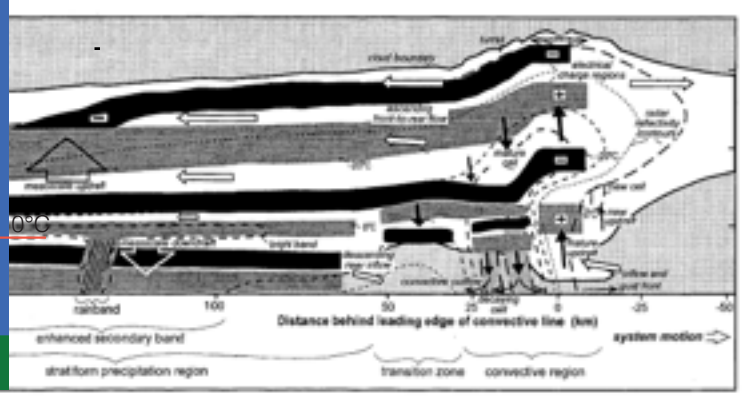
A lightning physicist's deep dive:

- What is a lightning flash? How and when does it emit light?
- How does the cloud scatter this light?
- GLM's relationship to other lightning datasets
  - *Validation: Should GLM have seen something when another system detected a flash?*
  - *Operations: How do I use GLM alongside ground network data?*
- LMA data are useful in this context as a connecting lattice which can reliably and continuously connect datasets to one another: *continuous* measurements of *flash rate* and *extent*

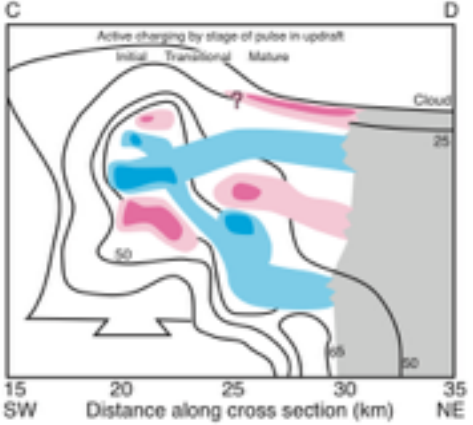


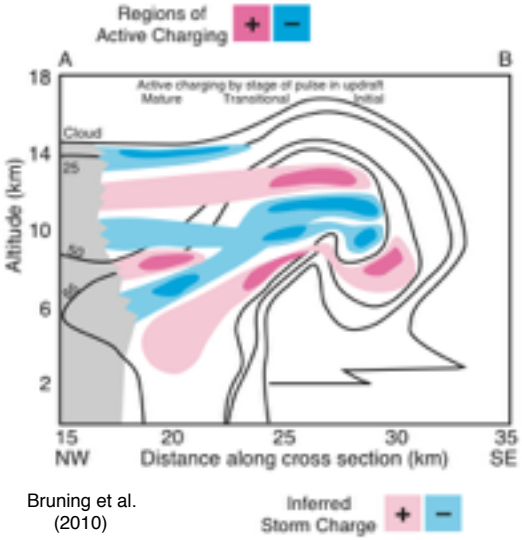
+ cloud flash (+IC) or bolt-from-blue –CG

–IC or –CG



Stolzenburg et al. (1998)





Bruning et al. (2010)

Inferred Storm Charge

A meteorologist's deep dive:

- How do we connect lightning to the *time scales* and *spatial extents* which characterize storm dynamics, cloud microphysics, and other processes?

GLM: the first *continuous* observations of lightning's optical pulses over *large areas*

LMAs: *continuous* observations of lightning's VHF pulses over *large areas*

We have learned to use LMAs beyond counting flashes; how do those patterns translate to GLM?

- Translates to day-0 user-readiness guidance

Apply similar methods to GLM data and see how it differs from our sense of how LMA data look.

What does lmatools do?

lmatools: concrete implementation of a data analysis methodology for distilling how lightning fills 3D space and changes with time

- Reads, processes, analyzes Lightning Mapping Array data.
- Open source: https://github.com/deeplycloudy/lmatools
- Python language

Suitable for real-time, batch or interactive analyses.

Summer 2016: Better organization and simplified code. Easier to learn and extend.

USES TO DATE

- Contains code to reproduce key analyses and plots described in publications
  - *Bruning and MacGorman (2013, AMS)*
  - *Ware (2015, TTU MS Thesis)*
  - *Bruning and Thomas (2015, JGR)*
  - *Fuchs et al. (2015, 2016, JGR)*
  - *Chmielewski and Bruning (2016, JGR)*
- Used to flash sort and grid entire WTLMA archive (2012-present) and process DC3, KTaL, and VORTEX-SE field campaign datasets

KEY COMPONENTS

Reader for ASCII VHF source data file

Flash sorting and flash area calculations

- HDF5 file format: stores flash meta-properties (e.g., area) and corresponding original VHF events and their parent flash
- Can tag flashes with CG / IC designation

Gridded products

- CF-compliant NetCDF

GUI tool to manually track cells by stepping through grids

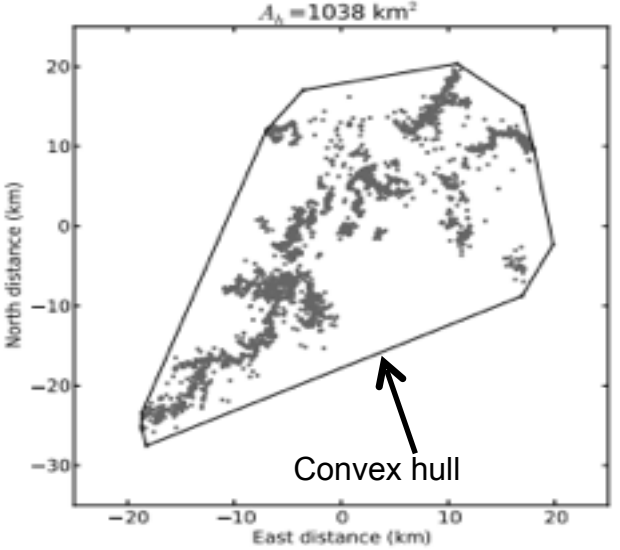
- Standardized ASCII (JSON) polygon log file format

Time series view of flash statistics along tracked polygons

- ASCII CSV tables, time series plots

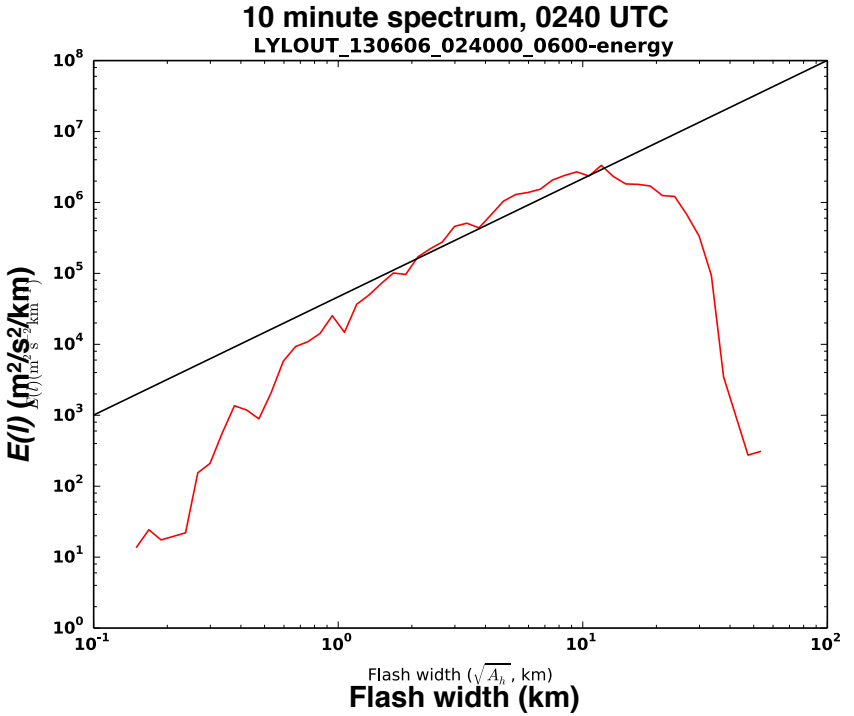
Flash detection efficiency simulation tools

Bruning and MacGorman (2013)

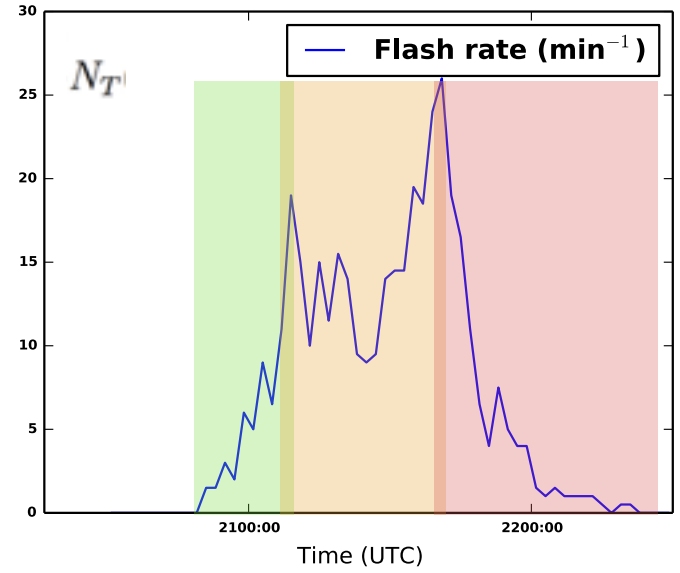


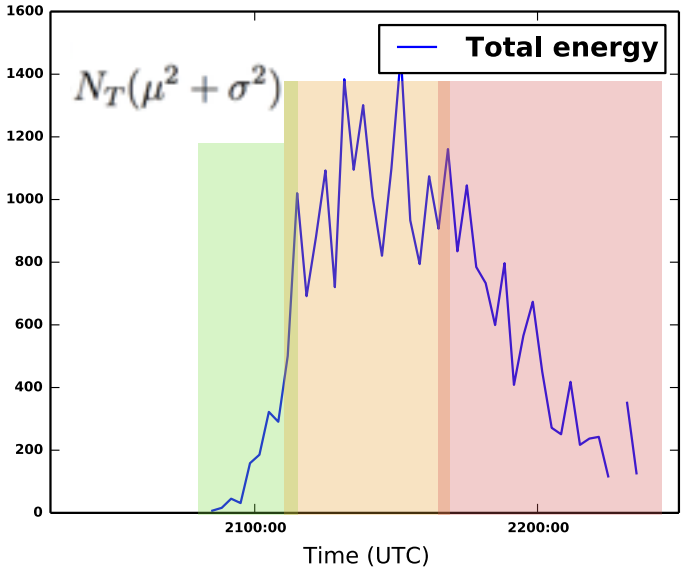
- Not all flashes are the same
- Flash sizes are distributed in a predictable way. Scaled energetically, the distribution looks like a thunderstorm's turbulent kinetic energy spectrum. (Bruning and MacGorman, 2013)
  - Peak at ~10 km scale
  - 5/3 power-law in the inertial range at < 5 km
- Suggests that eddy-scale flow is organizing charge

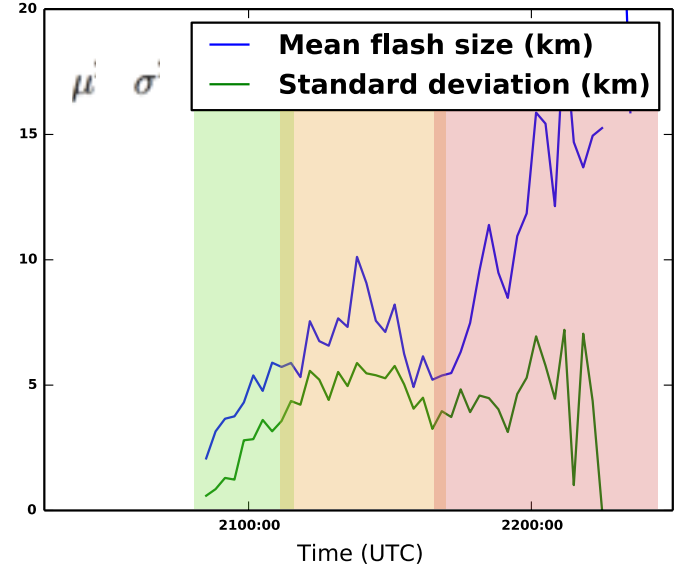
$$l = \sqrt{A_h} \qquad E(l) = \frac{\rho^2 l^2 d^3}{2\epsilon_0} = Kl^2$$

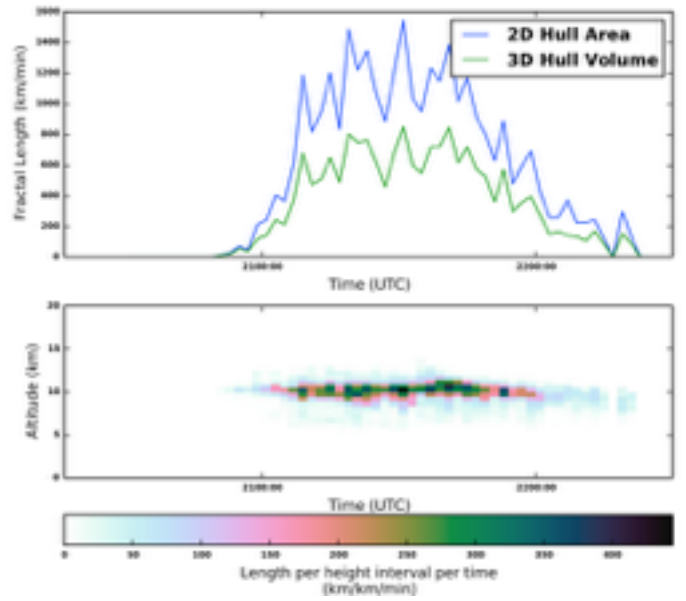


22 June 2012 - DC3 IOP 21a - isolated cell in Colorado
Moments of the flash size distribution (Bruning and Thomas 2015)









Establishment

- Flash rates and total energy increase
- Small flashes
- Dominated by active convection

Modulation

- Average flash size fluctuates between 5-10 km wide
- No discernible energy trend as anticorrelated size and rate balance

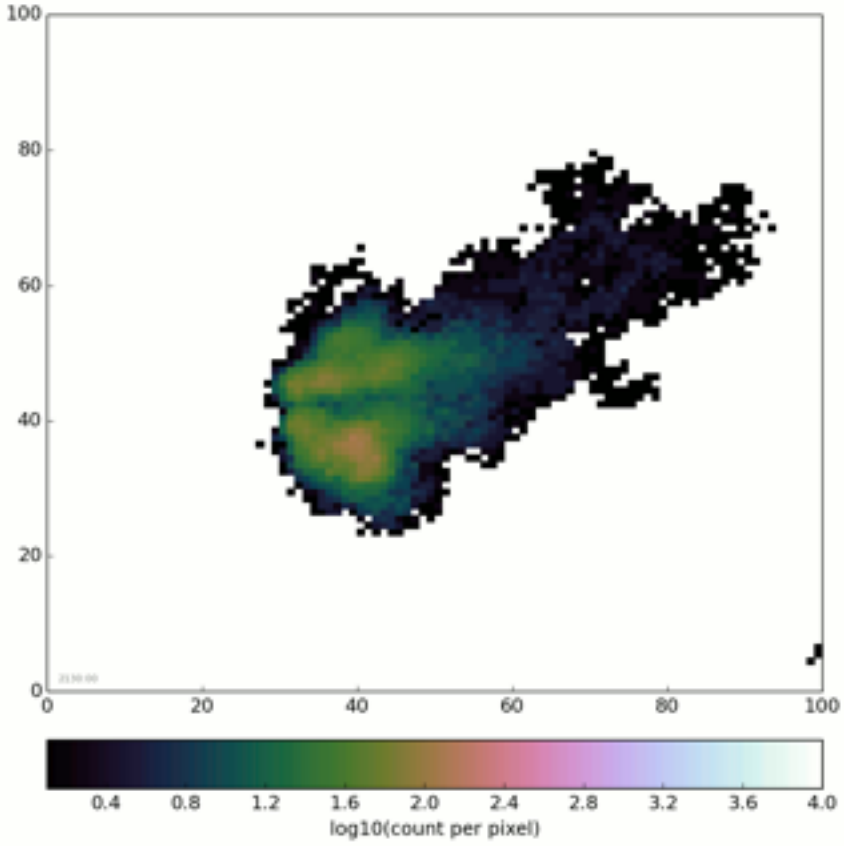
Decay

- Sizes increase
- Rates decrease
- Energy decreases less quickly. Consumes energy stored earlier

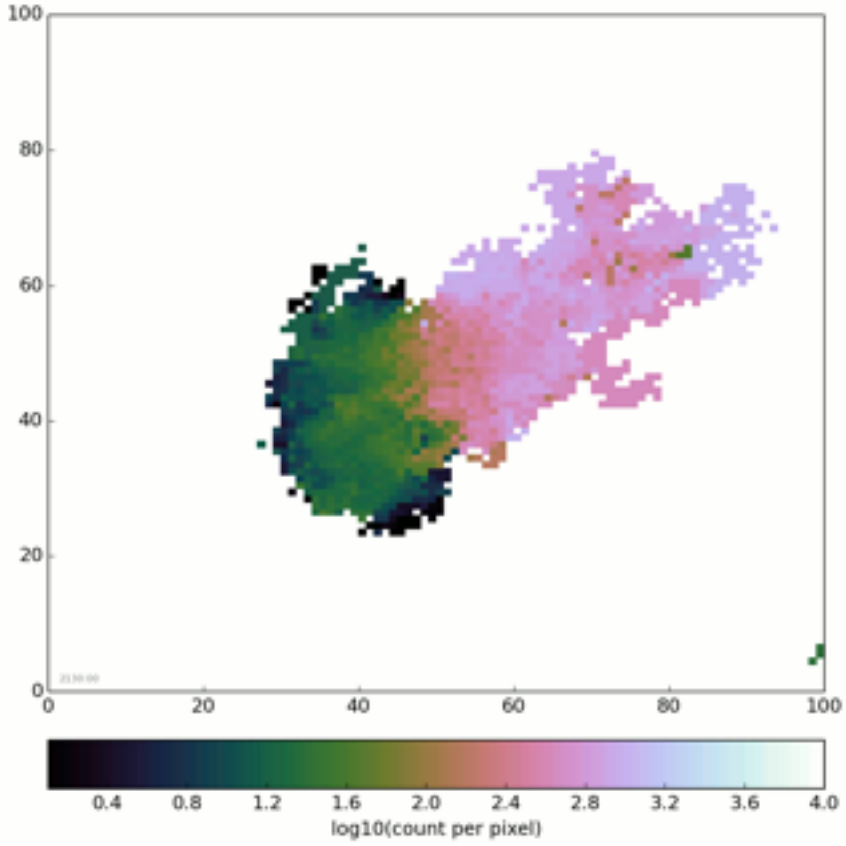
Spatial grids of flash size

07/16/2009 Supercell, OKLMA

Flash Extent Density (Count)



Average Flash Area (Total Area / Count)



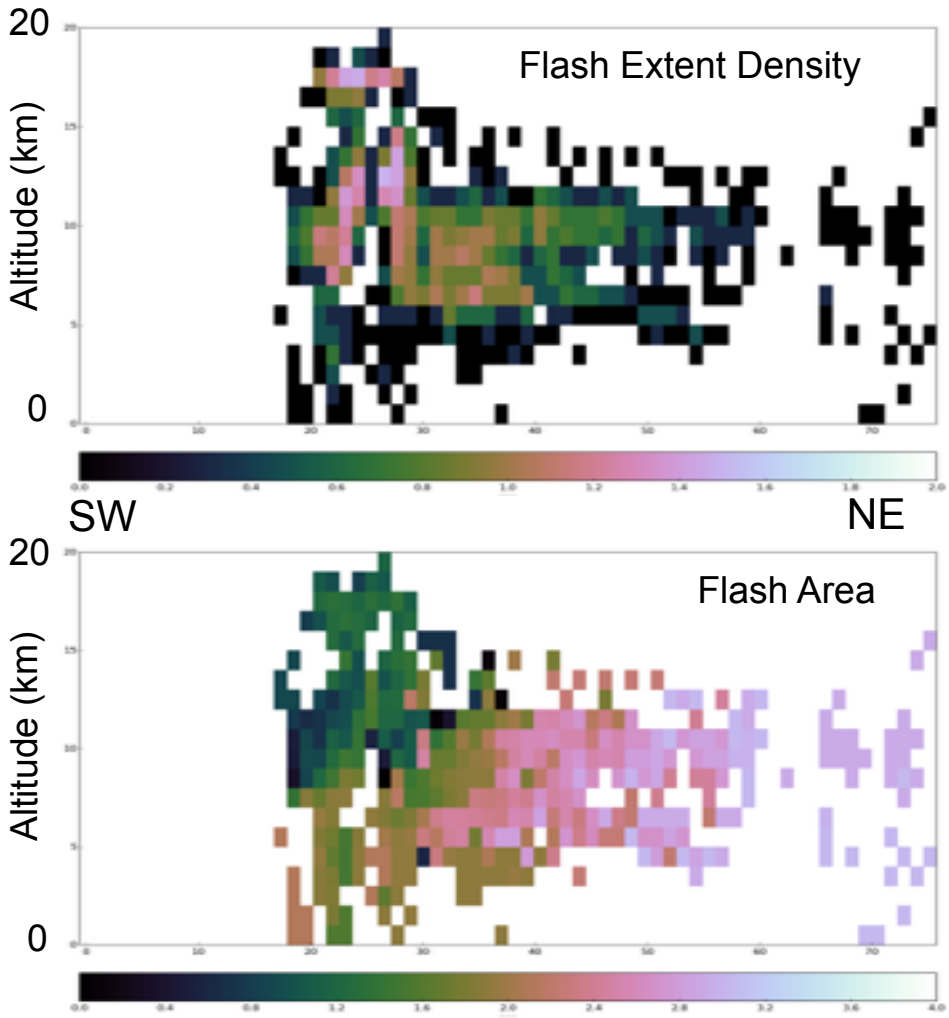
All flash color scales are $\log_{10}$

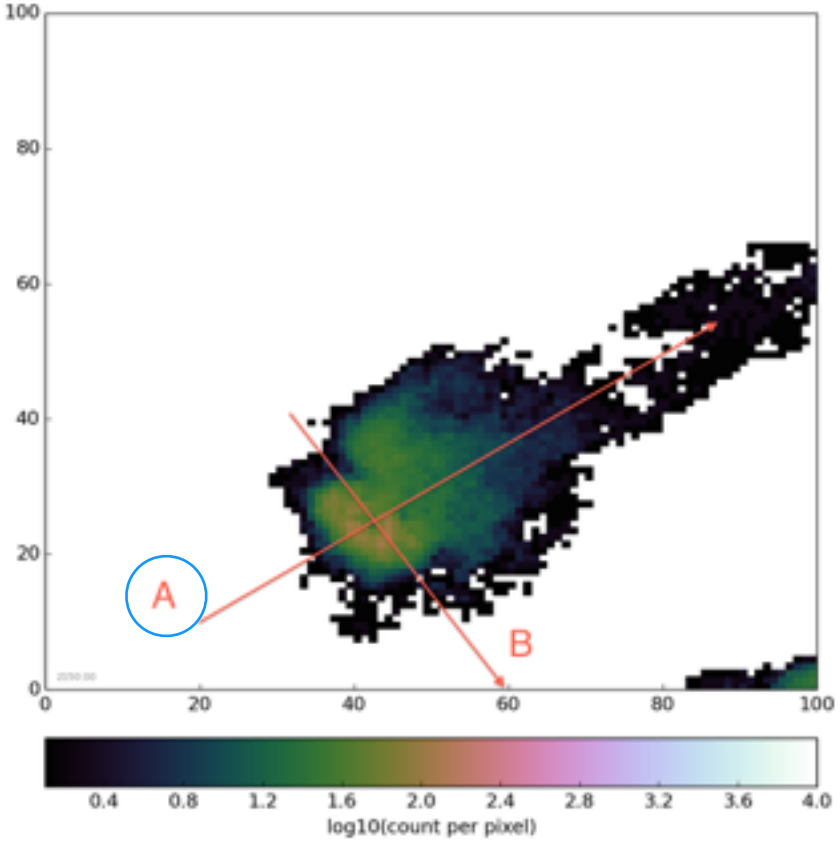
Ware, 2015

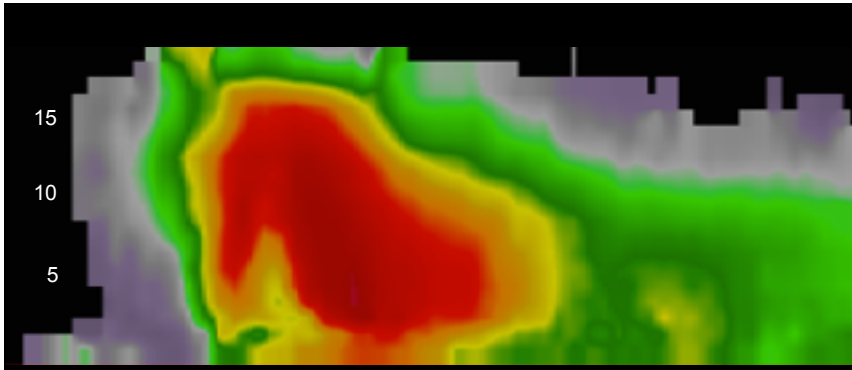
Vertical distribution of flash size

07/16/2009 Supercell

Cross-Section A



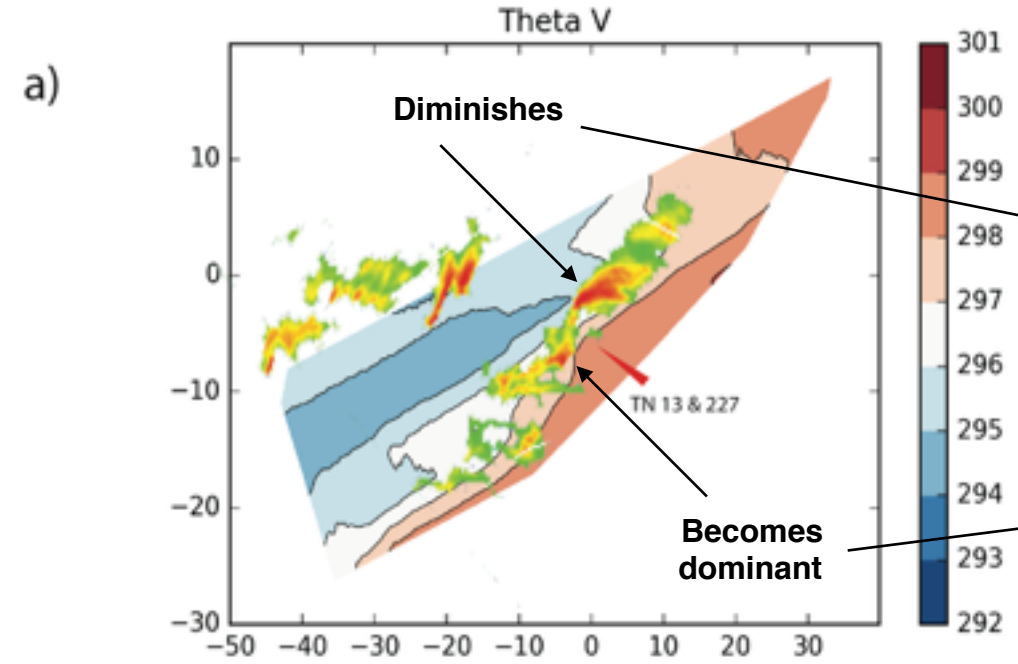


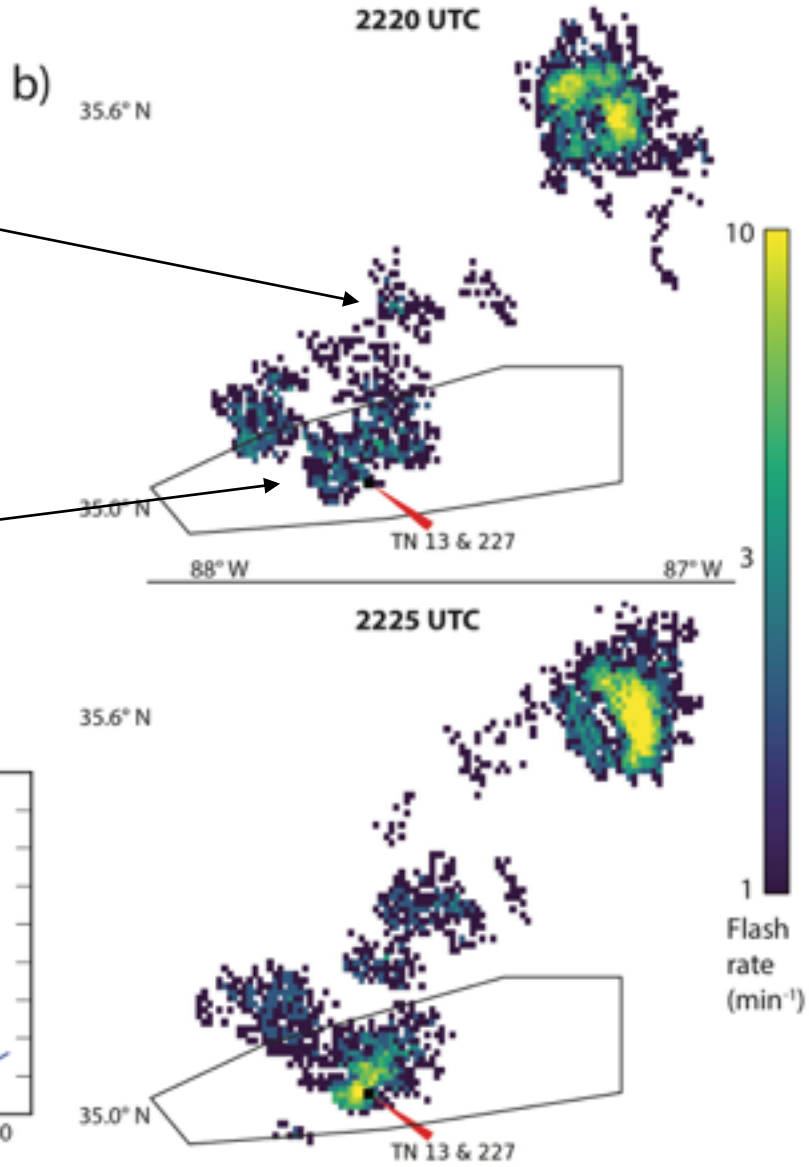


Radar Reflectivity

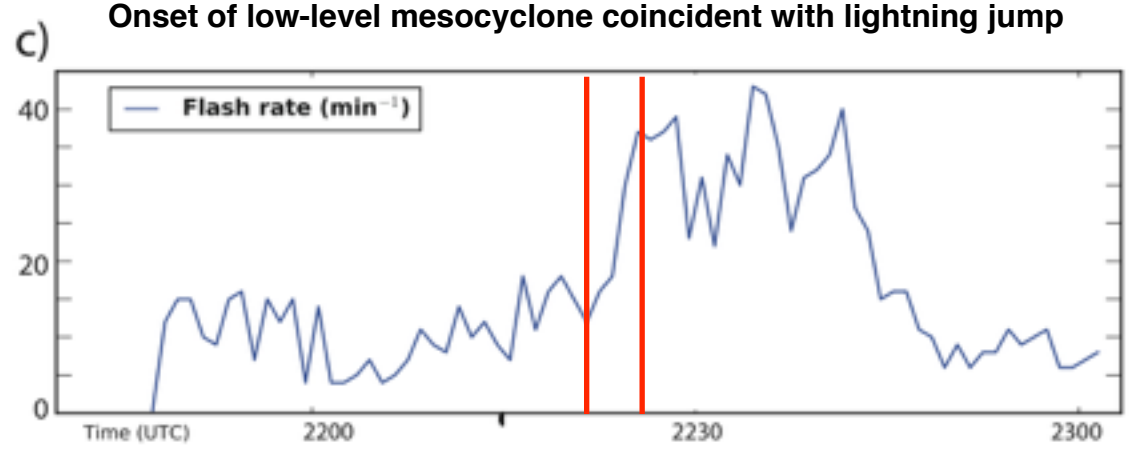
Ware, 2015

31 March 2016, VORTEX-SE





Onset of low-level mesocyclone coincident with lightning jump



Practical use: tools ready to go

Step one

- Sort VHF source data into flashes, create 2D and 3D grids, make plots for each grid type.

Supporting tools

- Interactive grid viewer and polygon creation tool
- Interactive (xlma-like) plotting package (brawl4d)

Step two

- Flash energy spectra, flash moment and energy time series, grid statistics within polygons.

GLMTOOLS?
ACCUMULATED LIGHTNING PRODUCTS FOR GLM

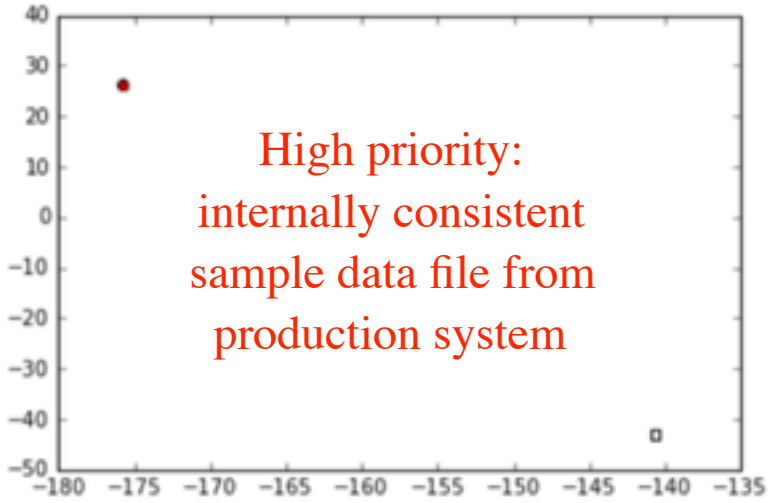
- In lmatools, functions for gridding, time series, polygon subsetting, etc. are dataset agnostic
- GLM has obvious group and flash metadata analogues to LMA flash size and energy: footprint and radiant energy.
- glmtools: create the same kinds of plots and time series statistics and compare to LMA
  - *Group and flash extent density*
  - *Average local energy and footprint (both groups and flashes)*
  - *GLM flash and group moment time series*
- Compare GLM with LMA data inside same polygon
  - *Ratios of time series trends and spatial patterns*

A START TOWARD GLMTOOLS

- Tests with sample data file:
  - *~100 lines of code to interactively plot a flash and all its groups and events given a flash_id*
  - *Built on xarray library*
  - *Data tools supporting hierarchically clustered, event-based data are new territory in meteorology*

Flash 15923

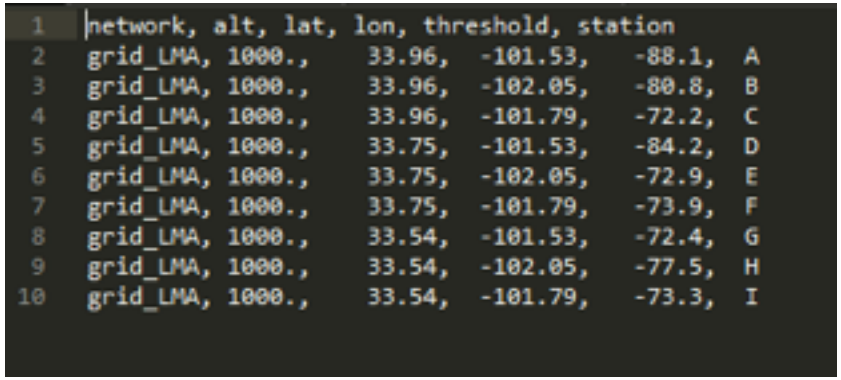
```
<xarray.DataArray 'number_of_events' (number_of_events: 3)>
array([24, 25, 26])
Coordinates:
    event_id                (number_of_events) int32 13521618 135
    event_time_offset       (number_of_events) datetime64[ns] 201
    event_lat               (number_of_events) float64 -42.97 -42
    event_lon               (number_of_events) float64 -140.7 -14
    event_parent_group_id   (number_of_events) int32 13280571 132
    product_time            datetime64[ns] 2015-08-12T21:25:15.26
    lightning_wavelength    float64 777.4
    group_time_threshold    float64 0.0
    flash_time_threshold    float64 3.33
    lat_field_of_view       float64 0.0
    lon_field_of_view       float64 -75.0
  * number_of_events        (number_of_events) int64 24 25 26
```



High priority: internally consistent sample data file from production system

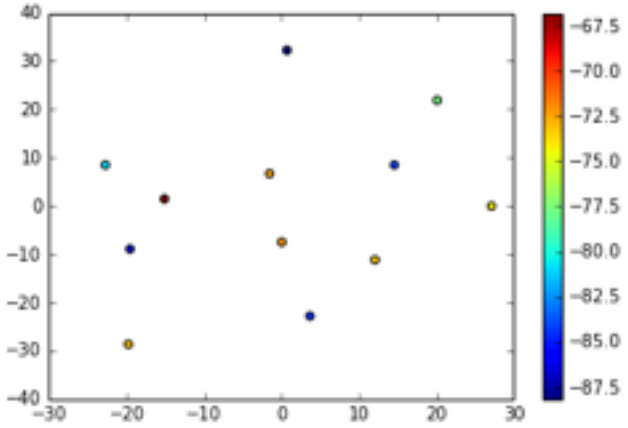
LMA Simulation Tool

- For GLM cross-comparison it is important to understand how LMA performance varies with range from the network
- Python Notebook, open-source
- Models source and flash detection efficiency and distortion of flash areas
- Considers the relative contribution of different stations within a network
  - *Can manually enter station locations and thresholds or take them directly from station log files*

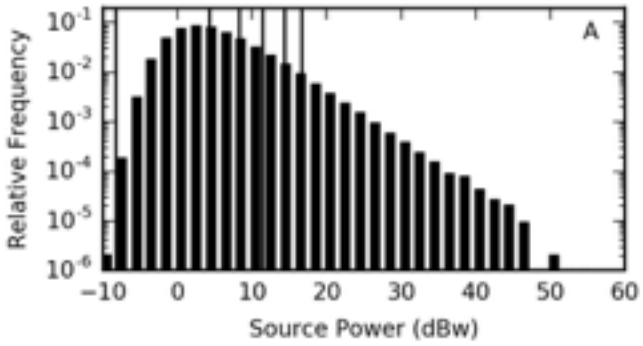


```
network, alt, lat, lon, threshold, station
grid_LMA, 1000.,    33.96,  -101.53,   -88.1,  A
grid_LMA, 1000.,    33.96,  -102.05,   -80.8,  B
grid_LMA, 1000.,    33.96,  -101.79,   -72.2,  C
grid_LMA, 1000.,    33.75,  -101.53,   -84.2,  D
grid_LMA, 1000.,    33.75,  -102.05,   -72.9,  E
grid_LMA, 1000.,    33.75,  -101.79,   -73.9,  F
grid_LMA, 1000.,    33.54,  -101.53,   -72.4,  G
grid_LMA, 1000.,    33.54,  -102.05,   -77.5,  H
grid_LMA, 1000.,    33.54,  -101.79,   -73.3,  I
```

https://github.com/vbalderdash/LMAsimulation



Station thresholds (dBm)



- **Monte-Carlo approach with Marquardt solutions** or Curvature Matrix error predictions
- Propagates sources to each station and adds timing error.
- Uses powers selected randomly from a reference distribution
- Station only contributes to solution if propagated source power above the set threshold for the station
- Reports:
  - Location errors
  - Source and flash detection efficiency (from climatology)

https://github.com/vbalderdash/LMAsimulation

User-defined Inputs

Setting up grid

```
xmin, xmax, xint = -200001, 199999, 5000
ymin, ymax, yint = -200001, 199999, 5000
alts = np.arange(500,20500,500.)
# alts = np.array([7000])
```

1) Locations of minimum and maximum grid points (in m relative to the network center), grid point spacing
2) Number of random sources at each grid point
3) Filtering:
   - Max χ^2
   - Minimum number of contributing solutions
   - Return just RMS error or mean errors and standard deviations

General calculations at grid points

Set number of iterations and solution requirements here

```
In [10]: iterations=100

# # for r,theta,z errors and standard deviations and overall detection efficiency
for i in range(len(x)):
    means[i], stds[i], misses[i] = sf.black_box(points2[i,0], points2[i,1], points2[i,2],
             iterations,
             stations_local,
             ordered_threshs,
             stations_ecef,center_ecef,tanp_all, c0,dt_rms,tanp,projl,
             chi2_filter=5.,
             min_stations=6,
             just_rms=False
             )
```

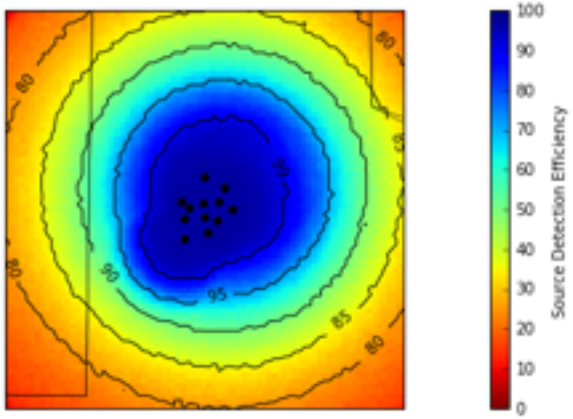
https://github.com/vbalderdash/LMAsimulation

Comparison of Detection Efficiency

```
In [17]: domain = 197.5*1000
maps = Basemap(projection='laea', lat_0=center[0], lon_0=center[1], width=domain*2, height=domain*2)
s = plt.pcolormesh(np.arange(xmin-xint/2.,xmax+3*xint/2.,xint)+domain,
                   np.arange(ymin-yint/2.,ymax+3*yint/2.,yint)+domain,
                   100-np.mean(misses[0,:,:,:], axis=2)*100./iterations,
                  cmap = 'jet_r')
s.set_clim(vmin=0,vmax=100)
plt.colorbar(label='Source Detection Efficiency')
CS = plt.contour(np.arange(xmin,xmax+xint,xint)+domain,
                 np.arange(ymin,ymax+yint,yint)+domain,
                 fde_a, colors='k',levels=(80,85,90,95,99))
plt.clabel(CS, inline=1, fontsize=10,fmt='%3.0f')
plt.scatter(stations_local[:,0]+domain, stations_local[:,1]+domain, color='k')

maps.drawstates()
plt.tight_layout()
plt.show()
```

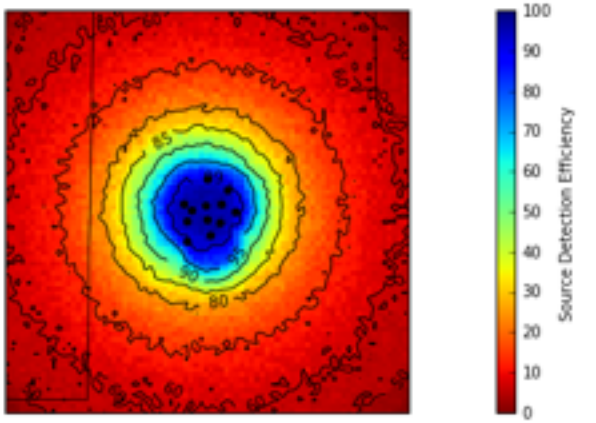
5-day average from Oct 2014



Fewer stations contributing (high thresholds, missing stations)

→

4 June 2012



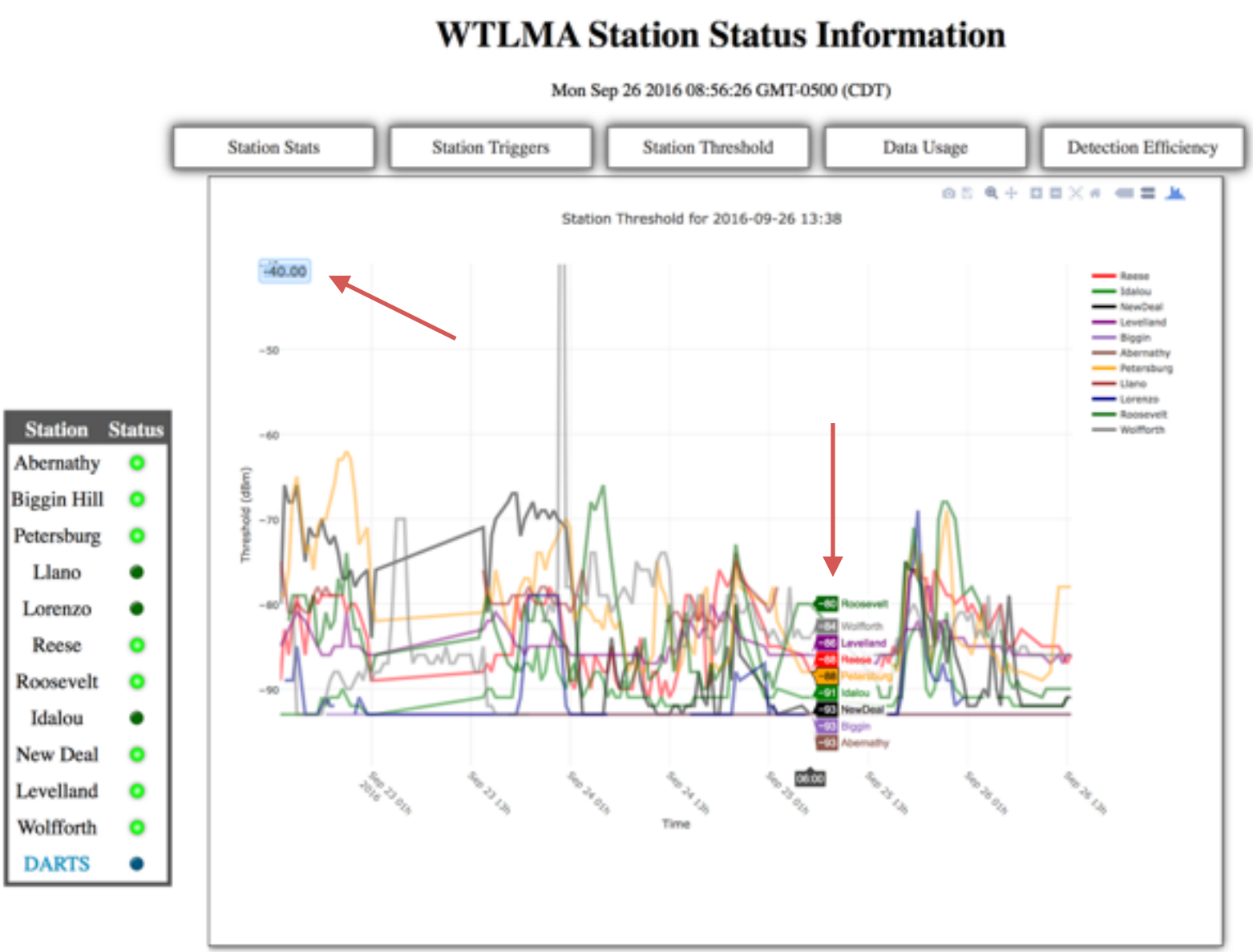
WTLMA Plots

Color: Source detection efficiency

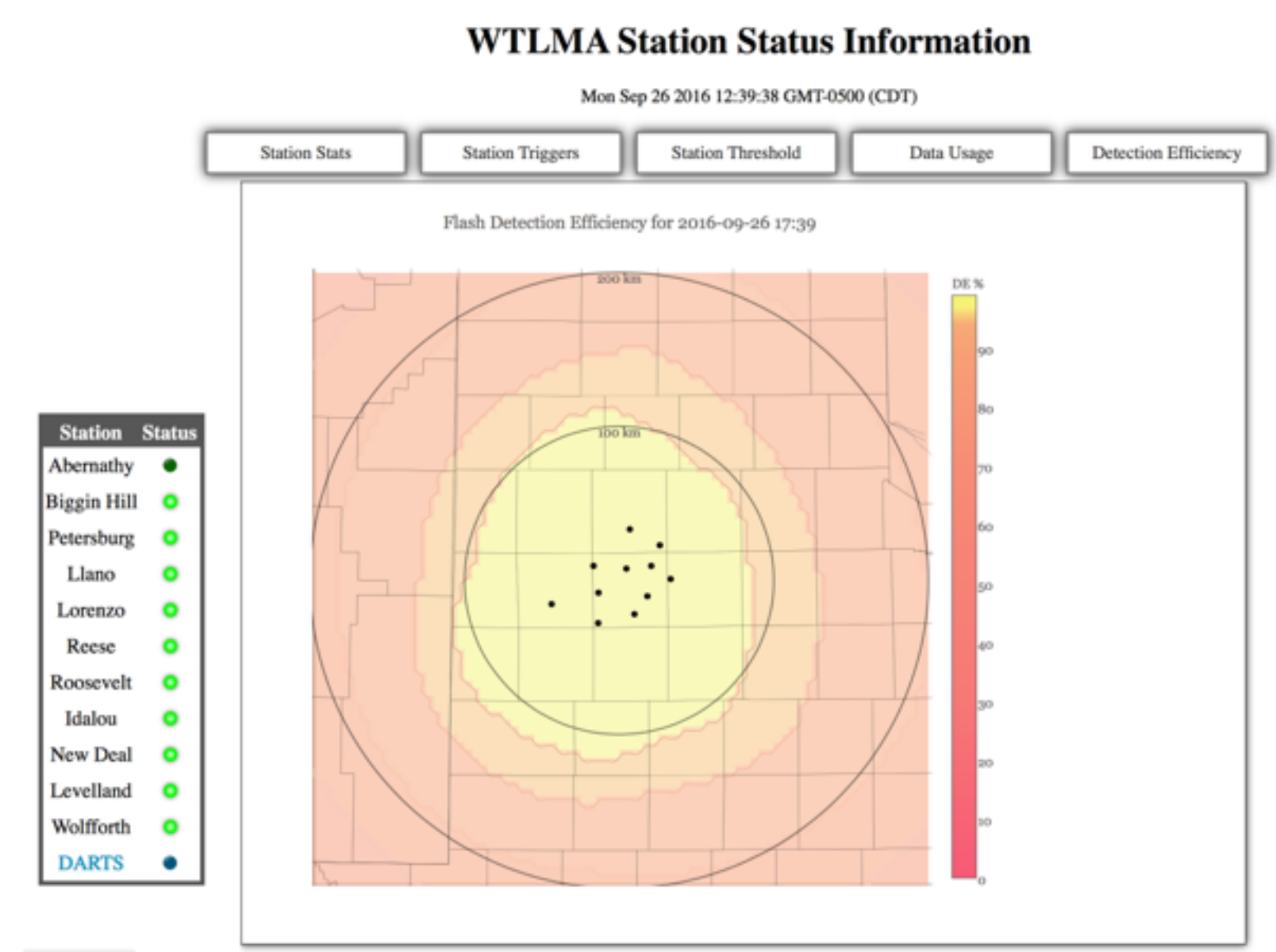
Contours: Flash detection efficiency

https://github.com/vbalderdash/LMAsimulation

Real-time LMA performance monitoring



REAL-TIME LMA PERFORMANCE MONITORING



SUMMARY

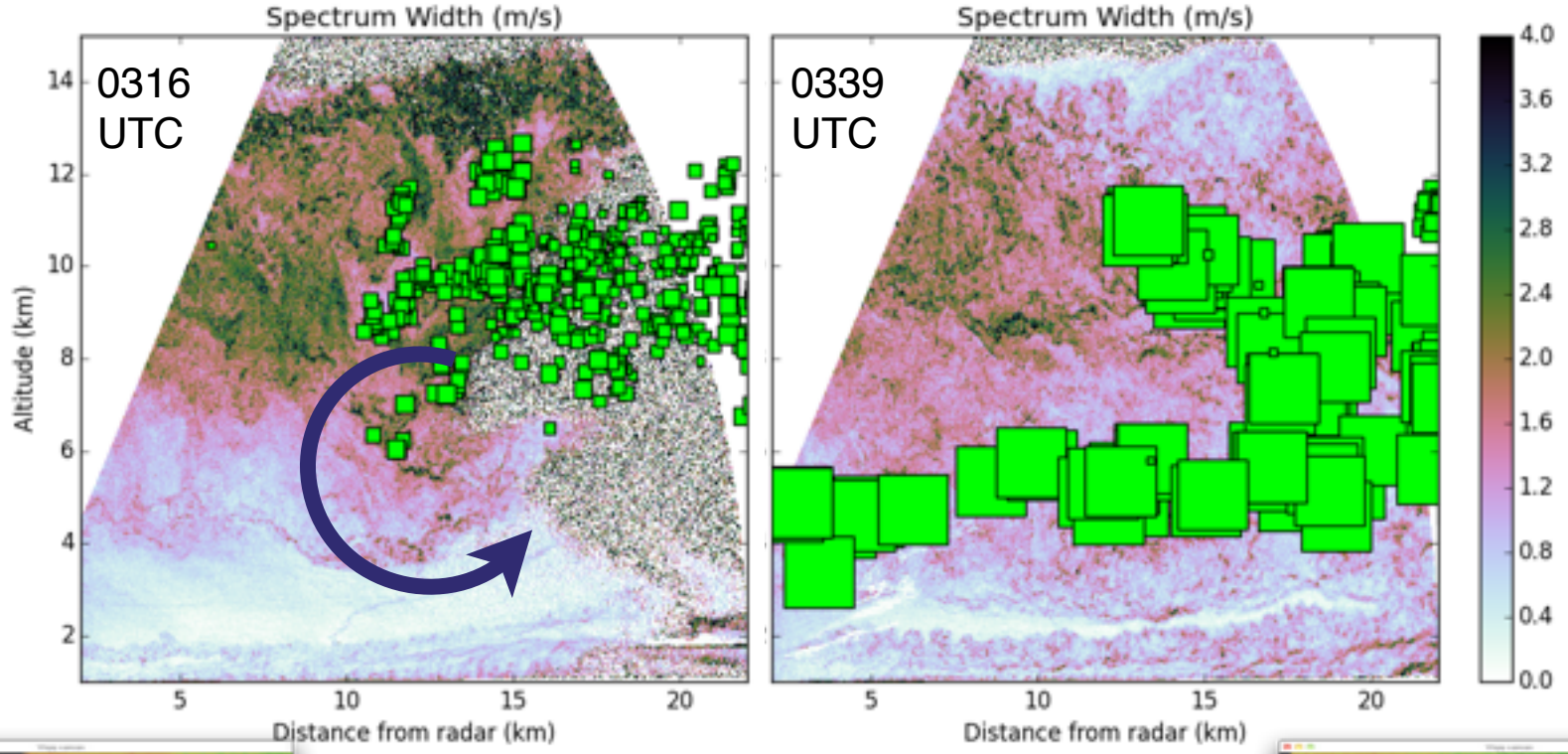
lmatools is a mature library encompassing a variety of peer-reviewed data processing and instrument characterization techniques focused on the *spatial* and *temporal* distribution of meteorologically interesting variability in lightning.

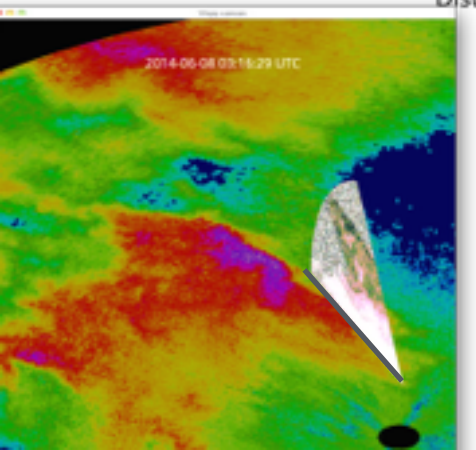
Application of these techniques to GLM and routine cross-comparison with LMA data is our planned starting point for deep-dive analysis.

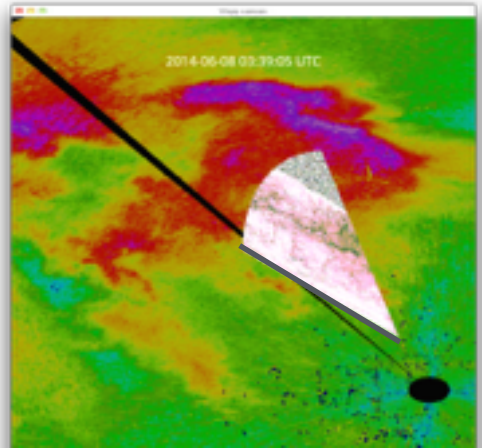
The tools can operate in real-time or on post-processed data, and interactively or in batch mode.

The tools support low-level data access for users who want to compare to non-lightning datasets via other meteorological tools in the Python ecosystem. The tools are open source and contributions are welcome.

Fewer, larger flashes in less-turbulent regions







- All LMA sources from flashes within 10° of RHI
- Size of symbol = area of flash to which source belongs