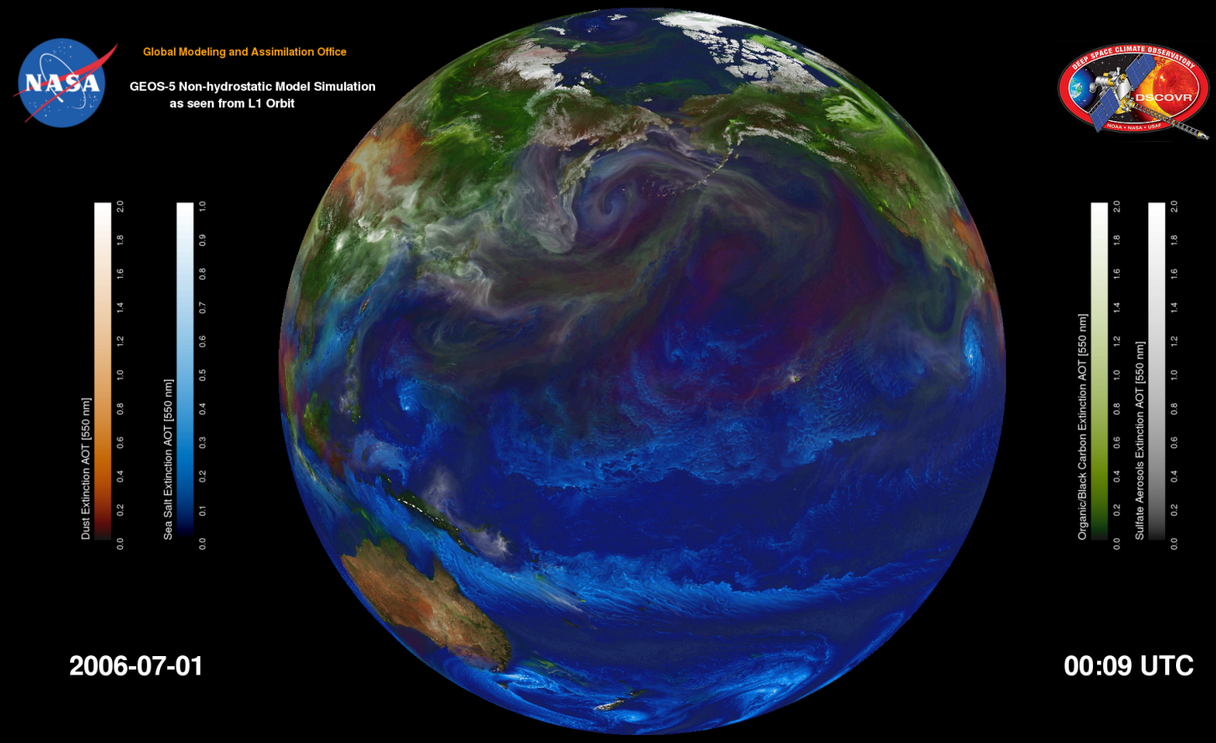
TRADITIONAL DATA-DRIVEN SCIENTIFIC DISCOVERY METHODS DO NOT SCALE TO LARGE DATASETS

- 7km GEOS-5 “Nature Run”
- 1 dataset, 3.5 PB
- theoretically: openly accessible
- practically: precomputed pics



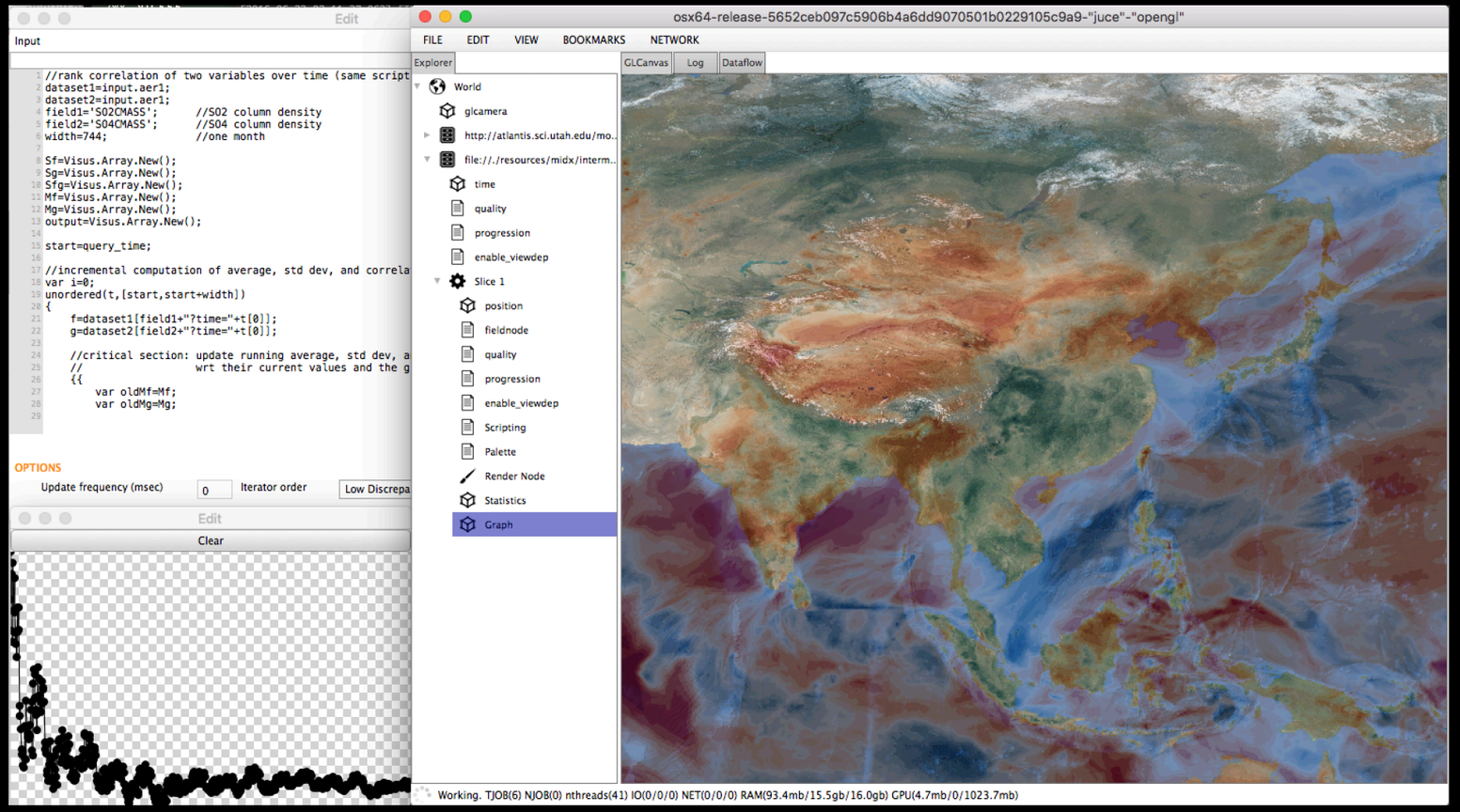
COMMON WORKFLOW FOR SCIENTIFIC DATA ANALYSIS

1. Data Management: Acquisition, Conversion, and Regridding
2. Computation and Analysis
3. Visualization / Comparison

SPECIFIC TECHNICAL CHALLENGES THAT PREVENT INTERACTIVE SCALING OF DATA-DRIVEN DISCOVERY TO LARGE MODELS

- File formats unsuitable for streaming
- Batch mode data analyses
  - system: submit job and wait
  - algorithm: only final results
- Programming models not progressive
- Server-side analysis does not scale ***to large communities***

WE ADDRESS THE CHALLENGE OF INTERACTIVE EXPLORATION AND ANALYSIS OF MULTI-PETABYTE DATASETS WITHOUT MASSIVE HPC RESOURCES

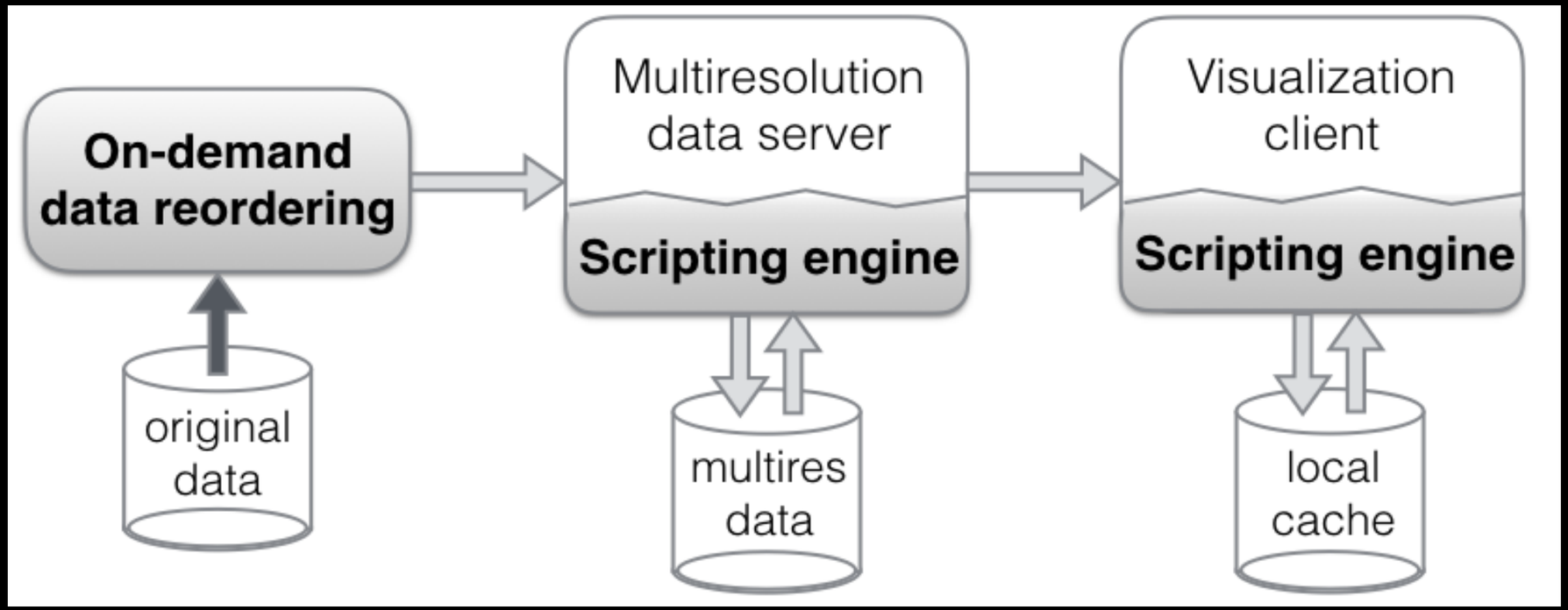


INTERACTIVE MULTIRESOLUTION EXPLORATION OF MASSIVE REMOTE DATASETS

(Please show multires_nature video now)

METHOD OVERVIEW

- **Generic EDSL** scripting
- **Progressive Runtime** environment
- **On-demand** data reordering

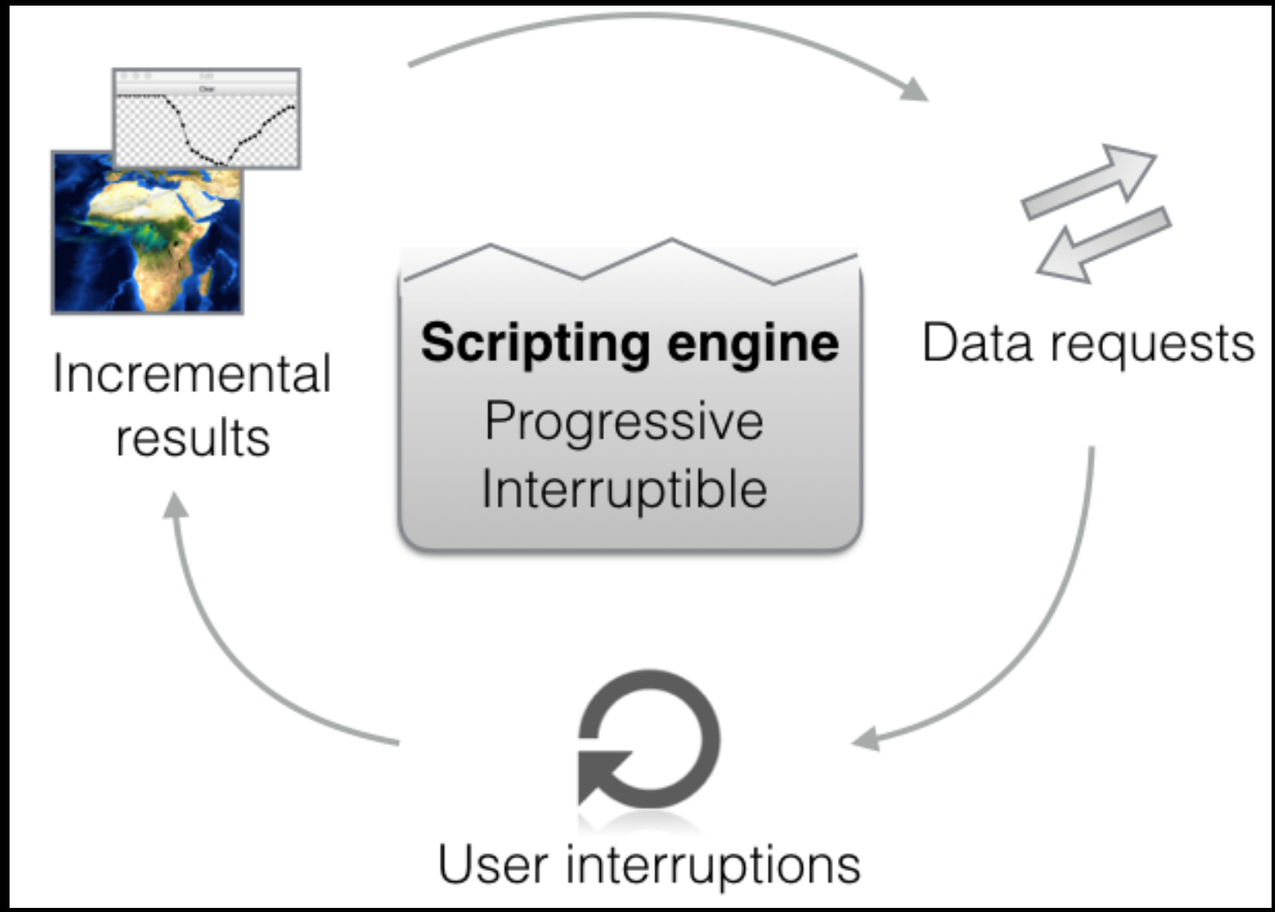


EMBEDDED DOMAIN SPECIFIC LANGUAGE AND RUNTIME FOR PROGRESSIVE COMPUTATION

- Incremental computation results
  - EDSL supports…
    - abstract data type (location, resolution, format)
    - unordered loops
    - incremental publishing
  - renders current measure of data access irrelevant

PROGRESSIVE RUNTIME FOR INCREMENTAL SCRIPT EXECUTION

- incremental results
- resolution level
- loop order and parallelization
- server-side processing



WHY AN EDSL?
THE IMPORTANCE OF GENERICITY IN ANALYSIS SCRIPTS

- Genericity facilitates runtime utilization of…
  - incremental execution
  - data format advantages
  - superior loop ordering
  - remote processing
  - parallelization

EMBEDDED DSL FOR INCREMENTAL COMPUTING

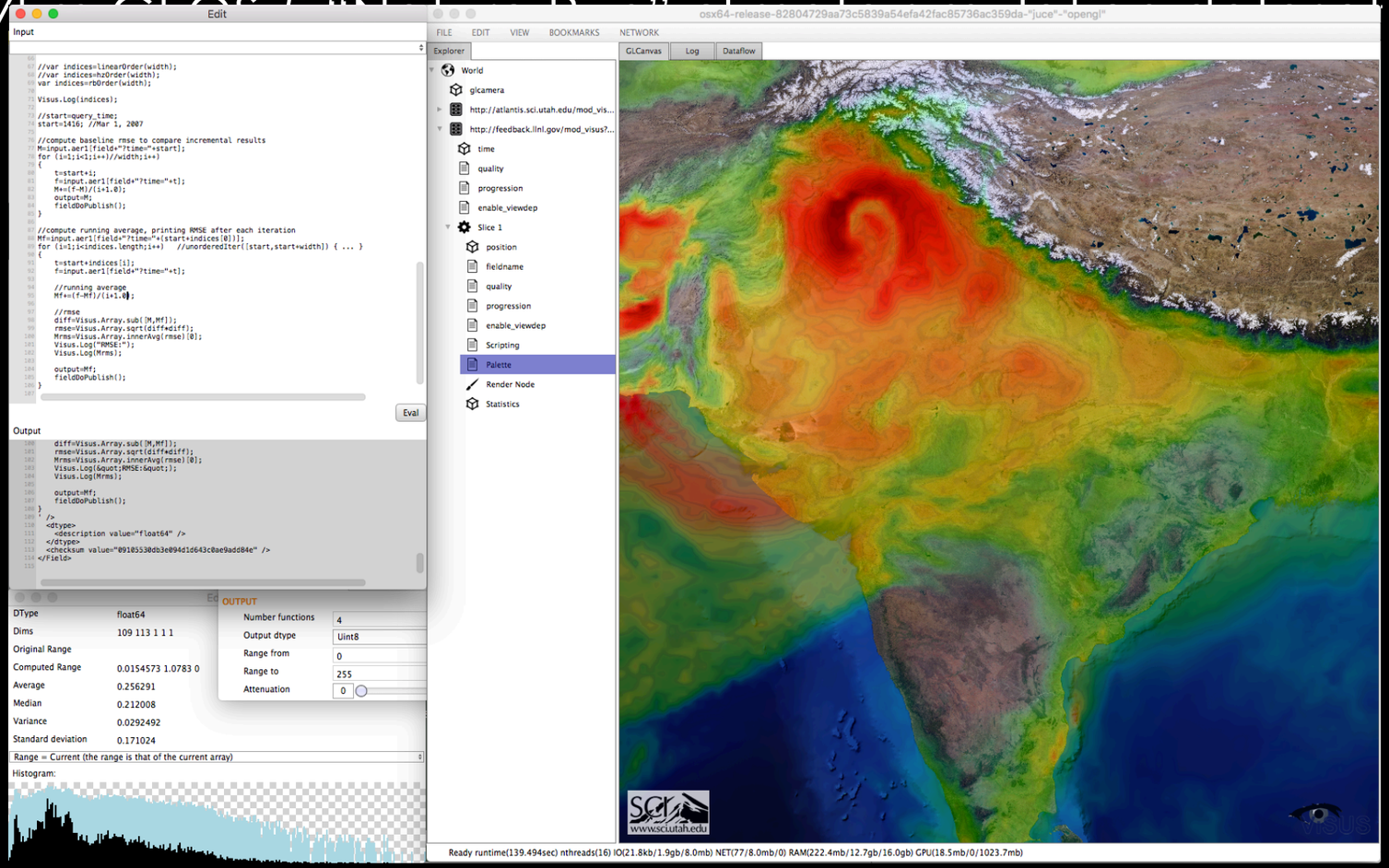
- Minimal extensions to host language
  - ***doPublish, scientific data type, generic loops***
- Example operations of our data type:
    - elementwise combinations (add, subtract, multiply, divide, …)
    - statistical calculation (average, standard deviation, range, median, …)
    - domain selection (crop, paste, resize, interleave, …)

EDSL SCRIPT FOR INCREMENTAL COMPUTATION OF AVERAGE

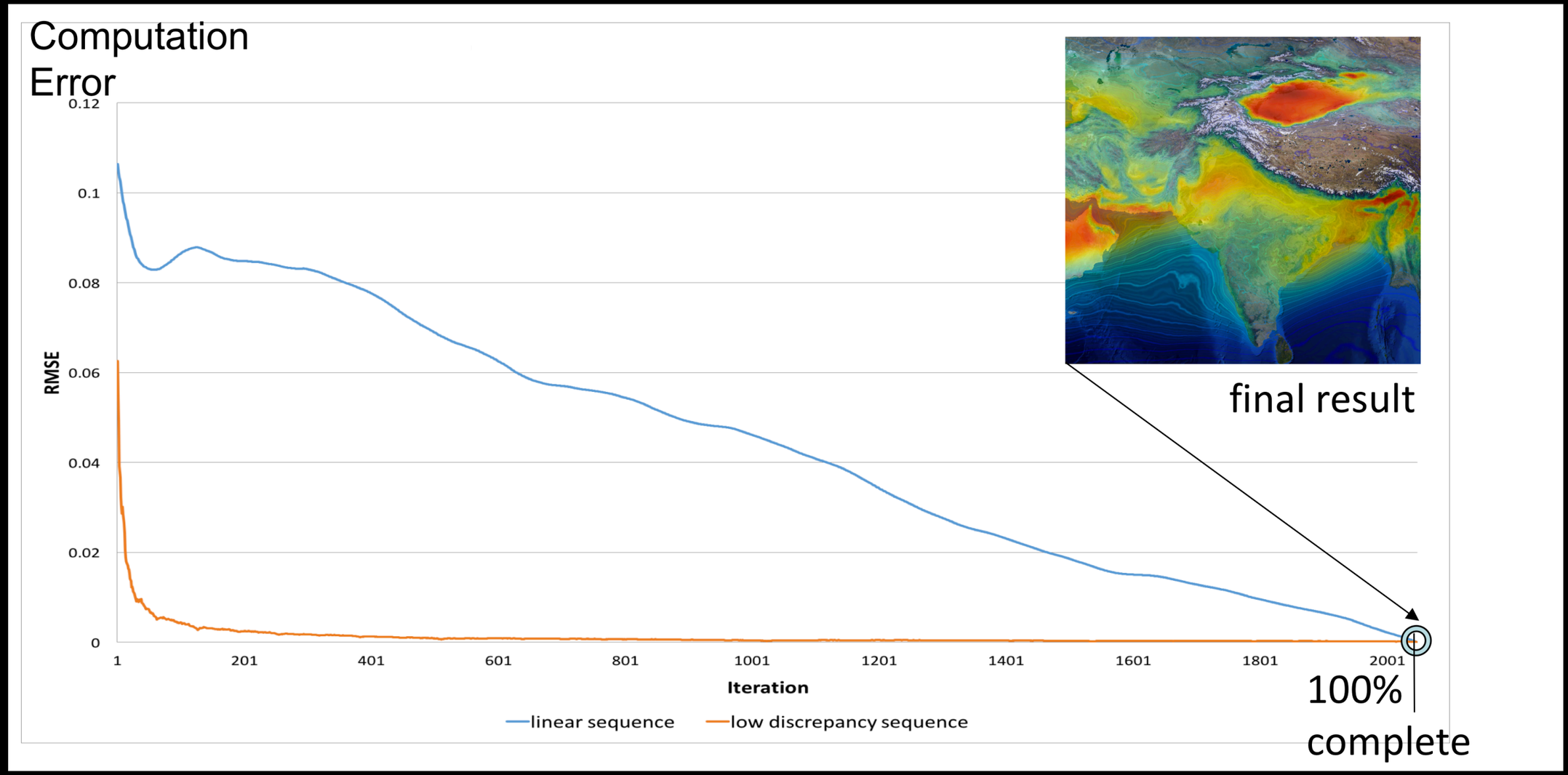
```
var output = Array.new();
var i = 0;
unordered(time, [start, end])          // generic loop
{
  var field = Array.read('fieldname', time);
  // critical section (output and i must be updated atomically)
  {{
    output += (field - output) / (i + 1);   // update incremental average
    i++;
  }}
  doPublish();              // make intermediate result available
}
```

EFFECT OF LOOP ORDERING AND STREAMING DATA FORMAT FOR PETA-SCALE CLIMATE DATA ANALYSIS AND VISUALIZATION

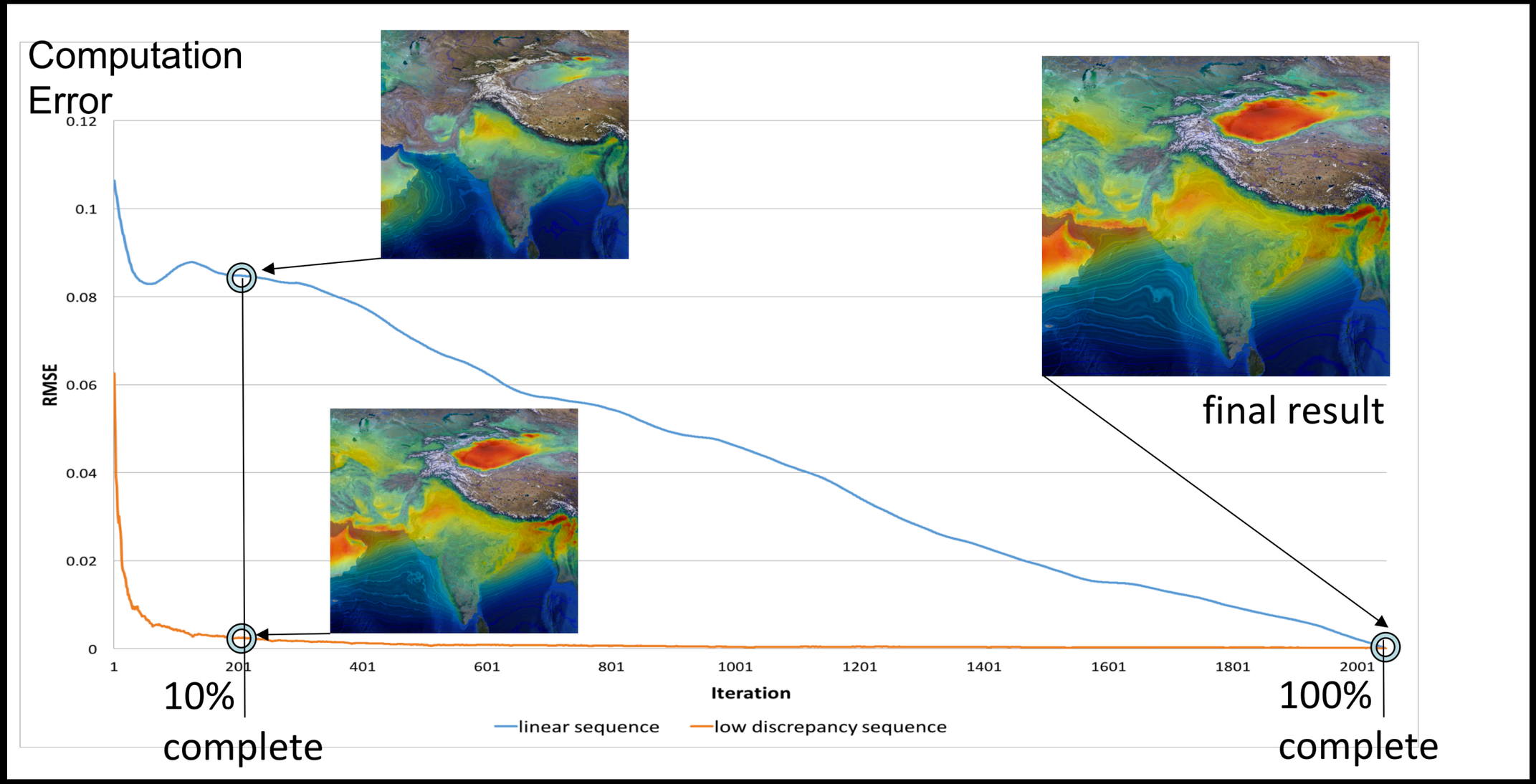
- **3.5 petabyte** 7[illegible]



PROGRAMMING MODEL AND RUNTIME SYSTEM ALLOW ALTERNATIVE DATA ORGANIZATION AND PROGRESSIVE COMPUTATIONS



PROGRAMMING MODEL AND RUNTIME SYSTEM ALLOW ALTERNATIVE DATA ORGANIZATION AND PROGRESSIVE COMPUTATIONS



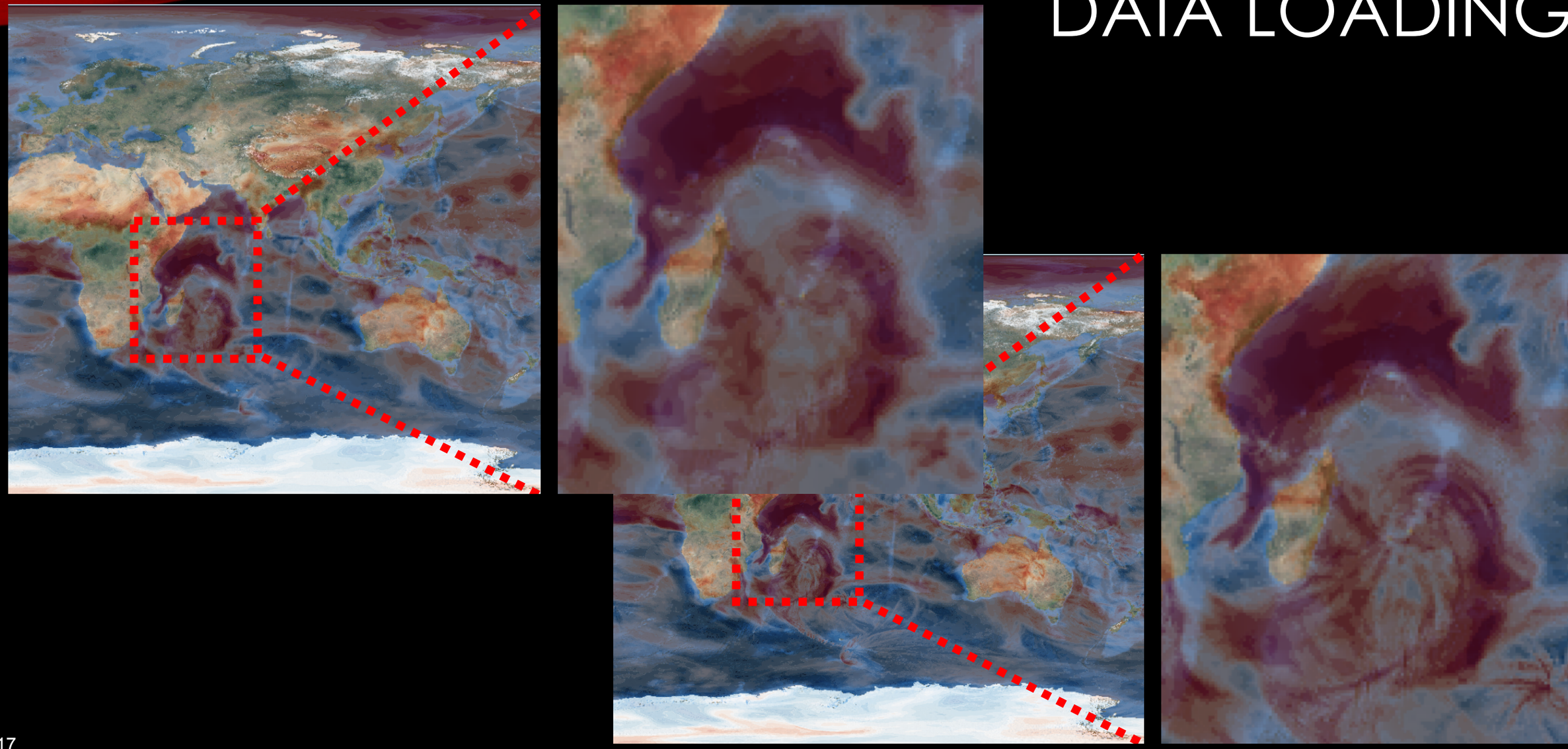
COMPARISON OF DIFFERENT LOOP ORDERINGS

(Please show order comparison video now)

ON-DEMAND DATA REORDERING FOR INTERACTIVE VISUALIZATION AND ANALYSIS OF MASSIVE, DISPARATELY LOCATED DATA

- Interactive access to massive data
  - multiresolution data layouts (IDX)
  - on-demand conversion (operational at LLNL)

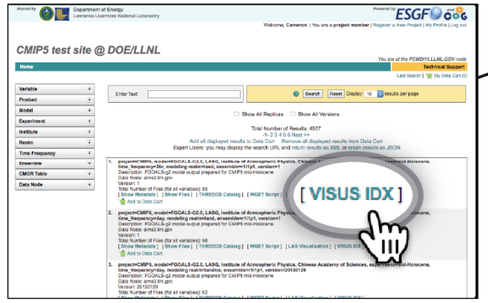
INCREMENTAL MULTIRESOLUTION DATA LOADING



ON-DEMAND DATA REORDERING FOR OPTIMAL ACCESS OF MASSIVE SPATIOTEMPORAL DATA

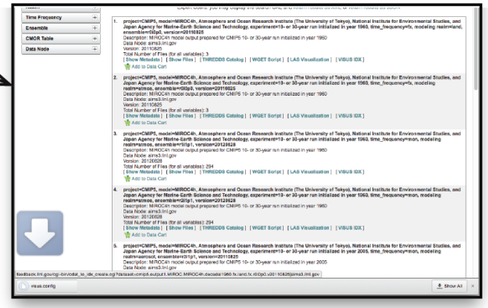
1.

ESG Search



Metadata Service

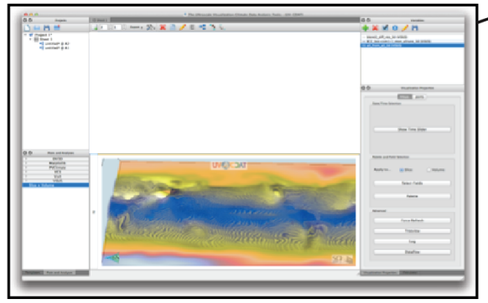
IDX Data Server



Download streaming metadata. Creates and registers a new volume with data server, but does not convert anything.

2.

Client Viewer



IDX Data Server

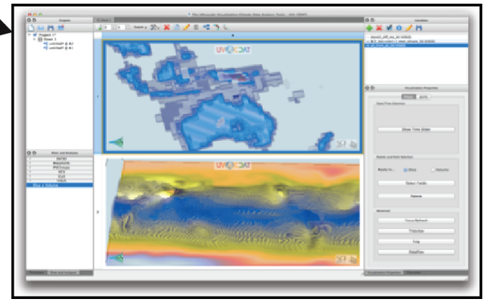
i.

ii.

IDX cache

Climate Data Converter

[climate data]

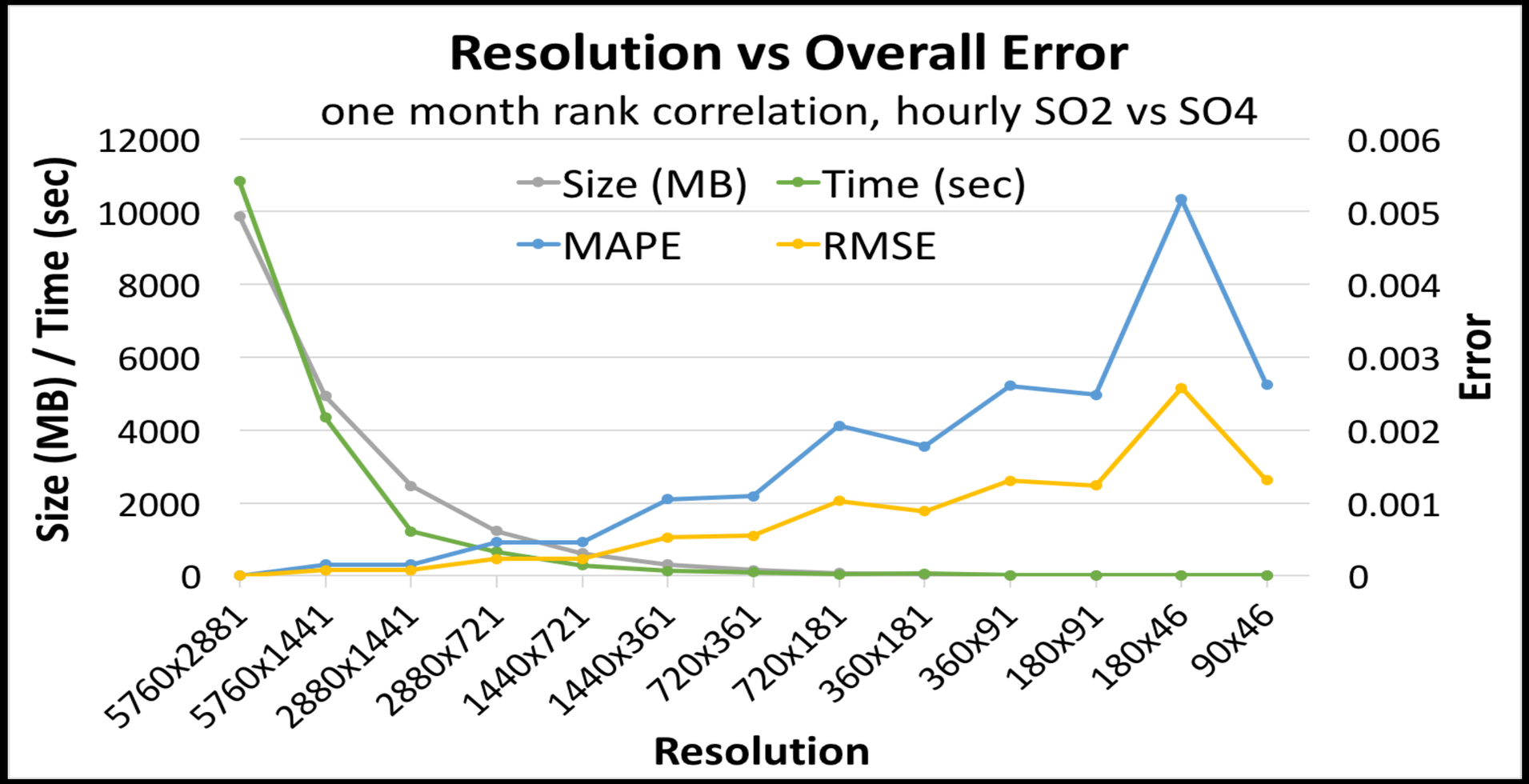


Client requests data, which is converted on-the-fly to idx multiresolution streaming format and cached for later use.

i. Data server first checks idx cache.

ii. If not cached, requested data is converted can now be loaded from cache.

EFFECT OF RESOLUTION ON SPEED AND ACCURACY FOR A LARGE COMPUTATION



DISPARATELY LOCATED DATA CAN BE PROCESSED IN MULTIPLE LOCATIONS AT MOST SUITABLE RESOLUTION

- Eases multi-ensemble analyses
  - automatic regridding (user has control)
  - server-side distributed computation

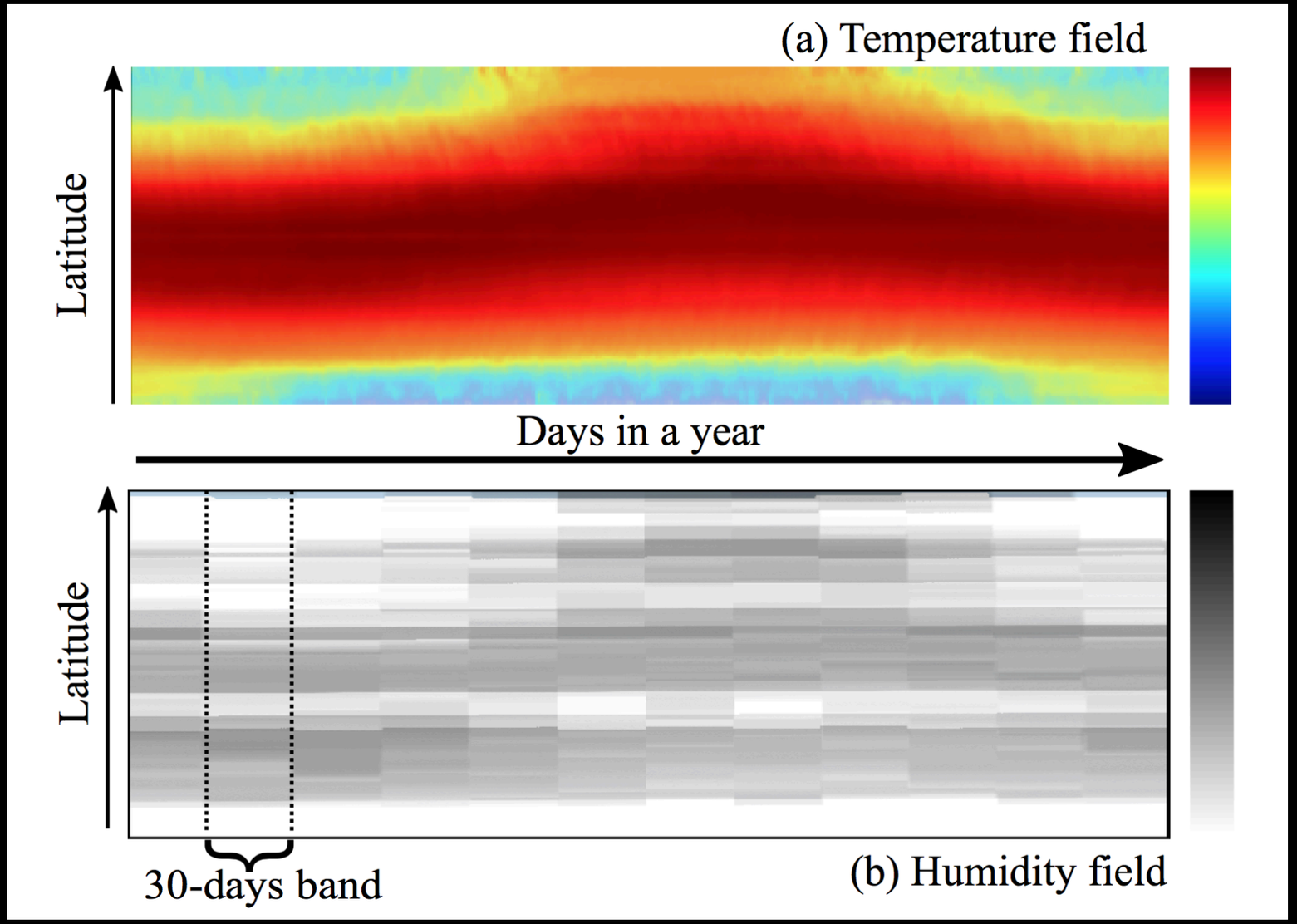
RUNTIME: SERVER-SIDE PROCESSING

- Server-side processing:
  - Identical scripting engine as client
  - Remote computation can dramatically reduce data transfer
  - Local vs remote computation specified per script

USING INTERACTIVE ANALYSIS FOR DATASET ERROR DISCOVERY

correct zonal average

error: every 30 days
duplicate first day!



CONCLUSION

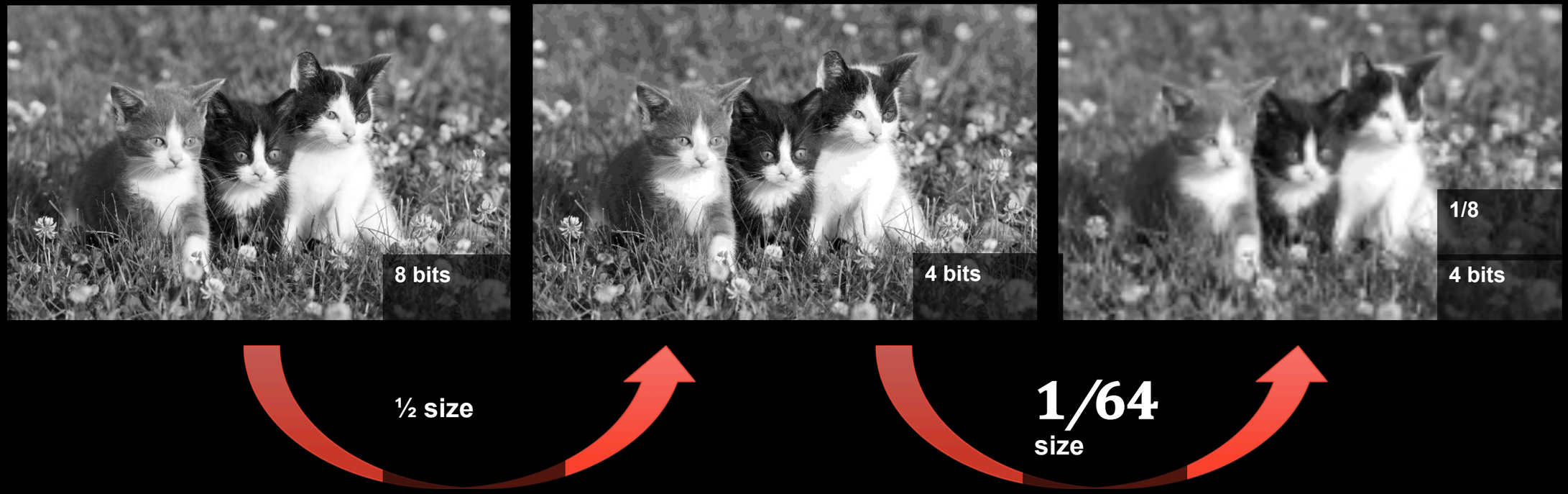
- EDSL + Runtime for **interactive**, **peta-scale** data exploration
- **Incremental results** of dataflow execution
- **On-demand reordering** for optimal data access

WHERE WE'RE GOING FROM HERE

- Compression of multiresolution data (w/ Peter Lindstrom, LLNL)
  - exploring bit-level precision (e.g., specify 2-bits per item)
- Use abstract computation graph for distribution of computation
- Automatically determine resolution level, loop order, remote, ...
- Web interface
- Docker deployment

FLEXIBLE COMPRESSION CAPABILITIES: COMBINING DATA REDUCTION IN BOTH RESOLUTION AND PRECISION

- Achieving better compression than each dimension alone



INTEGRATION WITH ESGF AND EXISTING ANALYSIS TOOLS

- 2013: Using Python SWIG wrappers, UV-CDAT/ViSUS integration
- 2014: Automatic regridding; scripting system is created
- 2015: On-demand conversion of ESGF-hosted data to IDX
- 2016: EDSL formalized; server-side scripting added
- 2017: Integrate CDMS2 module level to enable first class treatment of IDX datasets
- 2017: Multinode incremental server-side EDSL execution
- 2017: Web interface for EDSL-based data analysis and vis

THANK YOU

- Co-authors: Shusen Liu, Giorgio Scorzelli, Ji-Woo Lee, Peer-Timo Bremer, Valerio Pascucci
- Collaborators: Dean Williams, Sasha Ames, Anthony Hoang, Sam Fries at LLNL
- Funding: DOE NNSA, DOE DREAM, NSF, CCMSC, PIPER, ESGF, LLNL AIMS
- Support and Suggestions: Duong Hoang, Sidharth Kumar, Brian Summa, Amy Gooch, Vidhi Zala

EMBEDDED DSL AND RUNTIME FOR PROGRESSIVE SPATIOTEMPORAL DATA ANALYSIS AND VIS

- Focus on data analysis, not data management
- Incremental results, interruptible execution, interactive exploration
- Transparent handling of massive, disparately-located data

```
//Computes running average
field = 'TOTSCATAU';                    //aerosol scattering
start = query_time;                     //current time
width = 720;                            //720 hours (30 days)

output=Array.New();                     //initialize output
var i=0;
unordered(t,[start,start+width])  //Id iterator, index t
{
  f=input[field+"?time="+t];            //read field at time t

  //critical section for running average:
  //average and count must be updated atomically
  {{
    output += (f-output)/(i+1);         //Welford's method
    i++;
  }}

  doPublish();                          //show current result
}
```