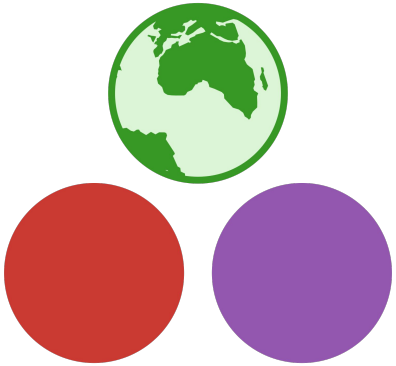


# JuliaGeo updates

Anshul Singhvi
and the JuliaGeo contributors

JuliaEO
Global Workshop on Earth Observation with Julia 2026

# What is JuliaGeo?

- Umbrella organization for geospatial infrastructure in Julia
- I/O, geometry processing, raster processing, etc.
- Goal is for the whole ecosystem to be compatible and composable

**Notable packages**:

- GeoInterface.jl
- GeometryOps.jl
- Rasters.jl
- NCDatasets.jl
- GeoDataFrames.jl
- Shapfile.jl, GeoJSON.jl, GeoArrow.jl, etc.
- GeoMakie.jl, Tyler.jl
- ArchGDAL.jl, LibGEOS.jl, Proj.jl

# The story of JuliaGeo

- Started out as a collection of I/O tools - GDAL, Shapefile, NetCDF, etc.
- Branched out into wrapping more geospatial libraries - LibGEOS, Proj, etc.
- Then into pure Julia algorithms in GeometryOps, Geodesy and friends.
- Now an ecosystem of interoperable packages:
  - I/O packages (too many to name here!)
  - Vector geometry processing tools - GeometryOps.jl, LibGEOS.jl, and more
  - Raster processing tools - Rasters.jl
  - Visualization tools - Tyler.jl, GeoMakie.jl
  - All tied together by **GeoInterface.jl**

# The last year

- Intro to ecosystem - broadly the same
- Vector I/O interface formalization - complete!
- Vector I/O consolidation under GeoDataFrames - complete!
- Vector data cubes - not there yet :(
- Formalized governance
- Lots of package improvements!

# Looking ahead

- Overhaul the JuliaGeo new user experience - website, tutorials, references and user guides
- Finish up and merge vector data cubes in Rasters.jl
- Geo.jl, a convenience package so you run `using Geo` and everything works
- Working regridding (maybe this week even)
- ... (to fill in later)

# Governance and Community

- New Steering committee and organizational structure (JuliaGeo/governance)
  - Currently we have 8 people.
  - Empowered to resolve technical and organizational/social questions.
  - Ideally meet once a quarter
- Monthly community meetings!
  - Still need to find more people to help run them though (hint hint :D)
- Next on the list: JuliaGeo mission statement and website overhaul

# GeoDataFrames

- New version 0.4 with big changes!
- Selection of I/O backends based on what packages are loaded (via extensions)
- Switched from ArchGDAL.jl methods to GeometryOps.jl methods for geometry processing

See the migration guide here!

To do:

- Spatial tree acceleration for geometry operations (needs some GeometryOps changes)

# GeometryOps

- Tree accelerations available for planar intersection!  Gives a logarithmic speed improvement for large polygons
- Simple and consistent interface for trees with any kind of node type (lat/long, spherical cap, etc.)
- WIP - spherical geometry operations!  Area, perimeter, distance etc. implemented - working on polygon intersection now

# Regridding

- Two approaches both powered by GeometryOps:
  - SphericalSpatialTrees.jl: big data regridding, single shot, chunk-aware
  - ConservativeRegridding.jl: create an operator to switch from one grid to another, mainly for climate models
- Both using spherical spatial trees

# Rasters

- Crazy rasterization!
- Speed and efficiency improvements
- More adherence to CF standards - and even more in the pipeline
- A *lot* of improvements in DimensionalData too

# Ecosystem integration

- Standardized function names across the ecosystem
-

# Plotting (GeoMakie)

- New GlobeAxis available and tested, to plot things on the globe!
  - Needs some help with UX and scrolling behaviour but otherwise works for most things

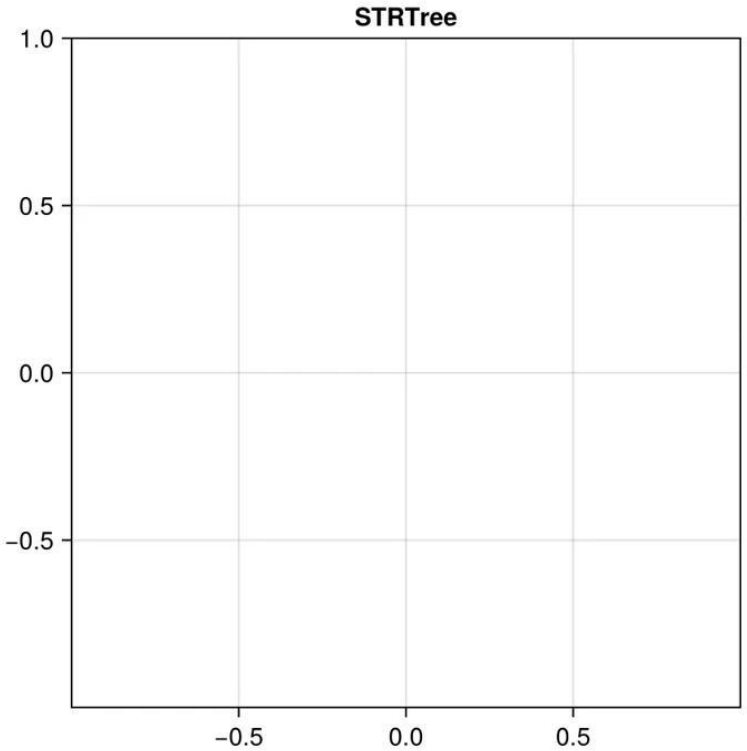
# Plotting (GeoMakie)

- Another square axis type in progress with Tyler integration etc.

# Thanks and see you next year!

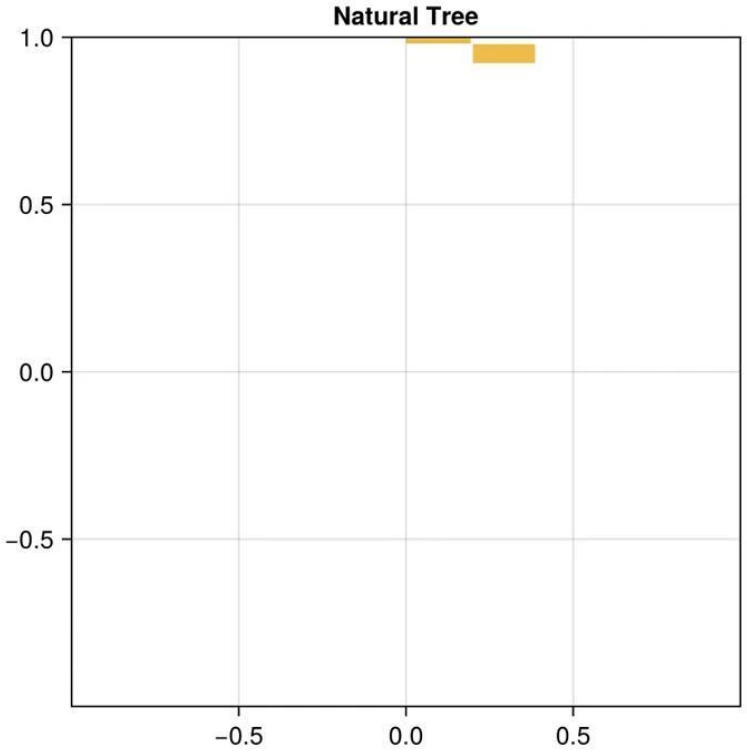
JuliaGeo

# STRtree on a circle



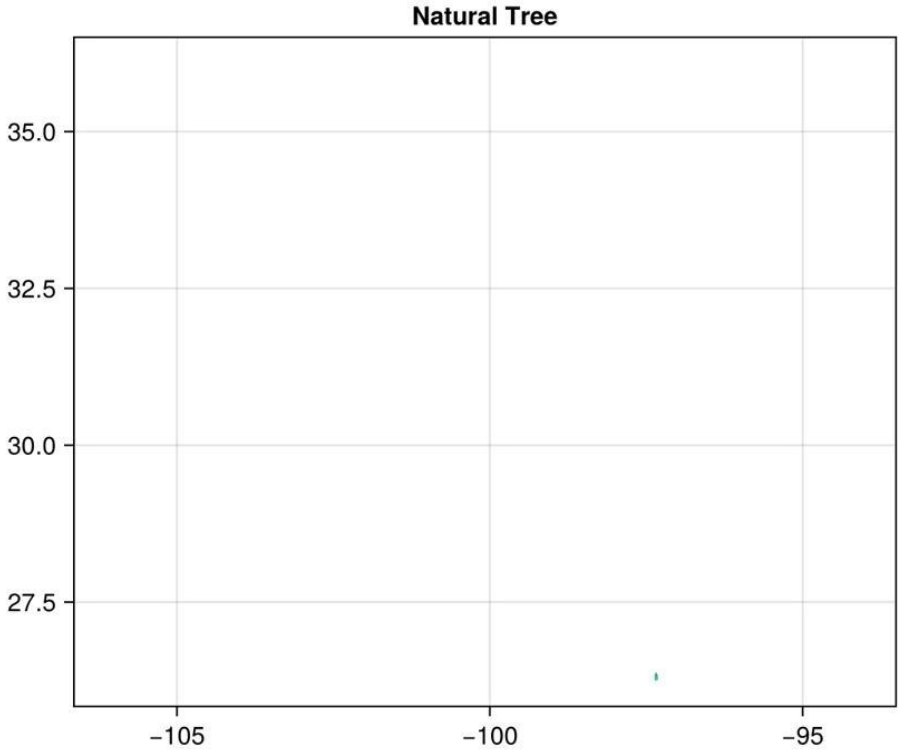
JuliaGeo

# Natural tree on circle



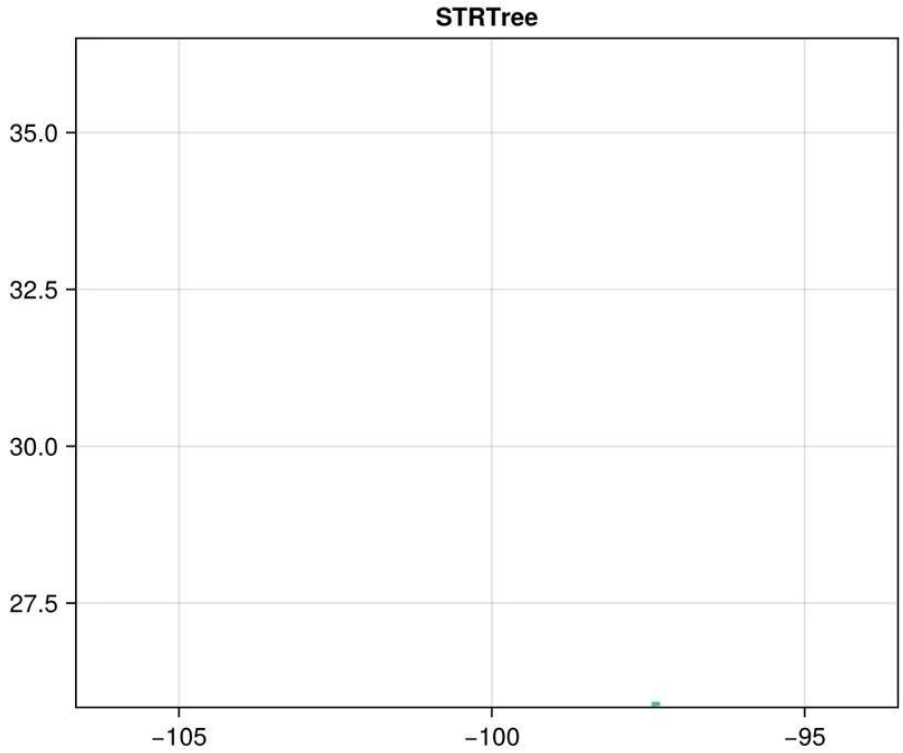
JuliaGeo

# Natural tree on Texas



JuliaGeo

# STRtree on Texas



# New manifold interface

- Is it a plane? Is it a sphere?  No, it's an ellipsoid!
- Specify sphere radius or ellipsoid parameters
  - Planar()
  - Spherical(radius=1)
  - Geodesic(datum=wgs84)
- Geometry operations can dispatch on the manifold - planar, spherical, geodesic can all have different implementations but benefit from shared infrastructure

# New manifold interface

Example: perimeter implementation

# New algorithm interface

- Store the manifold in the algorithm
- Re-usable and rebuildable
- struct MyAlg
- area(MyAlg(), geom)