C++ Acceleration of Thermodynamics Module in Metpy

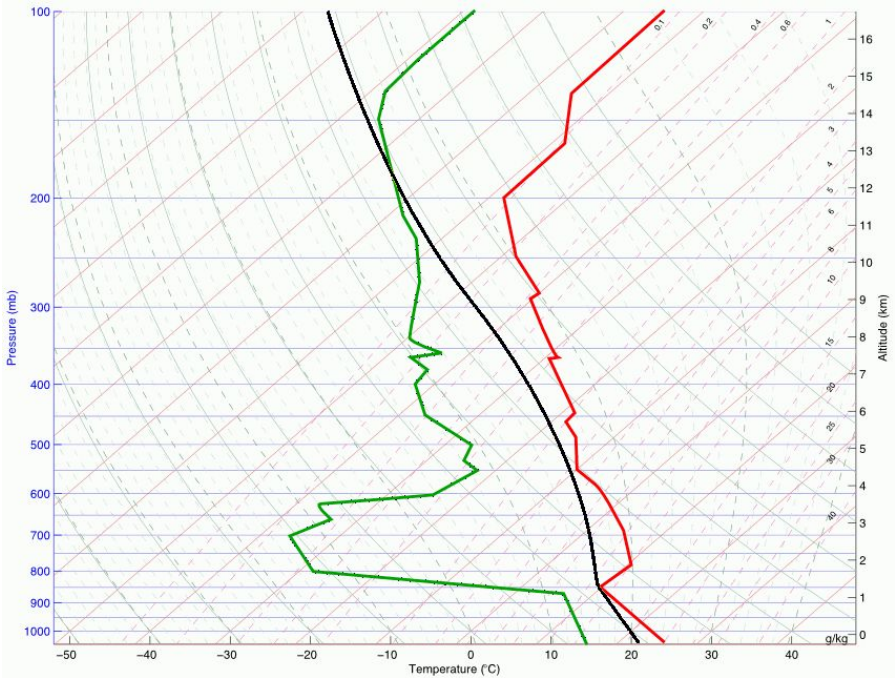
Internship at Unidata

Linfeng Li

07/23/2025

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

CAPE Convective Available Potential Energy



Key building blocks:

- LCL, LFC, EL
- Dry Lapse, Moist Lapse
- Parcel Profile
- Finding intersection
- Integration

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Pybind11

pybind11 is a lightweight header-only library that exposes C++ types in Python and vice versa, mainly to create Python bindings of existing C++ code.

```cpp
#include <pybind11/pybind11.h>

int add(int i, int j) {
    return i + j;
}

PYBIND11_MODULE(_calc_mod, m) {
    m.doc() = "accelerator module docstring";
    m.def("add", &add, "Add two numbers");
}
```

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Which part should be/not be c++

Rewriting everything in c++?

1 Units

2 Input check

3 Scalar/Array input

4 Dimension Broadcast

5 Constants

6 Computing

7 Assembling

8 Etc.

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Which part should be/not be c++

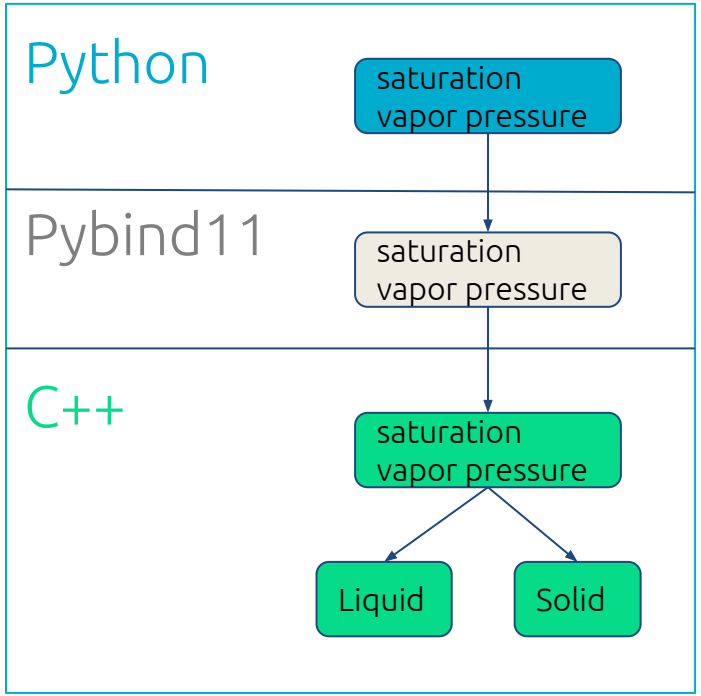
Example 1 saturation vapor pressure

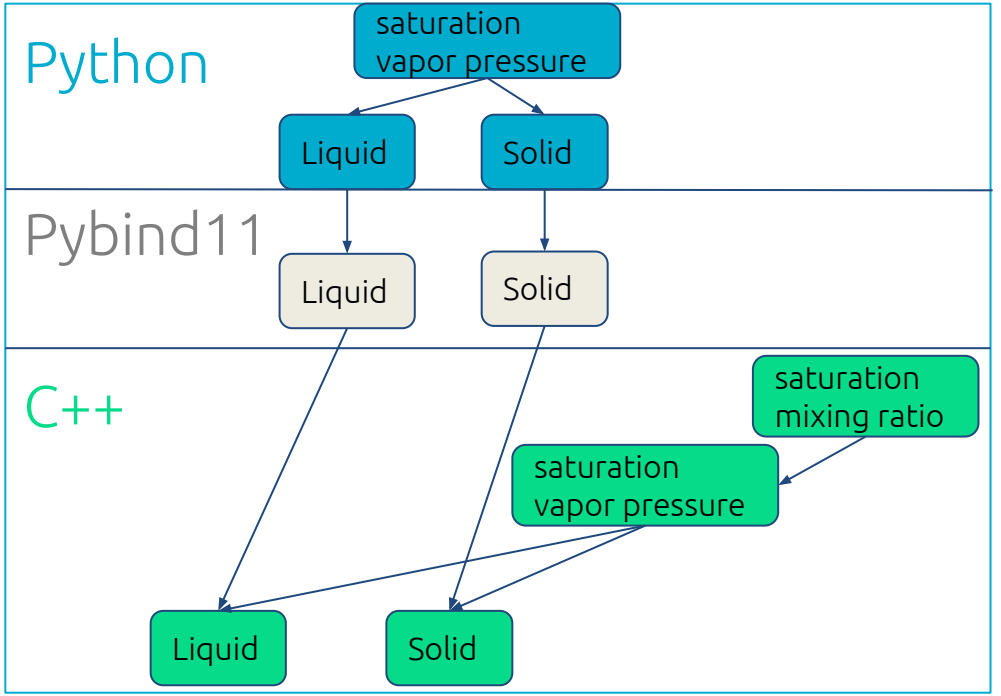
```
def saturation_vapor_pressure(temperature, *, phase='liquid'):
    match phase:
        case 'liquid':
            return _saturation_vapor_pressure_liquid._nounit(temperature)
        case 'solid':
            return _saturation_vapor_pressure_solid._nounit(temperature)
        case 'auto':
            return np.where(temperature > mpconsts.nounit.T0,
                            _saturation_vapor_pressure_liquid._nounit(temperature),
                            _saturation_vapor_pressure_solid._nounit(temperature))
        case _:
            raise ValueError(
                f'{phase!r} is not a valid option for phase. '
                f"Valid options are {'liquid', 'solid', 'auto'}.")
```

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Which part should be/not be c++

Example 1 saturation vapor pressure



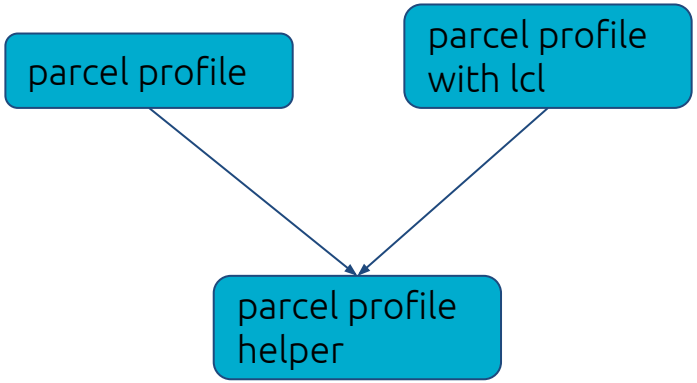


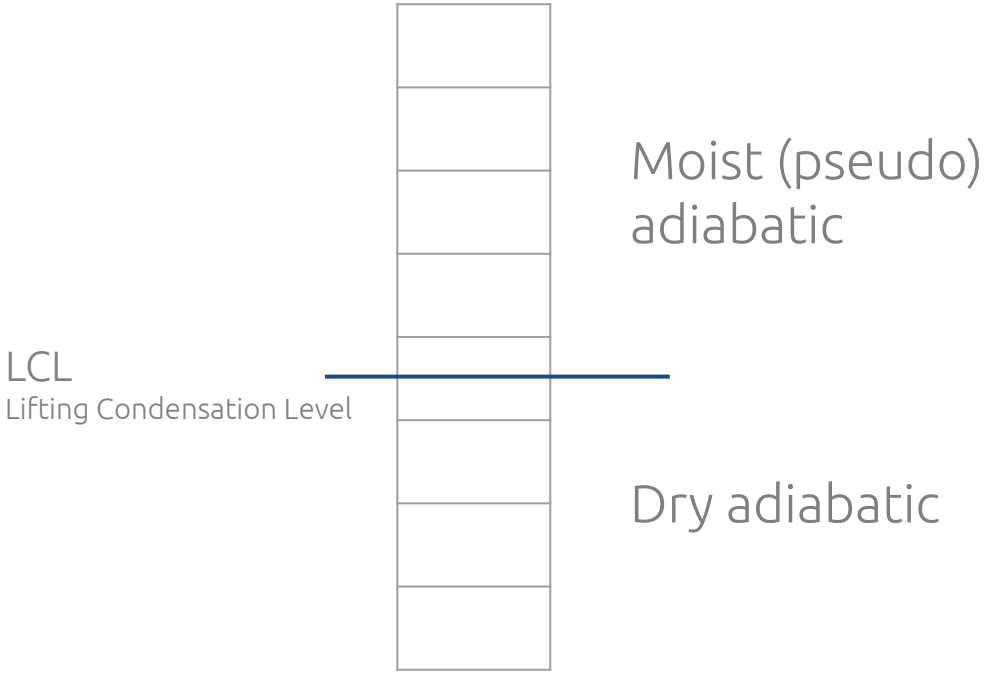
This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Which part should be/not be c++

Example 2 parcel profile helper

Python

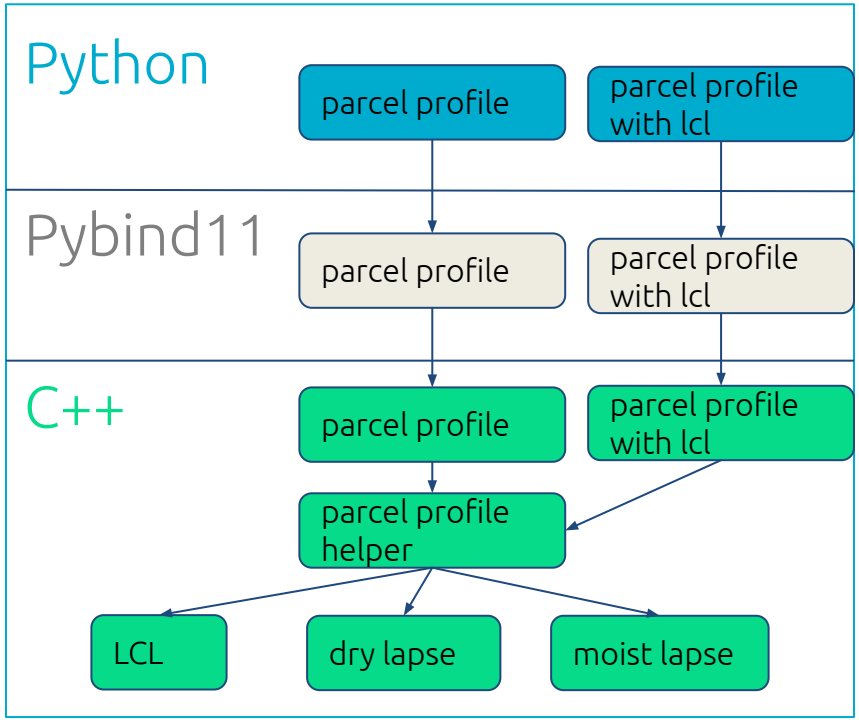


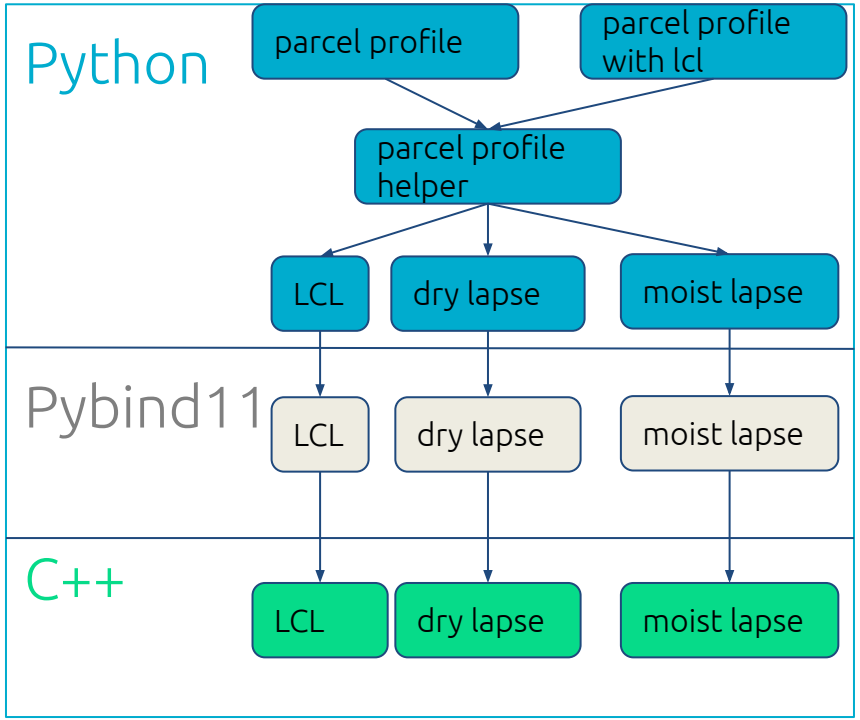


This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Which part should be/not be c++

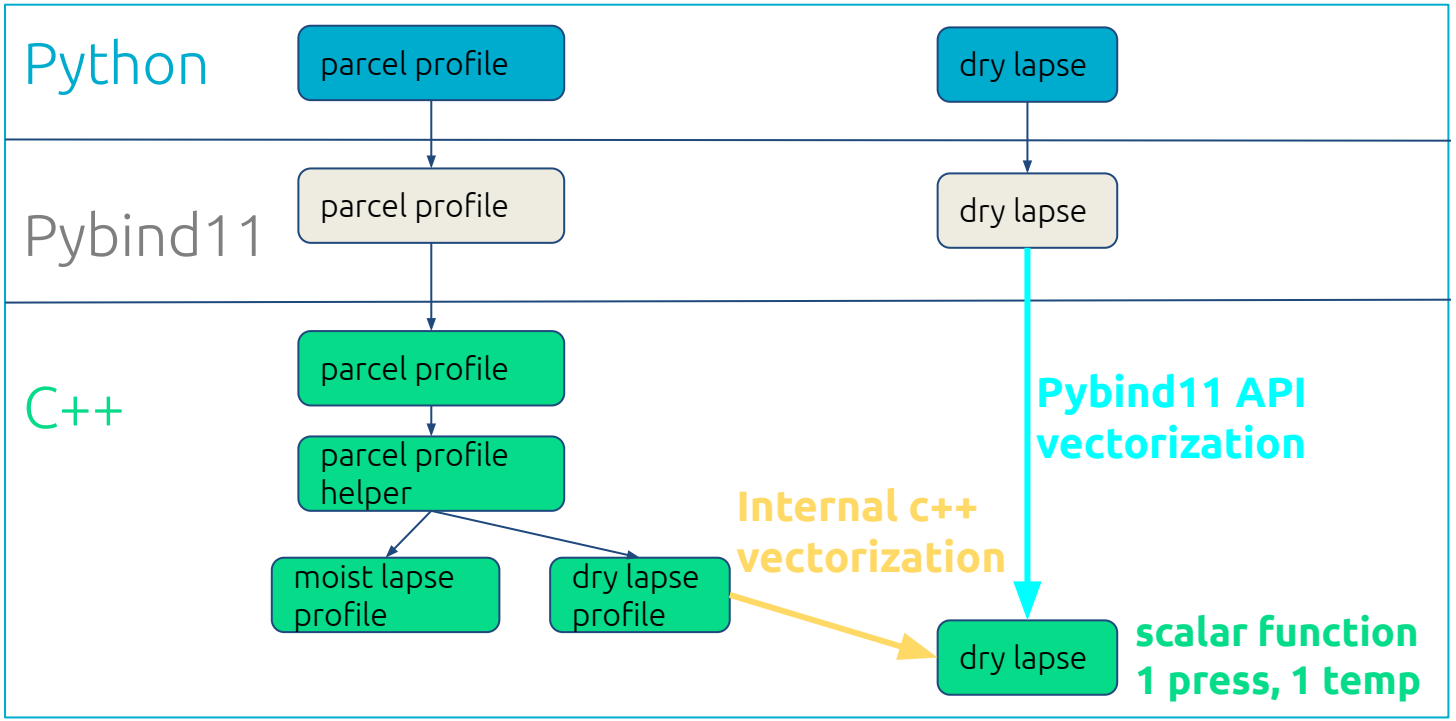
Example 2 parcel profile helper





This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Internal C++ Vectorization .vs. Pybind11 API vectorization



This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Function Vectorization

```
m.def("dewpoint", py::vectorize(DewPoint),
        "Calculate dewpoint from water vapor partial pressure.",
        py::arg("vapor_pressure"));

m.def("mixing_ratio", py::vectorize(MixingRatio),
        "Calculate the mixing ratio of a gas.",
        py::arg("partial_press"), py::arg("total_press"), py::arg("epsilon"));
```

```
double my_func(int x, float y, double z);

m.def("vectorized_func", py::vectorize(my_func));

>>> x = np.array([[1, 3], [5, 7]])
>>> y = np.array([[2, 4], [6, 8]])
>>> z = 3
>>> result = vectorized_func(x, y, z)
```

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Function Vectorization

py::vectorize doesn't work for **lcl**

lcl returns two values: p_lcl and t_lcl

scalar function

```cpp
std::pair<double, double> LCL(double pressure, double temperature, double dewpoint) {
    if (temperature < dewpoint) {
        std::cerr << "Warning in function '" << __func__
            << "': Temperature must be greater than dew point for LCL calculation.\n";
        return {std::numeric_limits<double>::quiet_NaN(), std::numeric_limits<double>::quiet_NaN()};
    }

    double q = SpecificHumidityFromDewPoint(pressure, dewpoint, "liquid");
    double moist_heat_ratio = MoistAirSpecificHeatPressure(q) / MoistAirGasConstant(q);
    double spec_heat_diff = mc::Cp_l - mc::Cp_v;

    double a = moist_heat_ratio + spec_heat_diff / mc::Rv;
    double b = -(mc::Lv + spec_heat_diff * mc::T0) / (mc::Rv * temperature);
    double c = b / a;

    double rh = RelativeHumidityFromDewPoint(temperature, dewpoint, "liquid");
    double w_minus1 = lambert_wm1(pow(rh, 1.0 / a) * c * exp(c));
    double t_lcl = c / w_minus1 * temperature;
    double p_lcl = pressure * pow(t_lcl / temperature, moist_heat_ratio);

    return {p_lcl, t_lcl};
}
```

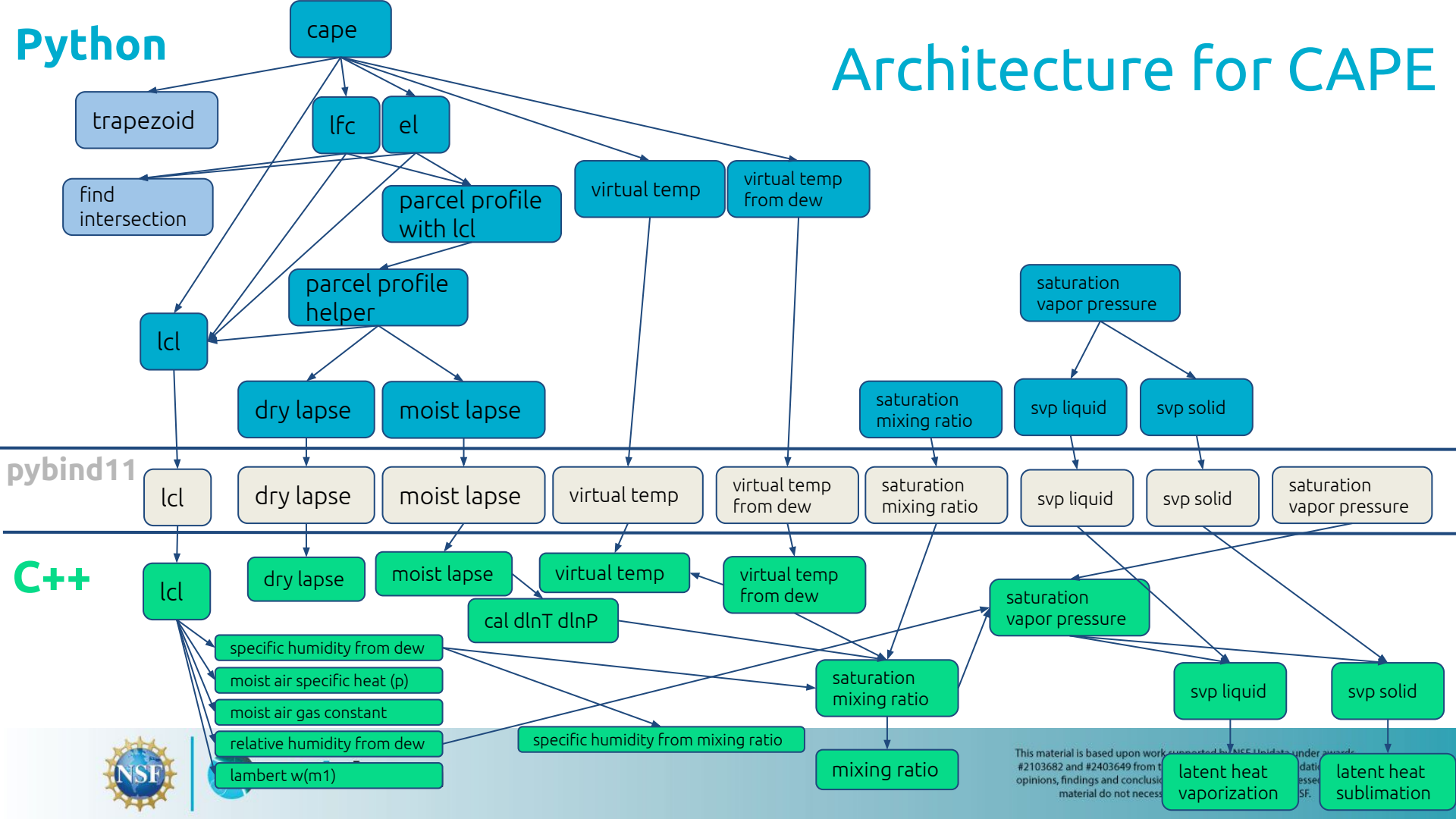
manually vectorization

```cpp
std::tuple<py::array_t<double>, py::array_t<double>> LCLVectorized(py::array_t<double> pressure,
                                                                   py::array_t<double> temperature,
                                                                   py::array_t<double> dewpoint) {

    // This helper ensures the arrays are in C-style contiguous memory.
    // If an input array is already contiguous, it's a zero-cost operation.
    // If it's a slice or has a different memory layout, it creates a copy.
    // This makes the subsequent looping simple and safe.
    auto p_contig = py::array::ensure(pressure, py::array::c_style);
    auto t_contig = py::array::ensure(temperature, py::array::c_style);
    auto d_contig = py::array::ensure(dewpoint, py::array::c_style);
```

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Architecture for CAPE



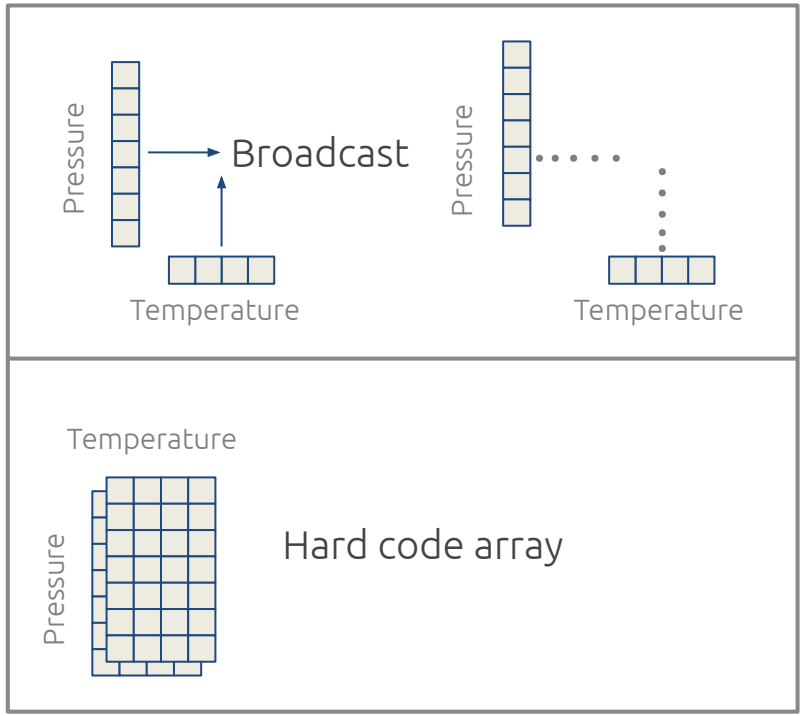
This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from t
opinions, findings and conclusi
material do not neces
SF.

airspeed velocity (asv) Performance Benchmark

Benchmark ↓	Value	Recent change
cape.CapeSuite.time_cape_cin	2.197ms	–6.6% (-155.871µs)
cape.CapeSuite.time_el	2.365ms	–55.2% (-2.913ms)
cape.CapeSuite.time_el_prof_provided	2.375ms	–55.6% (-2.972ms)
cape.CapeSuite.time_lfc	2.386ms	–54.8% (-2.896ms)
cape.CapeSuite.time_lfc_prof_provided	2.386ms	–54.7% (-2.882ms)
drylapse_broadcast.Drylapse_Broadcast_Array_Suite.time_drylapse_1D	89.547µs	–16.7% (-18.000µs)
drylapse_broadcast.Drylapse_Broadcast_Array_Suite.time_drylapse_2D	150.833µs	+17.6% (+22.614µs)
drylapse_broadcast.Drylapse_Broadcast_Array_Suite.time_drylapse_3D	4.720ms	+1487.3% (+4.423ms)
drylapse_hardcode.Drylapse_Hardcode_Array_Suite.time_drylapse_1D	89.858µs	–16.5% (-17.716µs)
drylapse_hardcode.Drylapse_Hardcode_Array_Suite.time_drylapse_2D	145.888µs	–1.3% (-1.848µs)
drylapse_hardcode.Drylapse_Hardcode_Array_Suite.time_drylapse_3D	5.489ms	+46.3% (+1.737ms)
lcl.LCLSuite.time_lcl((1, 1, 200))	353.956µs	–20.8% (-92.850µs)
lcl.LCLSuite.time_lcl((50, 1, 1))	298.066µs	–17.9% (-64.839µs)
lcl.LCLSuite.time_lcl((50, 200, 200))	702.773ms	–39.7% (-462.690ms)
lcl.LCLSuite.time_lcl((50, 800, 800))	11.544s	–42.8% (-8.655s)
moistlapse.MoistlapseSuite.time_moistlapse	446.519µs	–91.4% (-4.744ms)
moistlapse.MoistlapseSuite.time_moistlapse_matrix	3.559s	
moistlapse.MoistlapseSuite.time_moistlapse_vector	35.700ms	+484.7% (+29.594ms)
parcel_profile.ParProfSuite.time_parcel_profile	1.651ms	–65.2% (-3.097ms)

Dry Lapse Dilemma

Benchmark	Time Changes
Broadcast drylapse_1D	−16.7% (-18.000µs)
Broadcast drylapse_2D	+17.6% (+22.614µs)
Broadcast drylapse_3D	+1487.3% (+4.423ms)
Hard Code drylapse_1D	−16.5% (-17.716µs)
Hard Code drylapse_2D	−1.3% (-1.848µs)
Hard Code drylapse_3D	+46.3% (+1.737ms)



Python is super efficient in processing broadcasted array by its own.

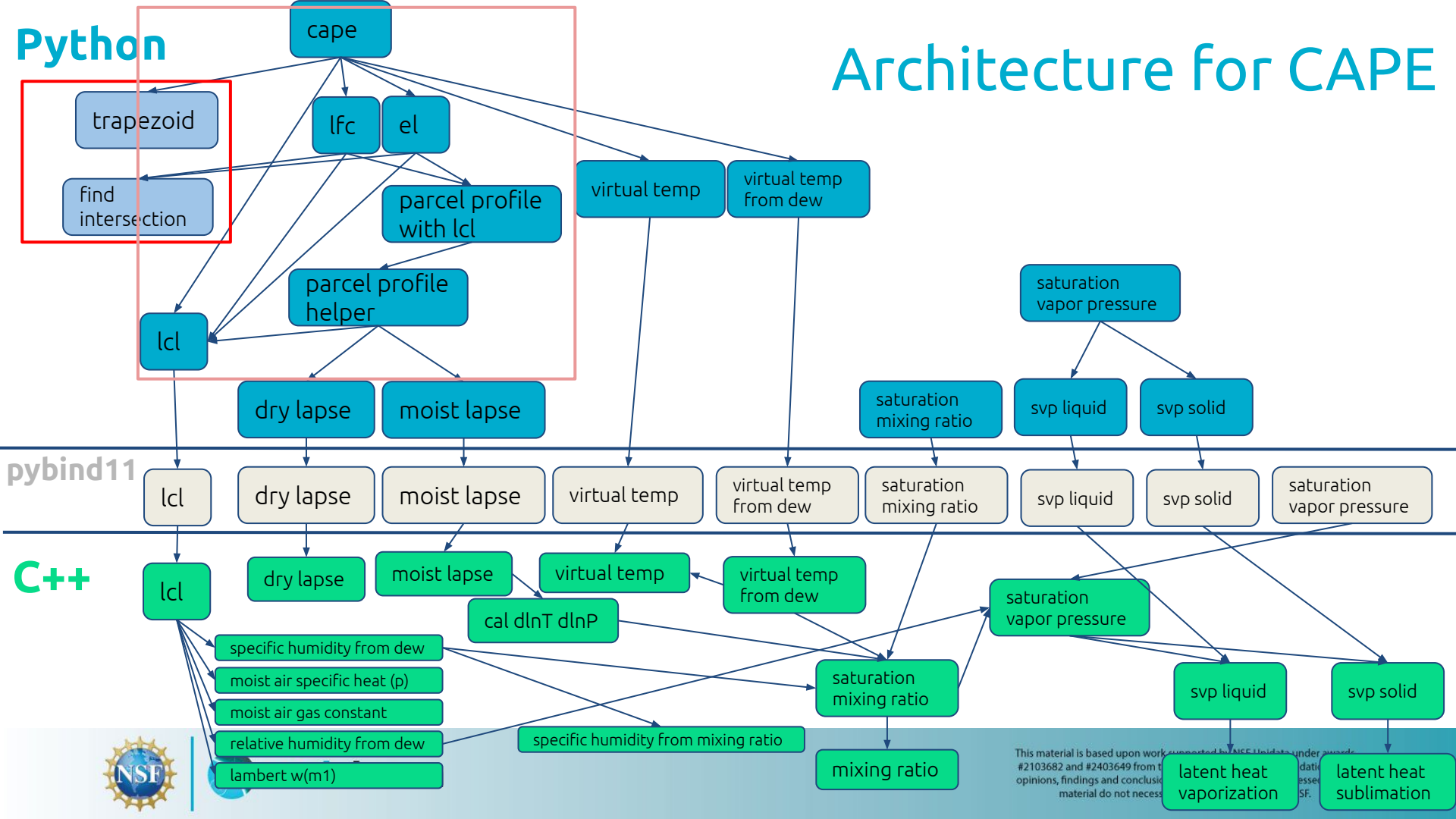
This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Future Work

- find_intersection may limit the speed.
- Optimize the calling chain. LCL is repeatedly called.
- Resolve the drylapse issue.
- Move more python-level functions to c++ level: parcel_profile_helper
- Main stem of CAPE has been touched, will look at more minor branches

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Architecture for CAPE



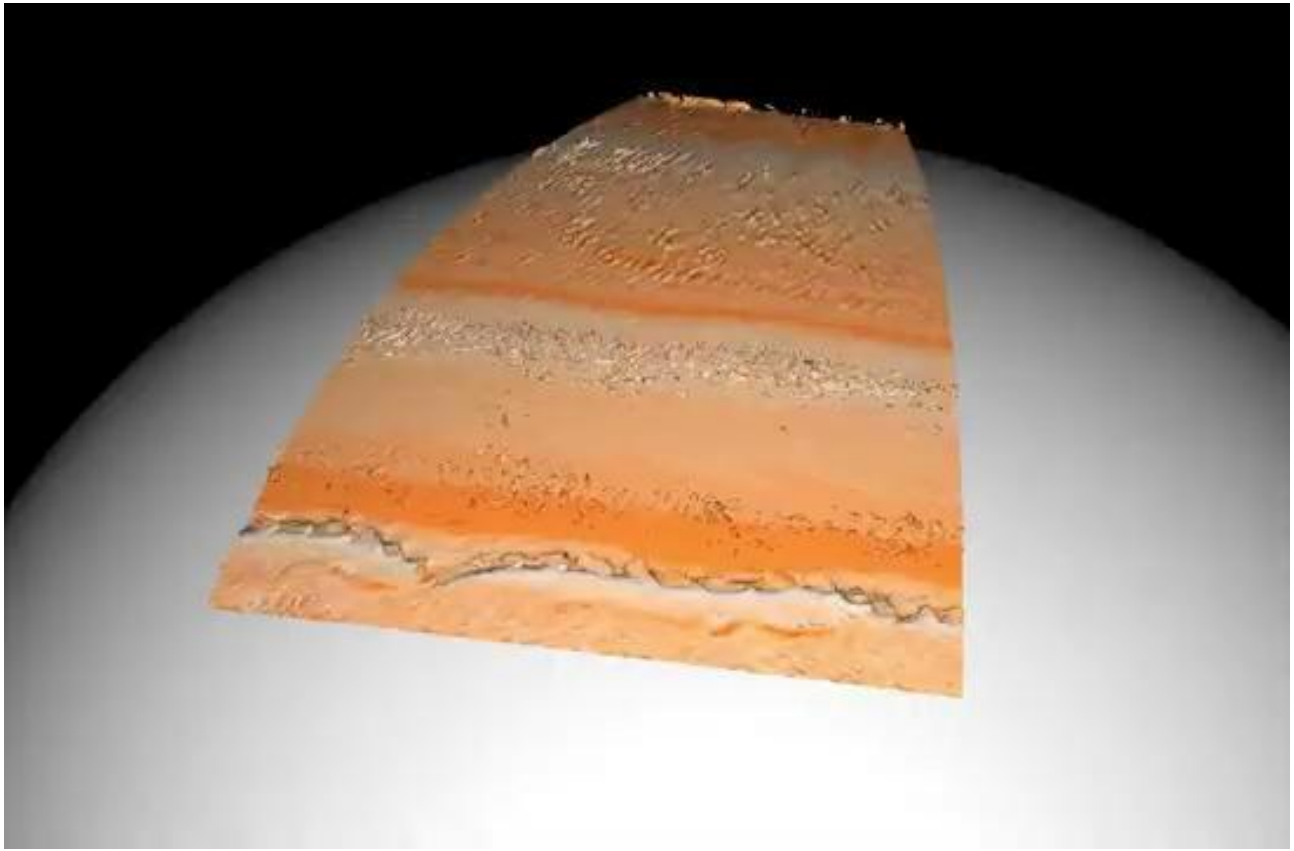
This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from t
opinions, findings and conclusi
material do not neces
SF.

Jupiter Visualization in IDV

- β-plane simulation
- 0.1 ~ 80 bar, 20 ~ 70 N, 0 ~ 52 E
- resolution: 170 * 300 * 400

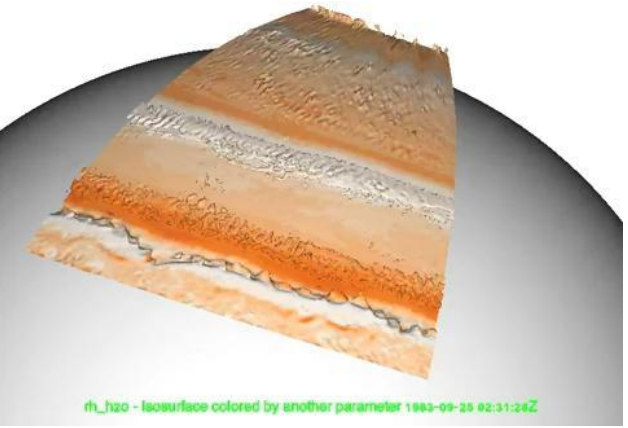
Data is in courtesy of Huazhi Ge, California Institute of Technology,
Large-Scale Atmospheric Dynamics in Jupiter's Mid-latitude — Compositional Structure, Thermal Structure, Large-Scale Dynamics, and Forward-Modelling Compared to Radio Observation — Simulated by the Nonhydrostatic Model

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

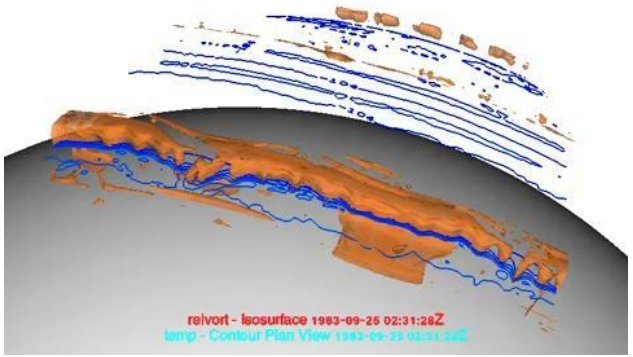


Intricate vortices and the distinct band-and-zone structure of Jupiter's atmosphere. Time span is 20 Earth days.

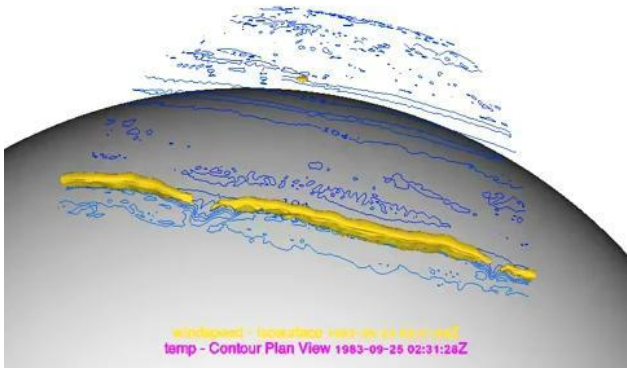
This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.



vorticity on H_2O relative humidity isosurface



relative vorticity isosurface



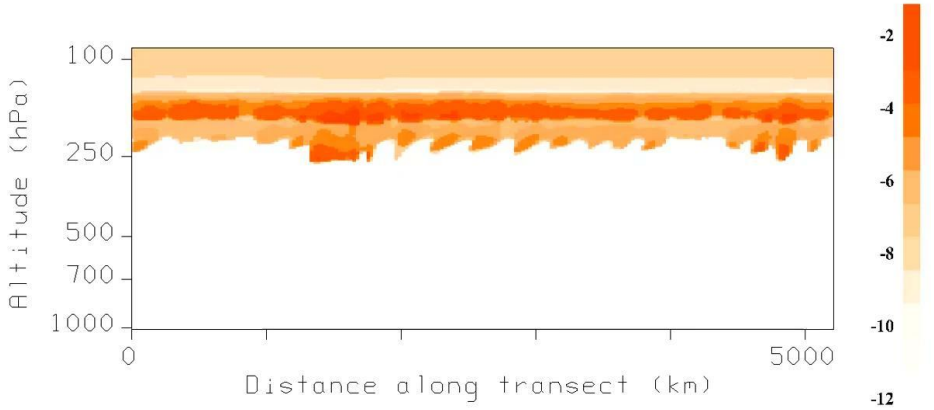
wind speed isosurface

Videos In courtesy of Yuan Ho, UCAR Unidata

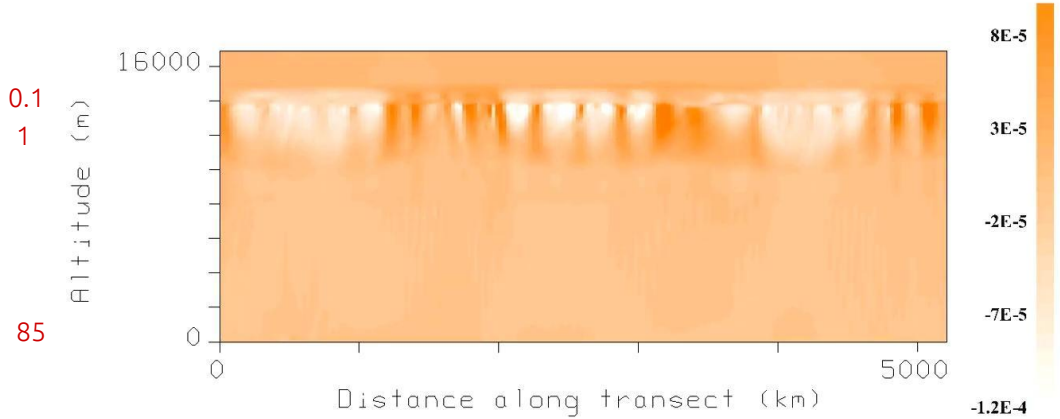
This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

Vertical Cross Section

H2O & NH3 cloud density

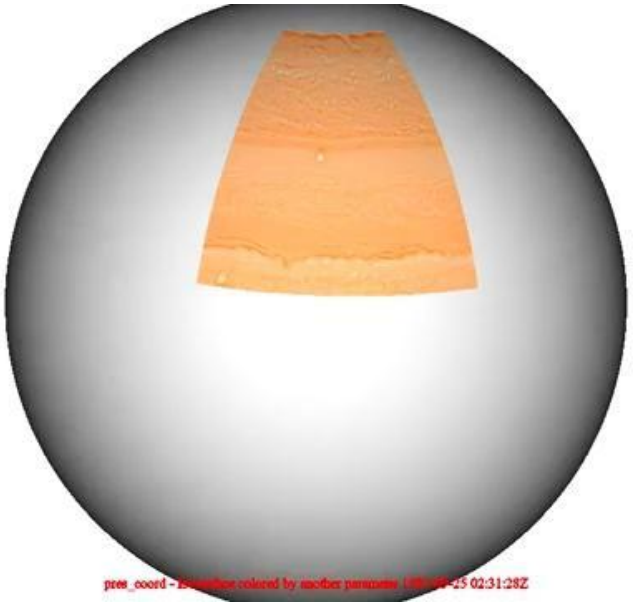


Relative vorticity

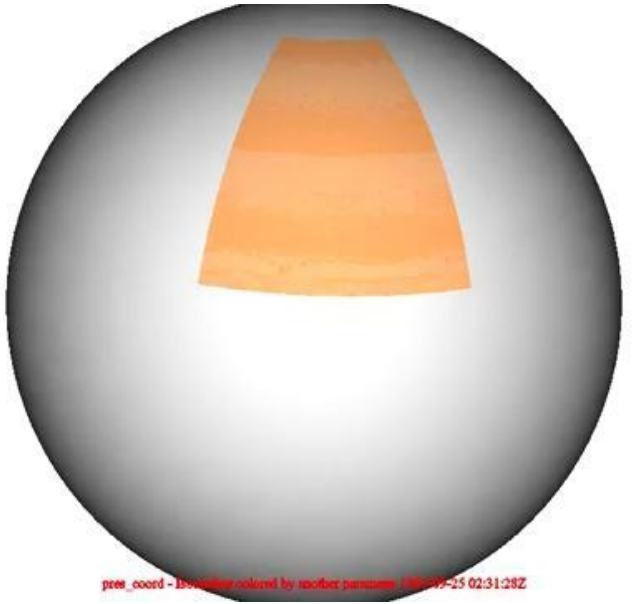


Shallow layer vortices at 0.1 bar extend down to 85 bar pressure level.

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.

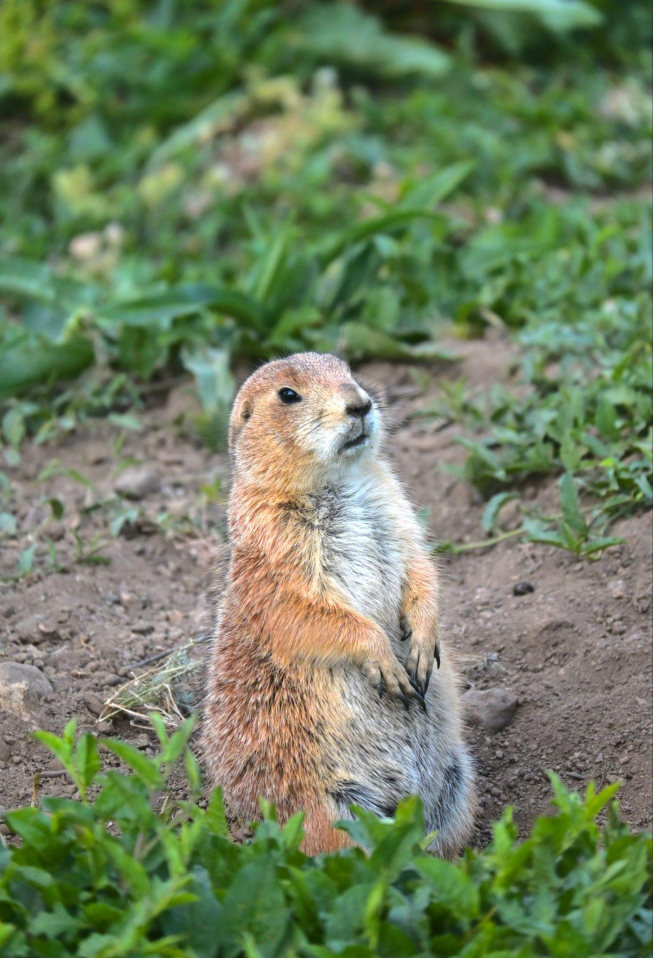


relative vorticity on 0.1 bar



relative vorticity on 2 bar

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.



Acknowledgement

NSF Unidata is one of the Univrsity Corporation for Atmospheric Research (UCAR)'s Community Programs (UCP), and is funded primarily by the U. S. National Science Foundation (Grant AGS-1901712).

This material is based upon work supported by NSF Unidata under awards #2103682 and #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.


Grandview Ave
30