Julia for AGC

Atell-Yehor Krasnopolski
IRIS-HEP Fellow
Universität Würzburg

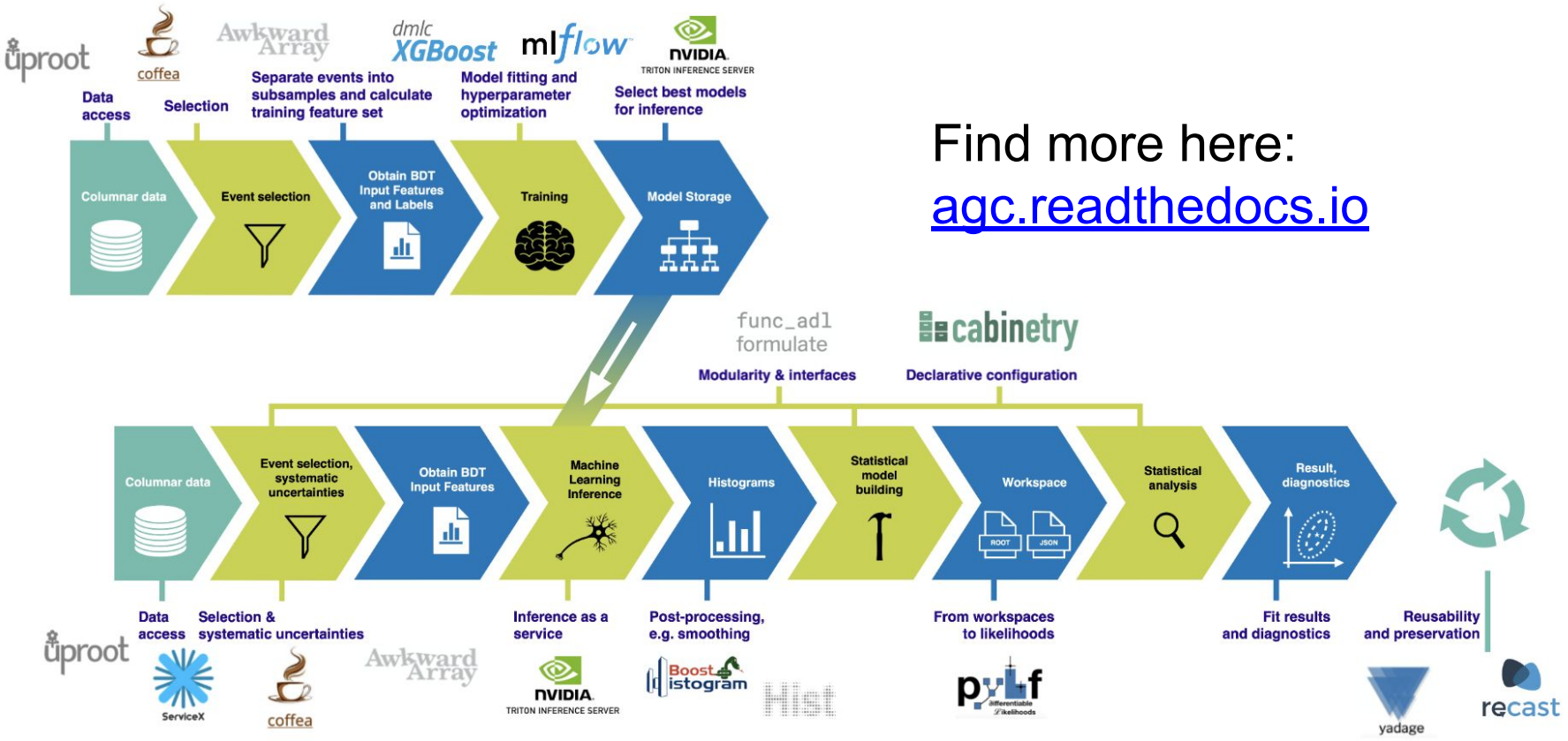
Jerry Ling
Harvard University

Alexander Held
UWM

Analysis Grand Challenge

- columnar data extraction from large datasets
- processing of that data (event filtering, construction of observables, evaluation of systematic uncertainties) into histograms
- statistical model construction and statistical inference
- relevant visualisations for these steps

Analysis Grand Challenge



Find more here:
agc.readthedocs.io

Data

number of files	total size	number of events
9	22.9 GB	10,455,719
18	42.8 GB	19,497,435
43	105 GB	47,996,231
79	200 GB	90,546,458
140	359 GB	163,123,242
255	631 GB	297,247,463
395	960 GB	470,397,795
595	1.40 TB	705,273,291
787	1.78 TB	940,160,174

julia for this task

- Less than 100 lines of code for the main loop
- Plotting, distributed computing, and working with complex data structures

- A bug in UnROOT.jl had to be fixed

```
function get_histo(tree, wgt; file_variation::Symbol=:nominal, evts=nothing)
    is_nominal_file = (:nominal == file_variation)
    hists = generate_hists(file_variation)
    #threads_ithreads for evt in tree
    for evt in tree
        # single lepton requirement
        (; Electron_pt, Muon_pt) = evt
        (count(>(25), Electron_pt) + count(>(25), Muon_pt) != 1) && continue

        # get_pt
        (; Jet_pt) = evt
        is_nominal_file && (Jet_pt_nominal = Jet_pt)

        for hist_type in (is_nominal_file ? keys(SHAPE_VARS) : (:nominal,))
            # modify pt
            is_nominal_file && (Jet_pt = SHAPE_VARS[hist_type](Jet_pt_nominal))

            scale_info = (; Jet_pt, wgt)

            # at least 4 jets
            jet_pt_mask = Jet_pt .> 25
            if count(jet_pt_mask) >= 4
                jet_btag = @view evt.Jet_btagCSVV2[jet_pt_mask]

                btag_count = count(>(0.5), jet_btag)
                # MASS HISTOGRAM
                if btag_count >= 2 # at least 2 btag
                    if evts !== nothing
                        if hist_type in keys(evts)
                            push!(evts[hist_type], evt.event)
                        end
                    end

                    (; Jet_eta, Jet_phi, Jet_mass) = evt

                    # construct jet lorentz vector
                    jet_p4 = @views LorentzVectorCyl.(Jet_pt[jet_pt_mask], Jet_eta[jet_pt_mask], Jet_phi[jet_pt_mask], Jet_mass[jet_pt_mask])

                    Njets = length(jet_btag)

                    # tri_jet combinatorics
                    max_pt = -Inf
                    local best_mass

                    # 1. all tri-jet combinations
                    for comb in Combinations(Njets, 3)
                        p4s = @view jet_p4[comb]
                        btags = @view jet_btag[comb]
                        # 2. keep those maximum(btags1,2,3) > 0.5
                        maximum(btags) <= 0.5 && continue
                        tri = sum(p4s)
                        _pt = pt(tri)
                        # 3. pick the tri-p4 with highest tri-pt
                        if _pt > max_pt
                            max_pt = _pt
                            best_mass = mass(tri)
                        end
                    end

                    # tri-p4 with highest tri-pt first
                    push!(hists[Symbol(:mbjj_4j2b_, (is_nominal_file ? hist_type : file_variation))], best_mass, wgt)
                    if is_nominal_file && (hist_type == :nominal)
                        @scale_var_loop :mbjj_4j2b best_mass
                    end
                # HT HISTOGRAM
                elseif btag_count == 1 # no more than 1 btag
                    HT = @views sum(Jet_pt[jet_pt_mask])
                    push!(hists[Symbol(:HT_4j1b_, (is_nominal_file ? hist_type : file_variation))], HT, wgt)
                    if is_nominal_file && (hist_type == :nominal)
                        @scale_var_loop :HT_4j1b HT
                    end
                end
            end
        end
    end

    hists
end
```

Atell-Yehor Krasnopolski

julia for this task

```python
class TtbarAnalysis(processor.ProcessorABC):
    def __init__(self, use_inference, use_triton):
        ...

    def only_do_IO(self, events):
        ...

    def process(self, events):
        # main loop here
        ...

    def postprocess(self, accumulator):
        ...
```

```julia
function get_all_hists(; ...)
    ...
end

function get_histo(process_tag::Symbol; ...)
    ...
end

function get_histo_distributed(process_tags::Vector{Symbol}; ...)
    ...
end

function get_histo(tree, wgt; file_variation::Symbol=:nominal, evts=nothing)
    # main loop here
    ...
end
```

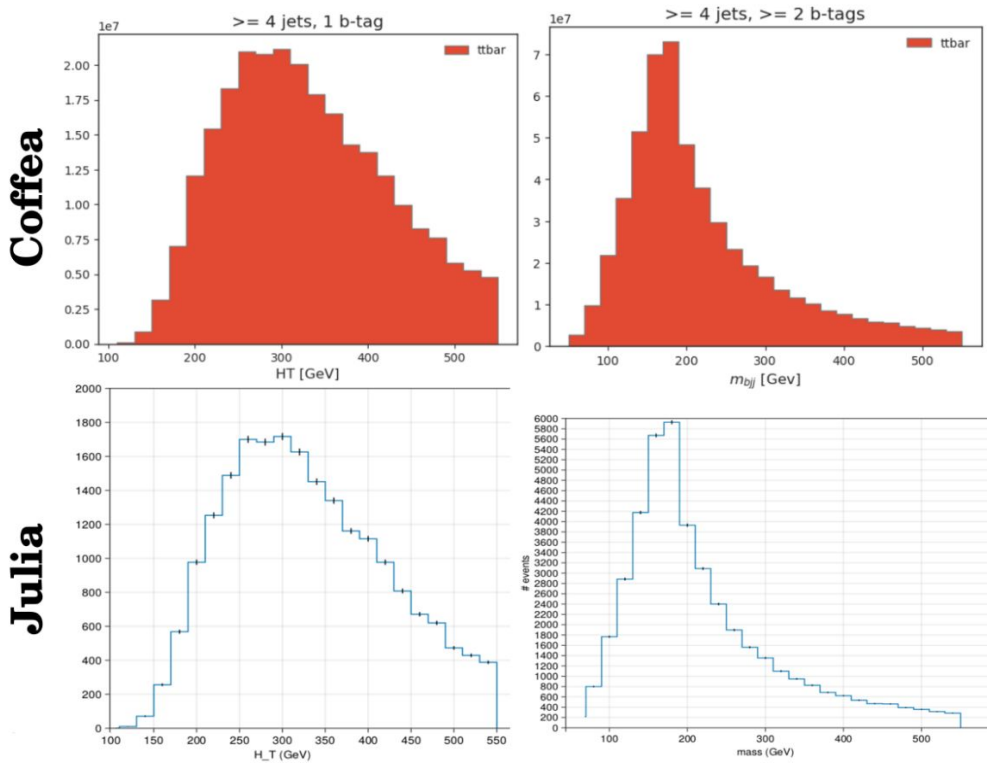
The cuts

```
# at least 4 jets
jet_pt_mask = Jet_pt .> 25
if count(jet_pt_mask) >= 4
    jet_btag = @view evt.Jet_btagCSVV2[jet_pt_mask]

    btag_count = count(>(0.5), jet_btag)
    # MASS HISTOGRAM
    if btag_count >= 2 # at least 2 btag
        # add to the mass (mbjj) histogram
        ...
    # HT HISTOGRAM
    elseif btag_count == 1 # no more than 1 btag
        # add to the HT histogram
        ...
    end
end
```

Main features

- The whole pipeline (except ML-related parts)
- Generating correct histograms with native Julia up to bin migrations
- Systematic variations implemented

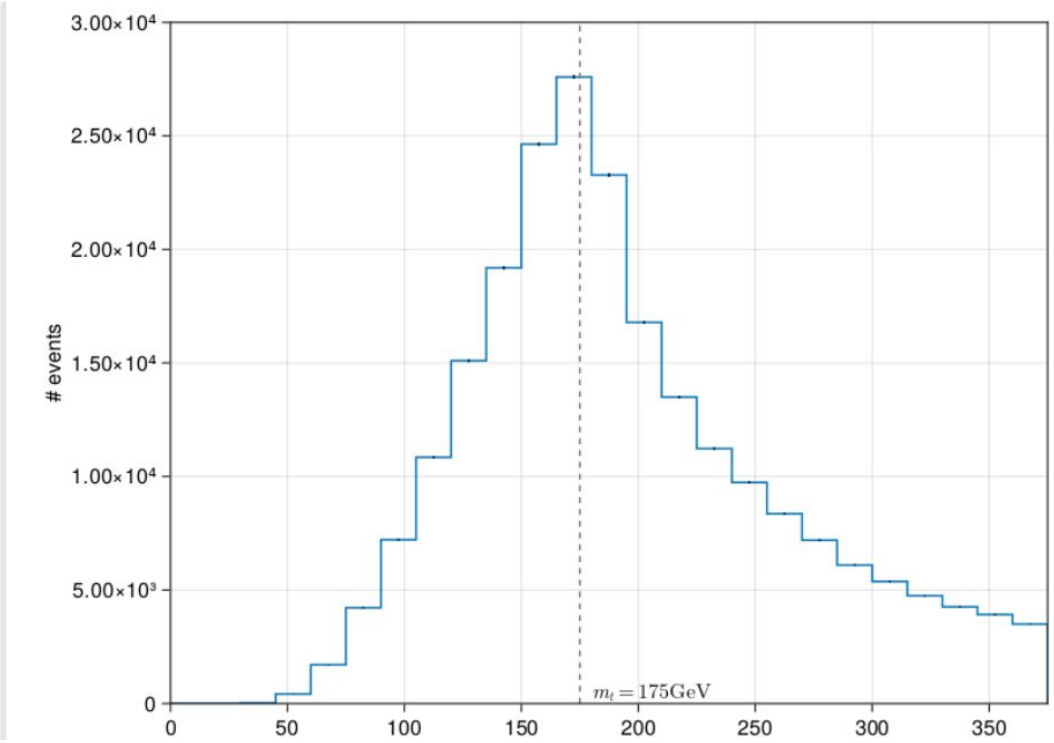


Systematic variations

```
julia> all_hists[:ttbar]
Dict{Symbol, Hist1D{Float64, Tuple{StepRangeLen{Float64, Base.TwicePrecision{Float64}, Base.TwicePrecision{Float64}, Int64}}}} with
36 entries:
  :HT_4j1b_pt_scale_up       => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :HT_4j1b_btag_var_0_up     => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :HT_4j1b_nominal           => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :mbjj_4j2b_nominal         => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :HT_4j1b_PS_var            => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :HT_4j1b_scale_var_up      => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :mbjj_4j2b_btag_var_2_up   => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :HT_4j1b_scaledown         => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :HT_4j1b_btag_var_3_down   => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :mbjj_4j2b_pt_scale_up     => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :mbjj_4j2b_PS_var          => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :mbjj_4j2b_btag_var_1_up   => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :mbjj_4j2b_btag_var_0_down => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  :mbjj_4j2b_btag_var_1_down => Hist1D{Float64}, edges=50.0:20.0:550.0, integral=0.0
  ⋮                          => ⋮
```

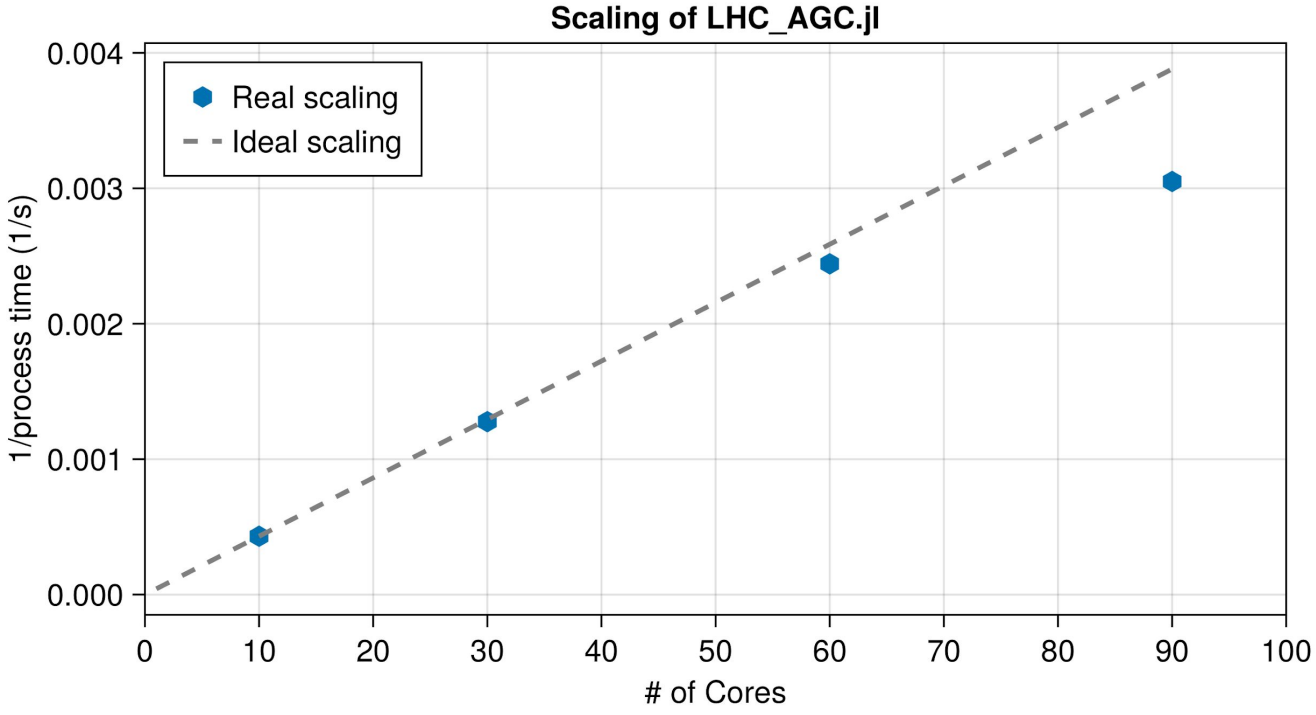
```
# let's not weight this one
res = @time LHC_AGC.get_histo(:ttbar; wgt=1.0, n_files_max_per_sample=1);
```

```
["/home/akako/Documents/github/LHC_AGC.jl/data/cmsopendata2015_ttbar_19980_PU25nsData2015v1_76X_mcRun2_asymptotic_v12_ext3-v1_00000_0000.root"]
/home/akako/Documents/github/LHC_AGC.jl/data/cmsopendata2015_ttbar_19980_PU25nsData2015v1_76X_mcRun2_asymptotic_v12_ext3-v1_00000_0000.root
 17.239748 seconds (243.85 M allocations: 11.323 GiB, 10.92% gc time, 0.18% compilation time: 61% of which was recompilation)
```



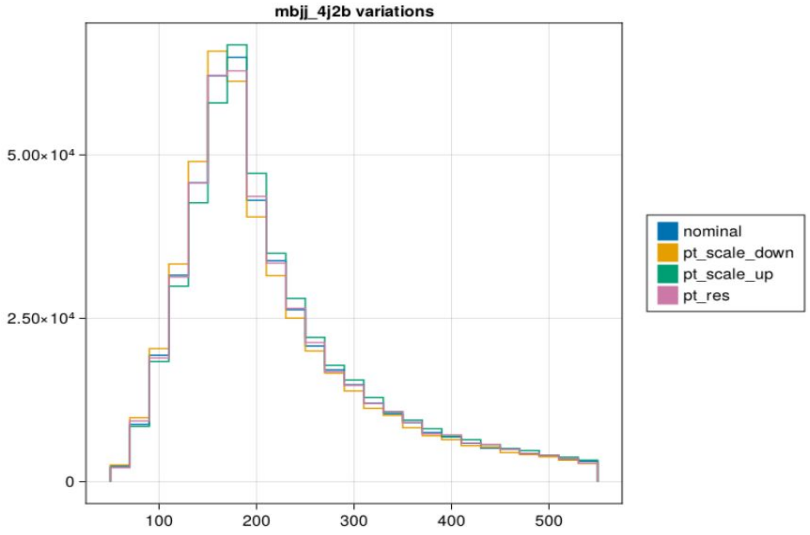
Results

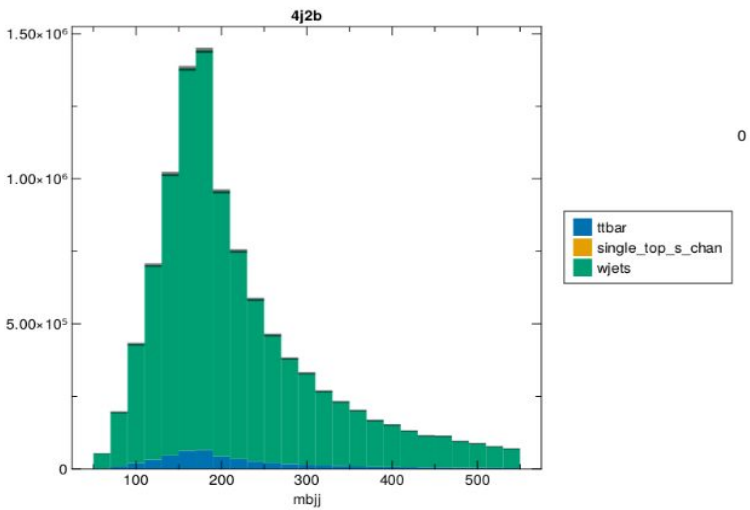
- Distributed version (AF UChicago HTCondor, 25 physical nodes)



Results

- Convenient visualisation tools
- The workspace can be fully exported to a JSON file compatible with Cabinetry/Pyhf
- Found some issues in the reference implementation





Further steps

- Release
- More tests
- Polish the repo, move it to JuliaHEP maybe