



WORKSPACE EXPLORER

pyhf Users and Developers Workshop

CERN

December 4th, 2023

Volker Austrup



MANCHESTER
1824

The University of Manchester

Historical Context

- ▶ Project started as part of combination effort
- ▶ Combination **SHOULD** be straightforward, as all analyses already scrutinised by ATLAS, **BUT ...**
- ▶ Inputs to combination:
 - ▶ produced in **various frameworks**
 - ▶ converted into **JSON** format
 - ▶ need to be understood and **validated**
→ very time-consuming
- ▶ Idea: Streamline validation process to ensure correctness of workspace contents

Examples of issues we saw:

```
{
  "bounds": [
    [
      0.0,
      10.0
    ]
  ],
  "fixed": true,
  "inits": [
    1.0
  ],
  "name": "ATLAS_norm_Z"
},
{
  "bounds": [
    [
      0.0,
      10.0
    ]
  ],
  "fixed": true,
  "inits": [
    1.0
  ],
  "name": "ATLAS_norm_ttbar"
},
```

```
{
  "data": {
    "hi": 1.017,
    "lo": 0.983
  },
  "name": "luminosity",
  "type": "normsys"
},
```

Additional normalisation systematic for luminosity

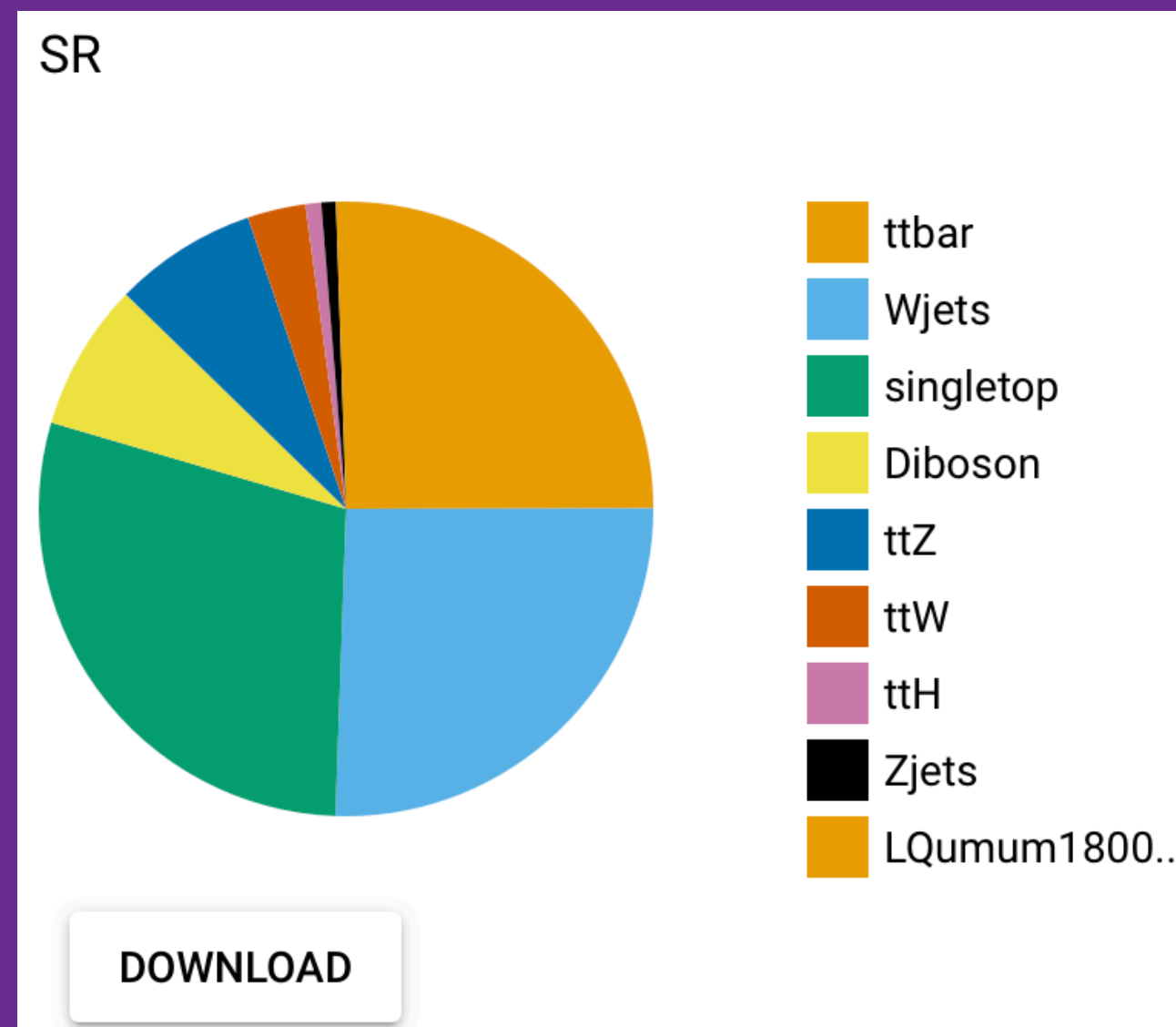
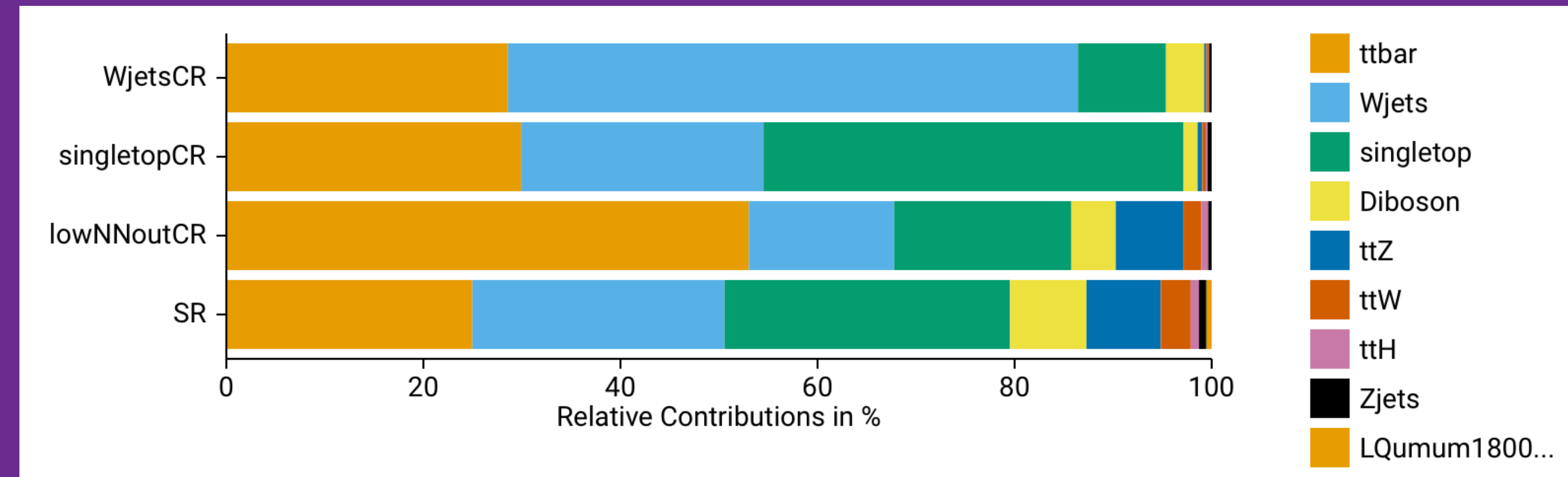
Fit parameters fixed when they should not be

But also cases of certain systematics missing completely!

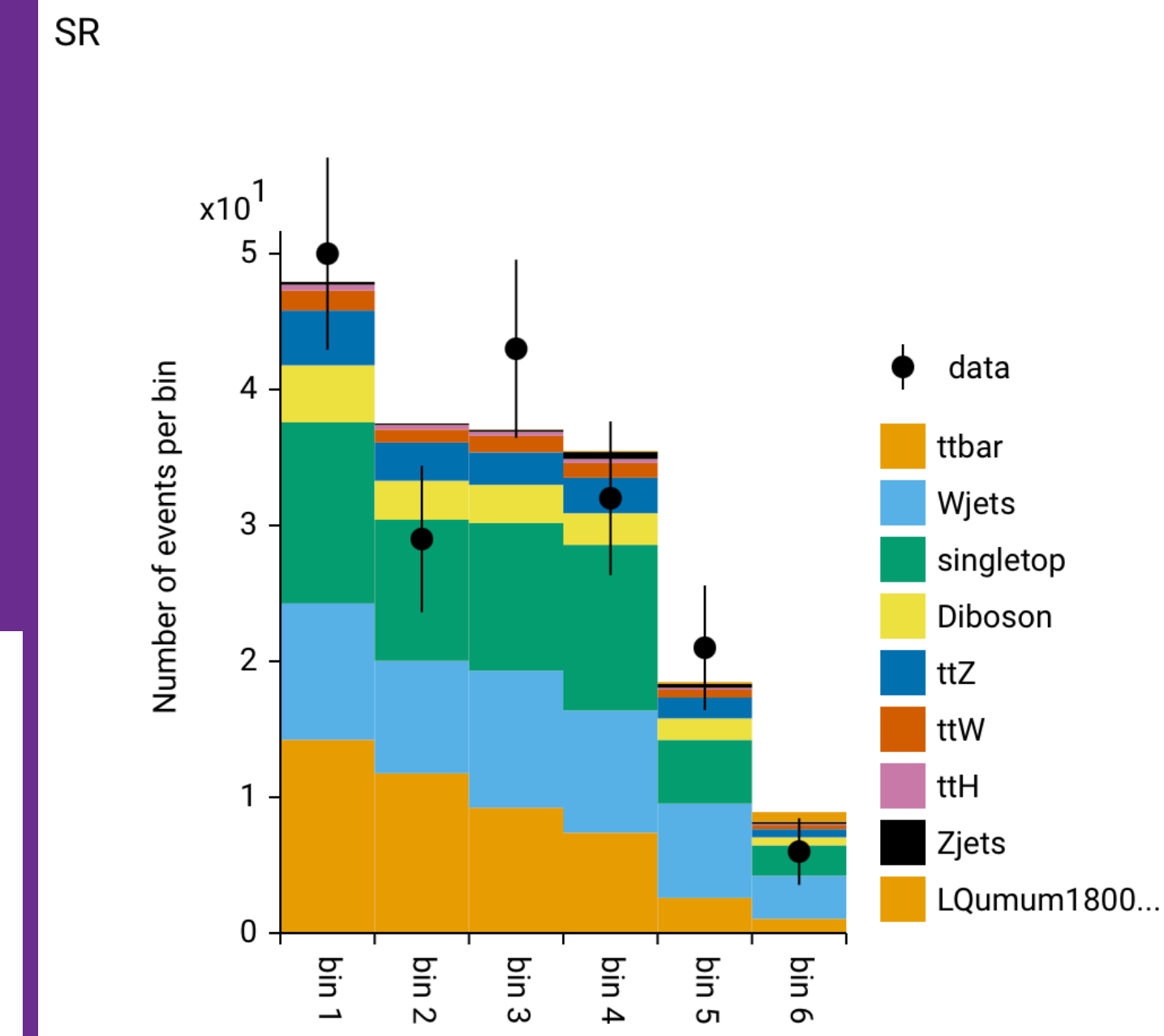
Issues hard to spot when not familiar with workspace

Overview

- ▶ Provide **easy-to-use interface** for analysers to validate their JSON workspaces → [WorkspaceExplorer](#)
- ▶ Runs in web browser, providing **visualisations** of workspace contents:
 - ▶ Bar charts, pie charts, data/MC comparisons, NP structure
- ▶ Connection to python-based backend allows retrieving **fit results**
- ▶ Load workspaces from **local files** or directly from **HEPdata** entries
- ▶ All plots can be downloaded as SVG



DOWNLOAD



Overview



Summary

Processes

ttbar		
Wjets		
singletop		
Diboson		
ttZ		
ttW		
ttH		
Zjets		
LQumum1800BR50		

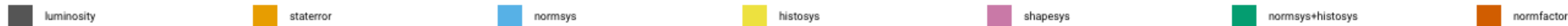
Regions

WjetsCR	
singletopCR	
lowNNoutCR	
SR	

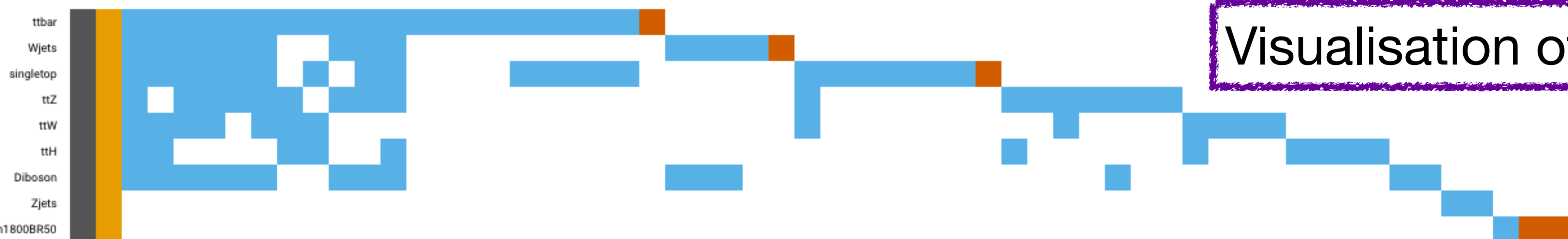
Normalisation Factors

mu_ttbar (floating)
mu_wjets (floating)
mu_singletop (floating)
SigXsecOverSM (floating)
signalnormalization (fixed)

Modifier Structure Chart



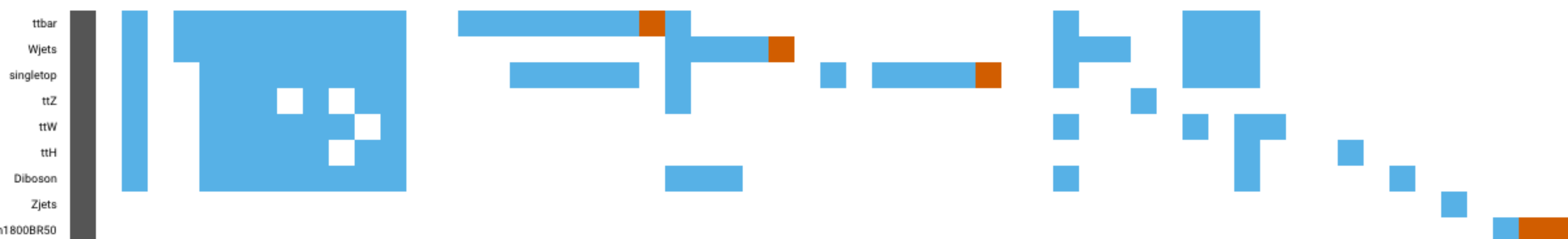
WjetsCR



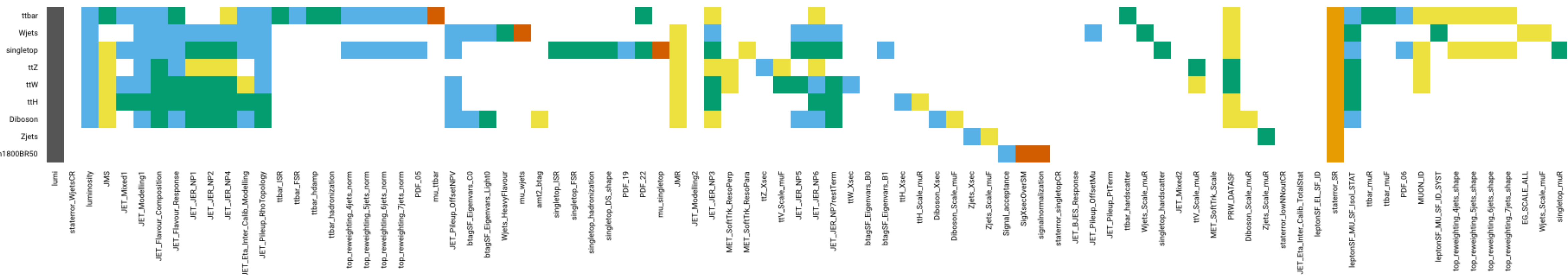
singletopCR



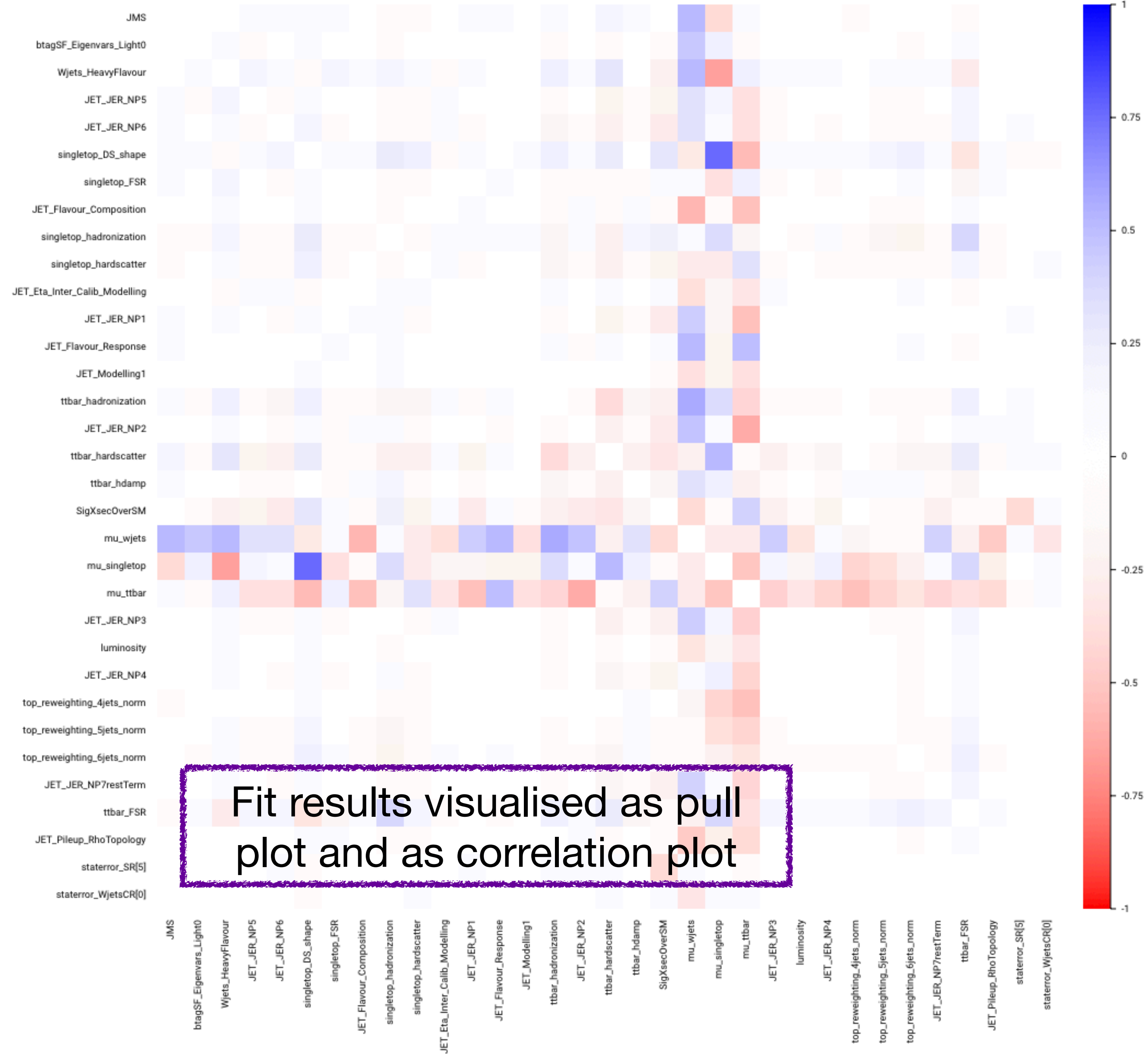
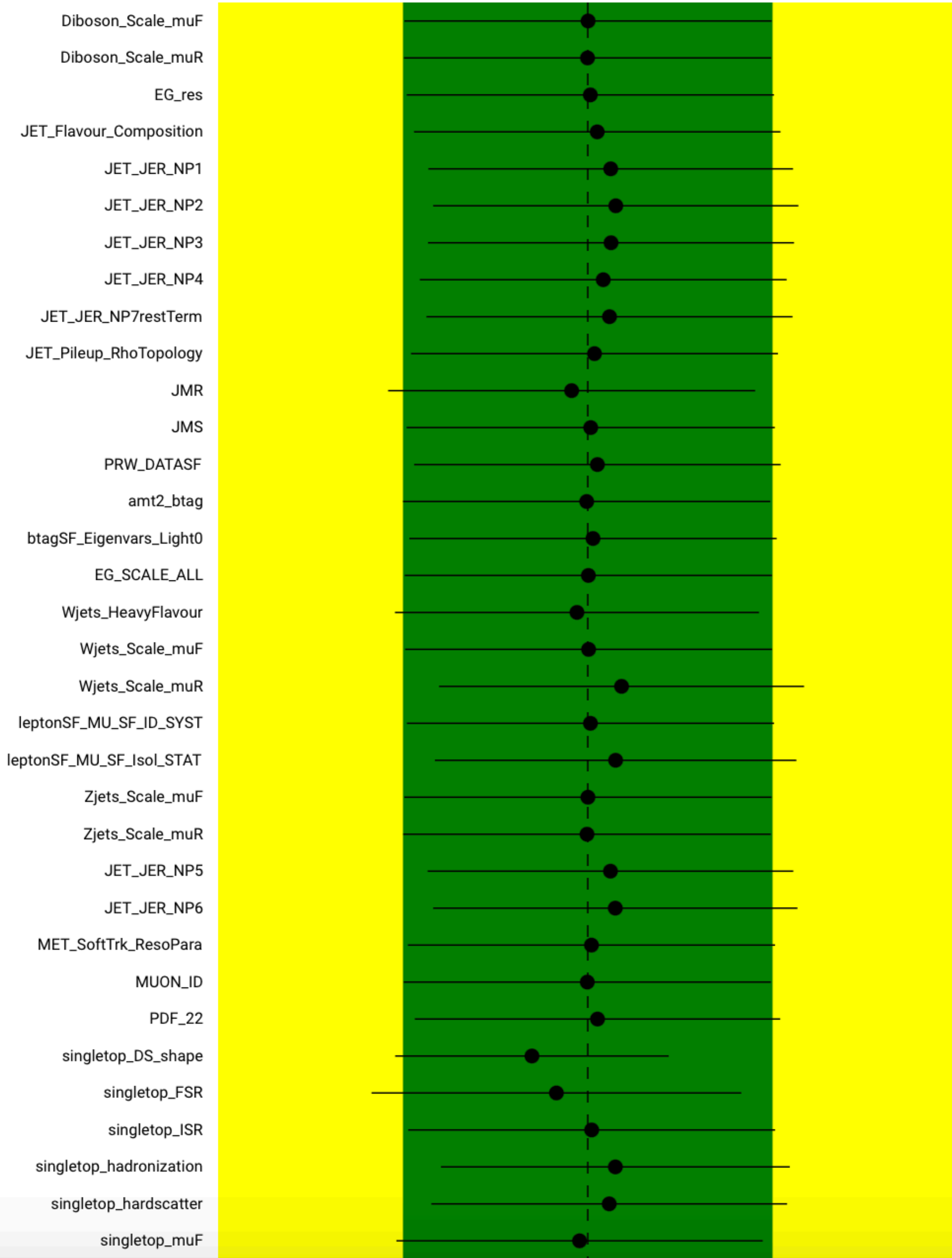
lowNNoutCR



SR



...



Fit results visualised as pull plot and as correlation plot

Application Structure

Frontend written in Typescript
Accessible at

workspaceexplorer.app.cern.ch



“Business logic”:
VueJS framework

Pinia for state
management



Quasar framework
for UI elements

QUASAR

Plots created from native
SVG elements



Send workspaces
to RESTful API

Retrieve fit results
from RESTful API

Code publicly available on Github

[WorkspaceExplorerFrontend](#)
[WorkspaceExplorerBackend](#)



Python-based backend

Hosted on CERN Openshift instance



Flask

Simple RESTful API
using Flask:

- POST workspace
- GET fit results

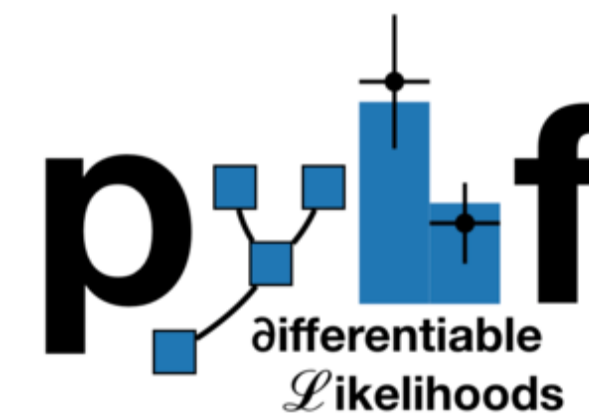


Celery + Redis
for asynchronous
task handling



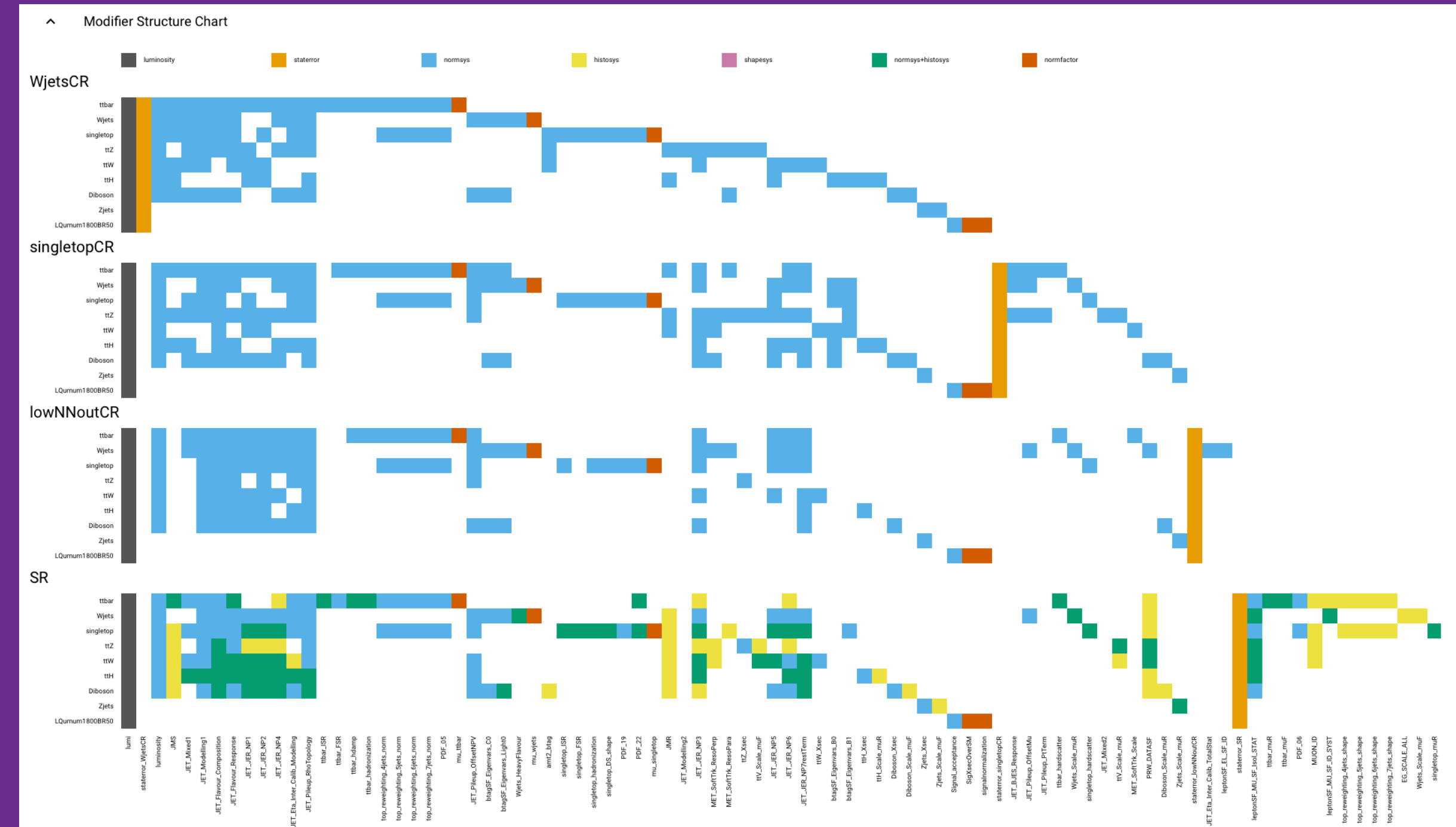
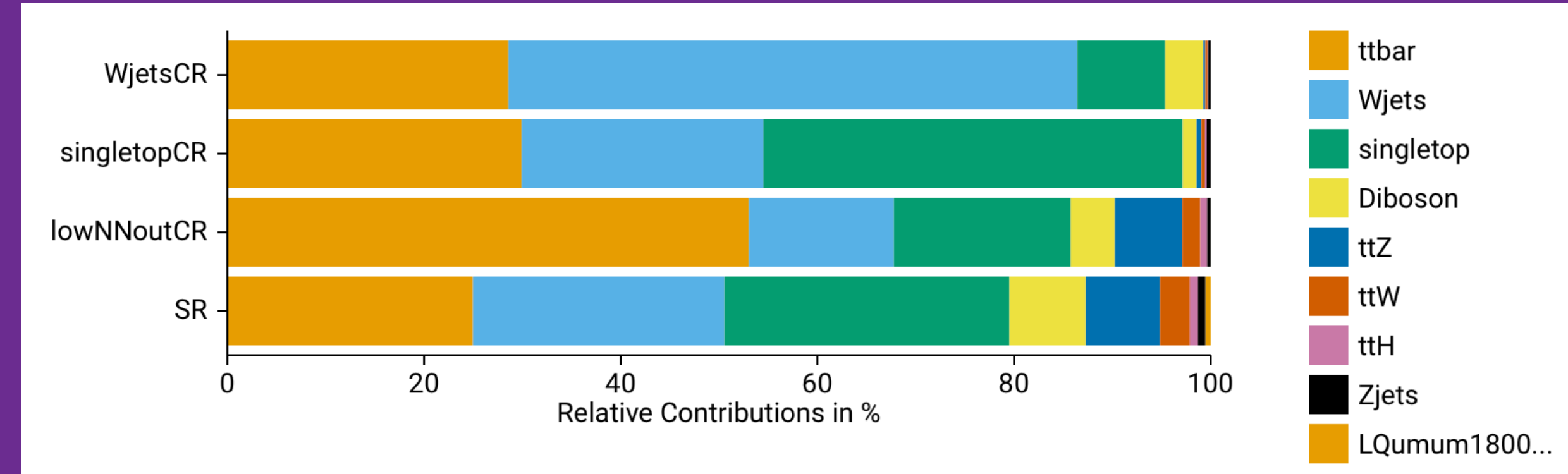
Profile likelihood fits using

pyhf + cabinetry



Usage Scenarios

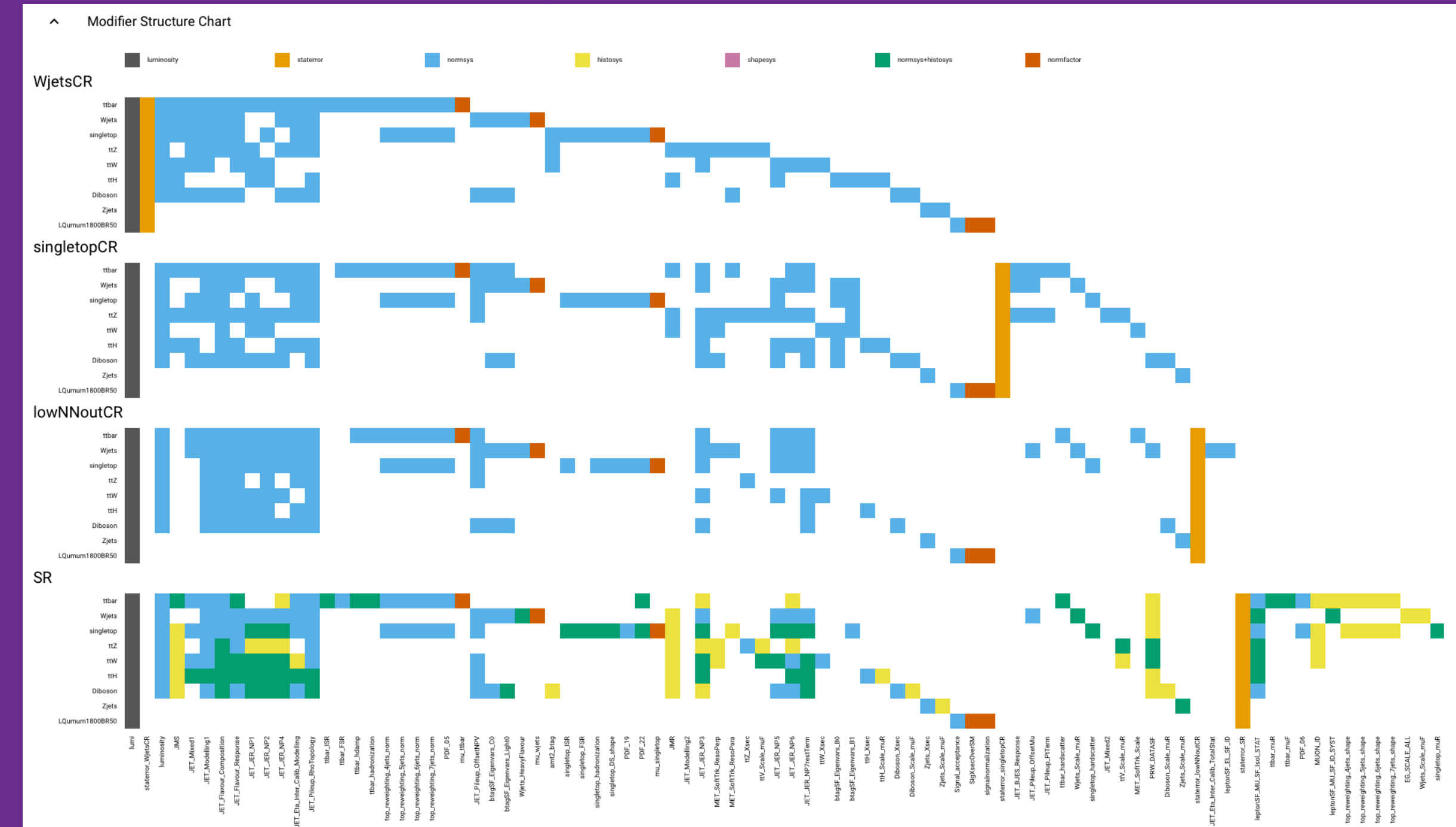
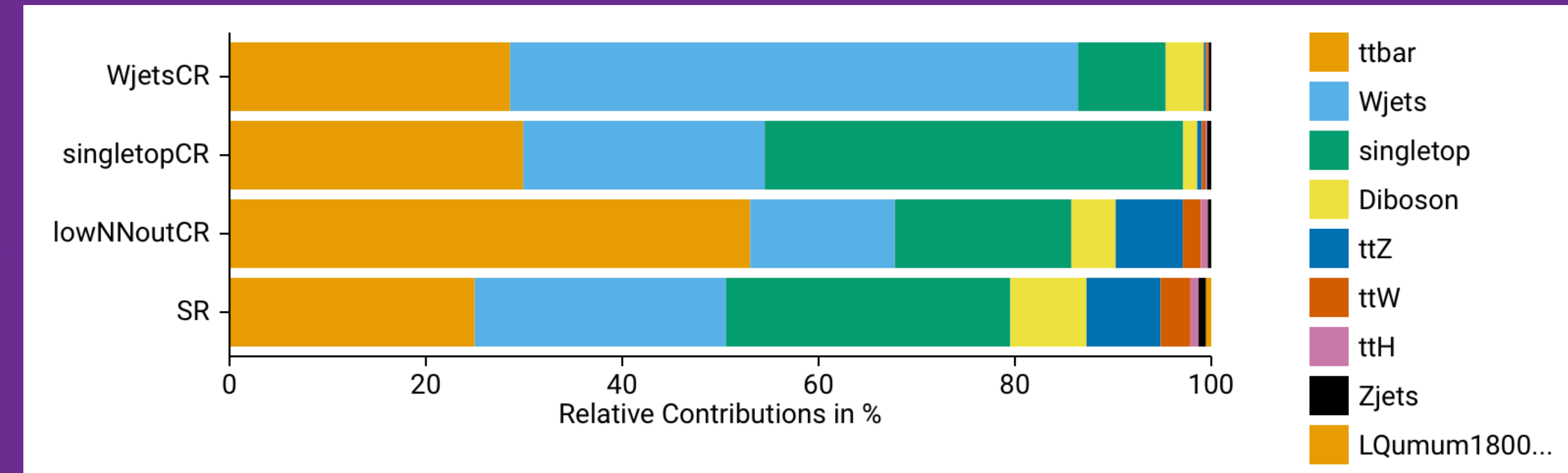
- ▶ **Validation** of workspaces:
 - ▶ easy way for analysers to confirm contents of workspaces
 - ▶ avoid mistakes due to conversion between different statistical frameworks
 - ▶ confirm all necessary information is available
- ▶ **Exploration** of unfamiliar workspaces:
 - ▶ aid understanding of analyses from outsider's perspective
- ▶ **Education**:
 - ▶ introduce newcomers to key concepts



Advantages

- ▶ Parallels to RooBrowser, but:
 - ▶ focus on HistFactory schema
 - ▶ no need to setup ROOT
 - ▶ runs natively in web browser
- low barrier to entry
- ▶ Can tie in directly with HEPdata:
 - ▶ load via ID of HEPdata entry
 - ▶ share via URL parameter, e.g.

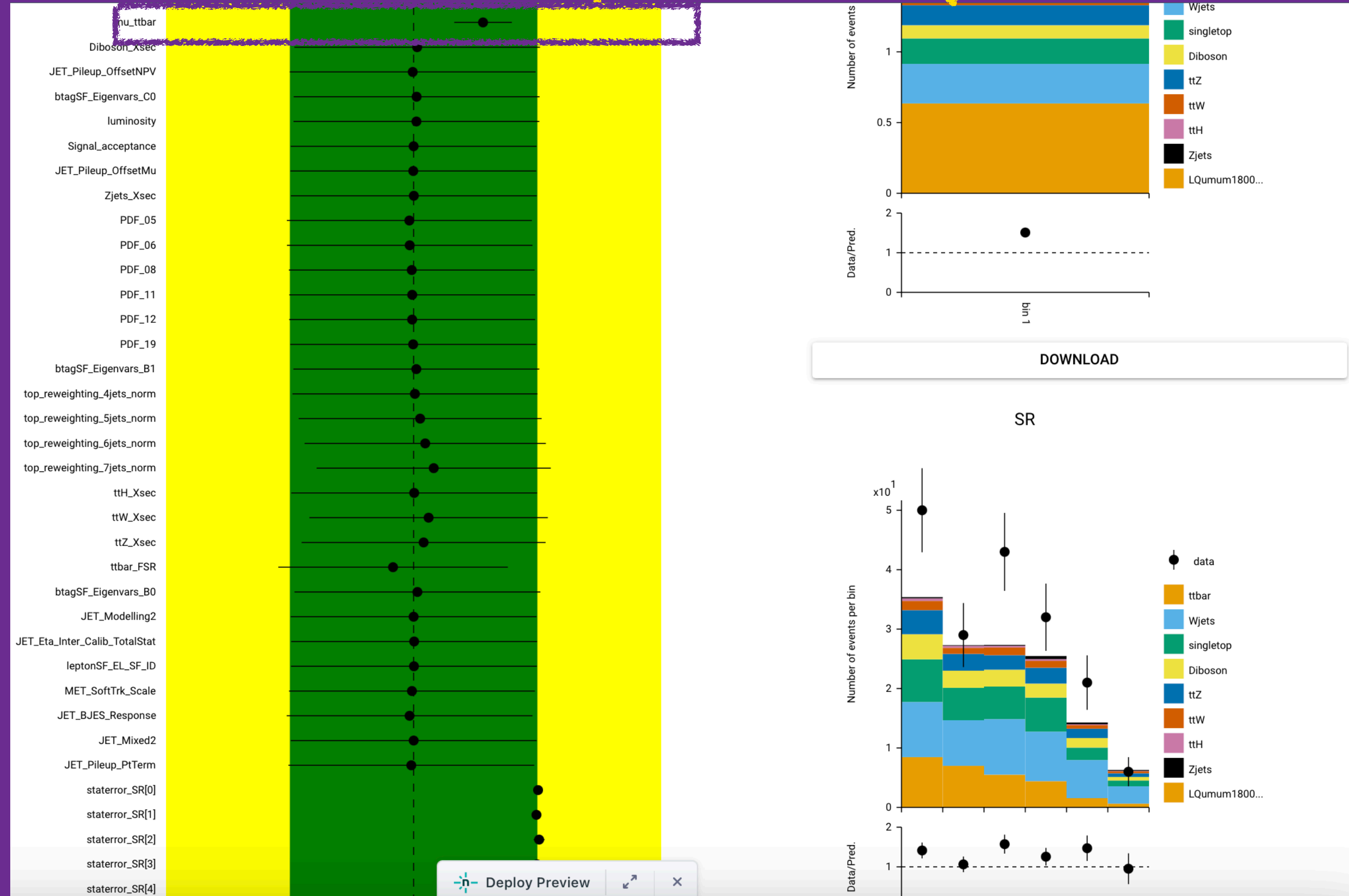
<https://workspaceexplorer.app.cern.ch/?id=2077557>



Interactive Pulls

- ▶ New, experimental feature: “pulling the pulls”
- ▶ Change pulls directly in the UI
- ▶ Work in progress, but preview to play around with [here](#)
- ▶ Investigate impact of nuisance parameters → “Fitting by hand”

Changing pulls in pull plot instantly reflected in post-fit plots



Interactive Pulls

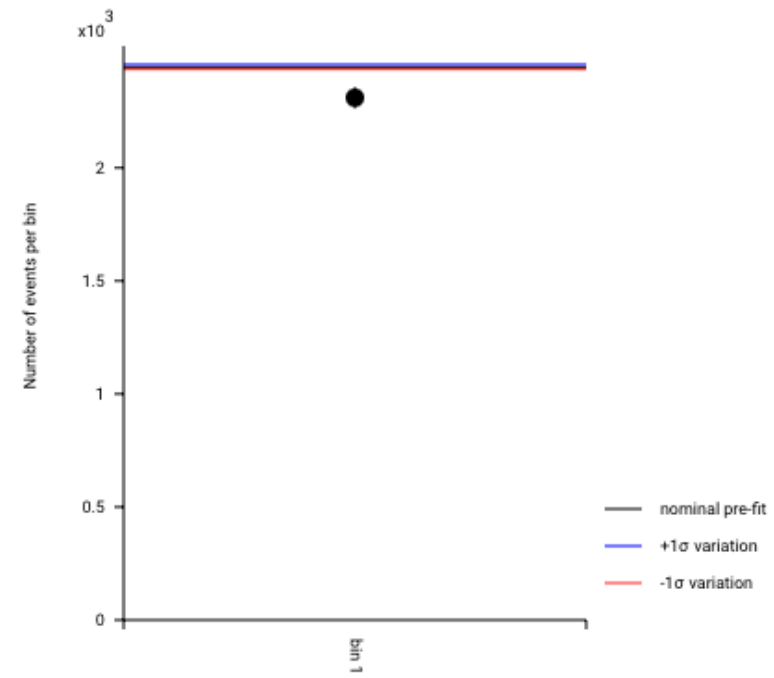
Systematic Uncertainty Chart

Filter by name
JER|

1 / 2

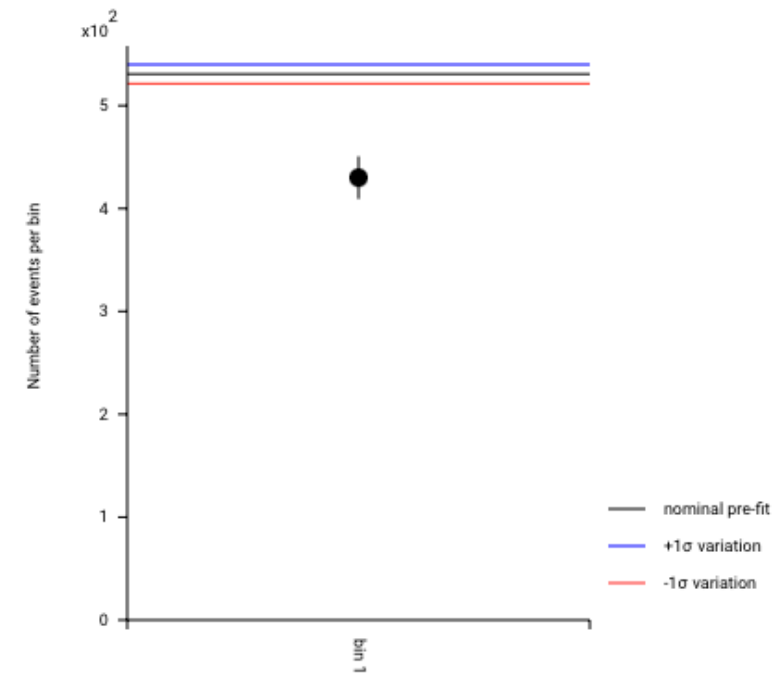
JET_JER_NP1

WjetsCR



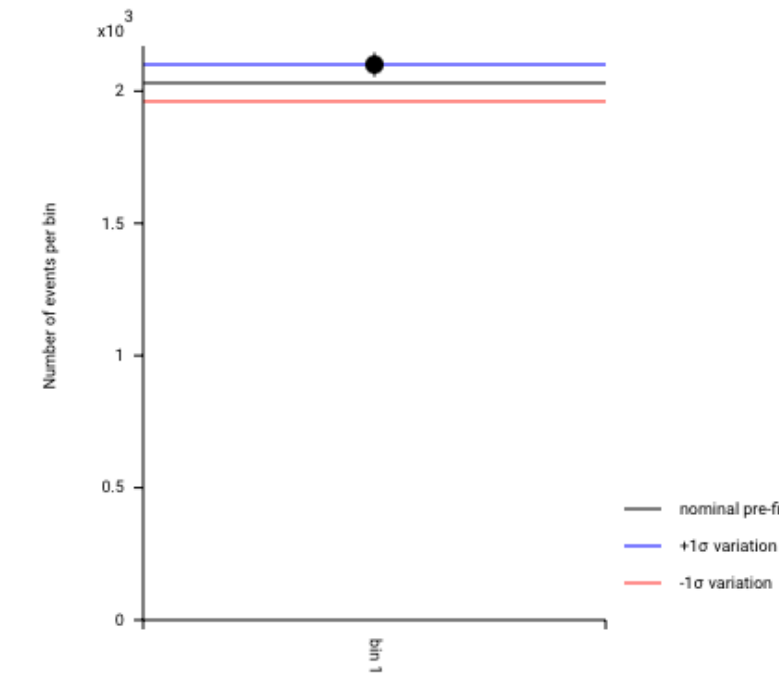
DOWNLOAD

singletopCR



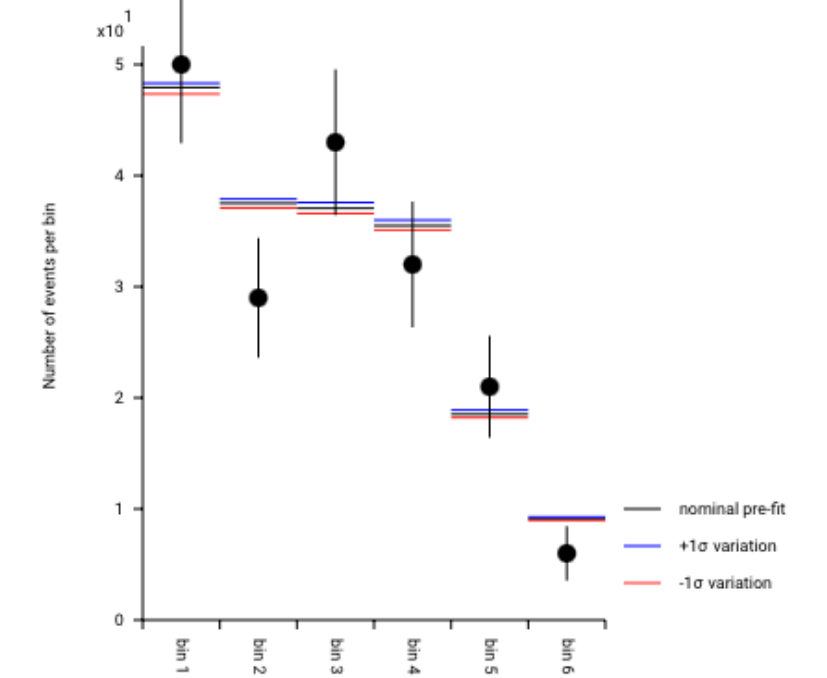
DOWNLOAD

lowNNoutCR



DOWNLOAD

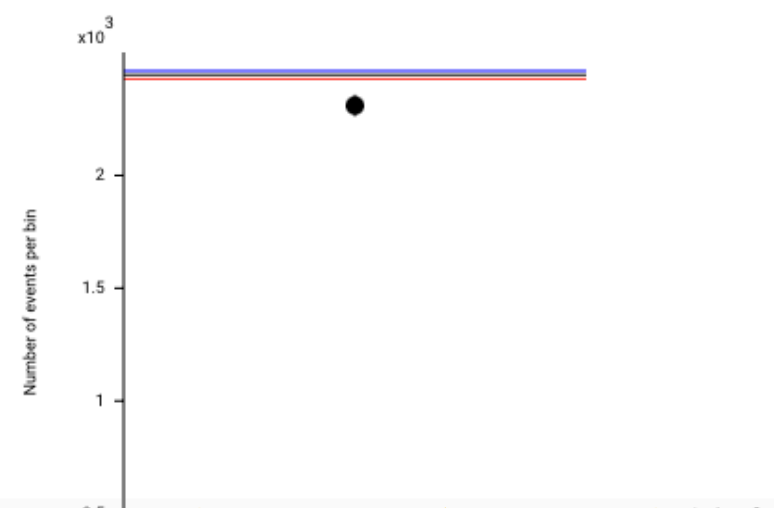
Signal Region



DOWNLOAD

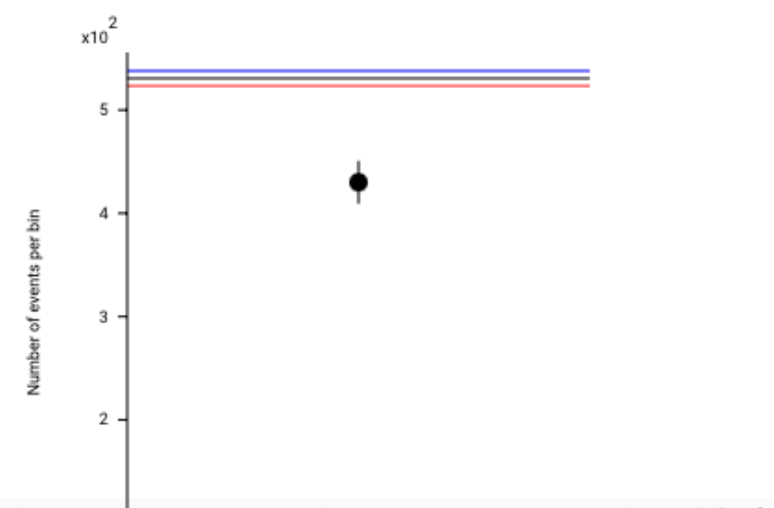
JET_JER_NP2

WjetsCR



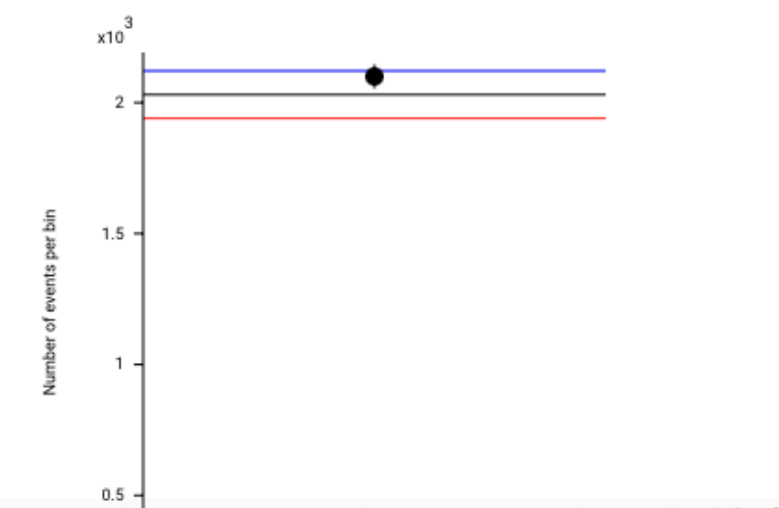
DOWNLOAD

singletopCR



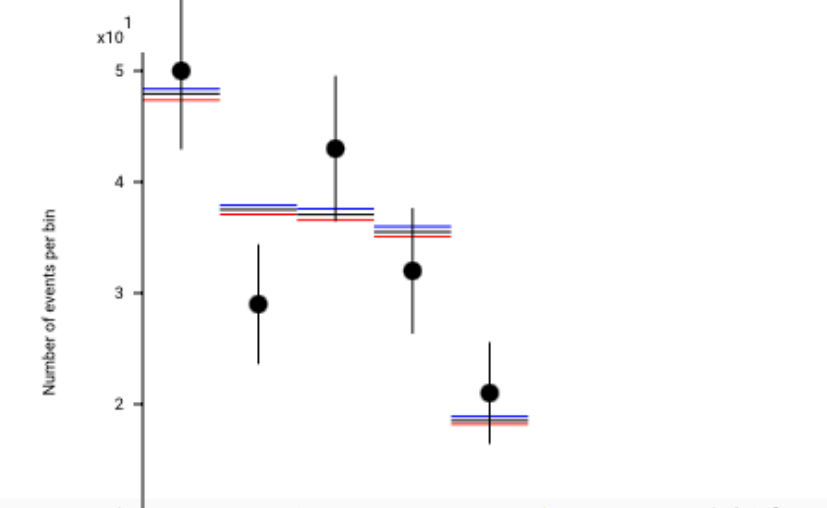
DOWNLOAD

lowNNoutCR



DOWNLOAD

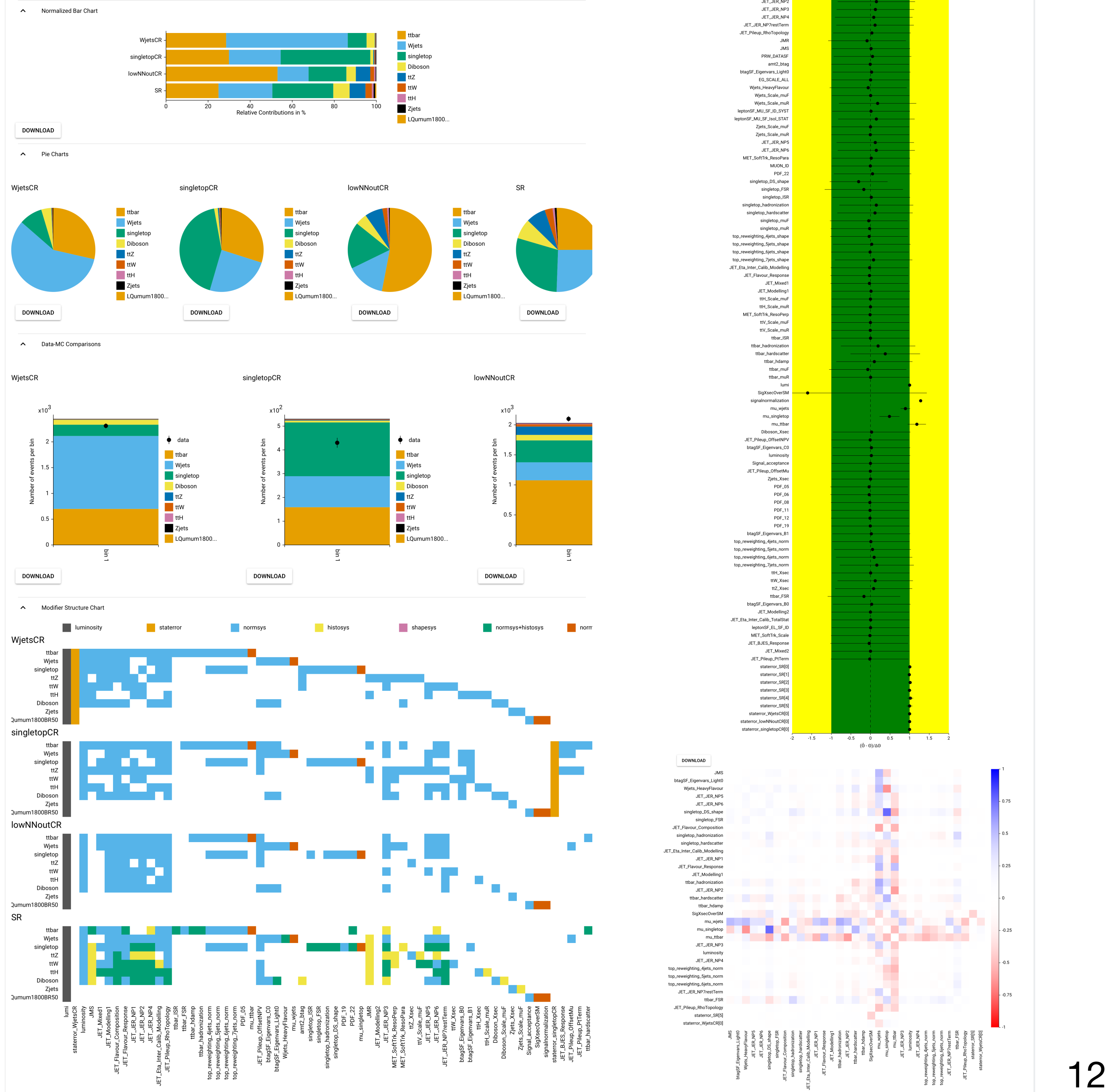
Signal Region



DOWNLOAD

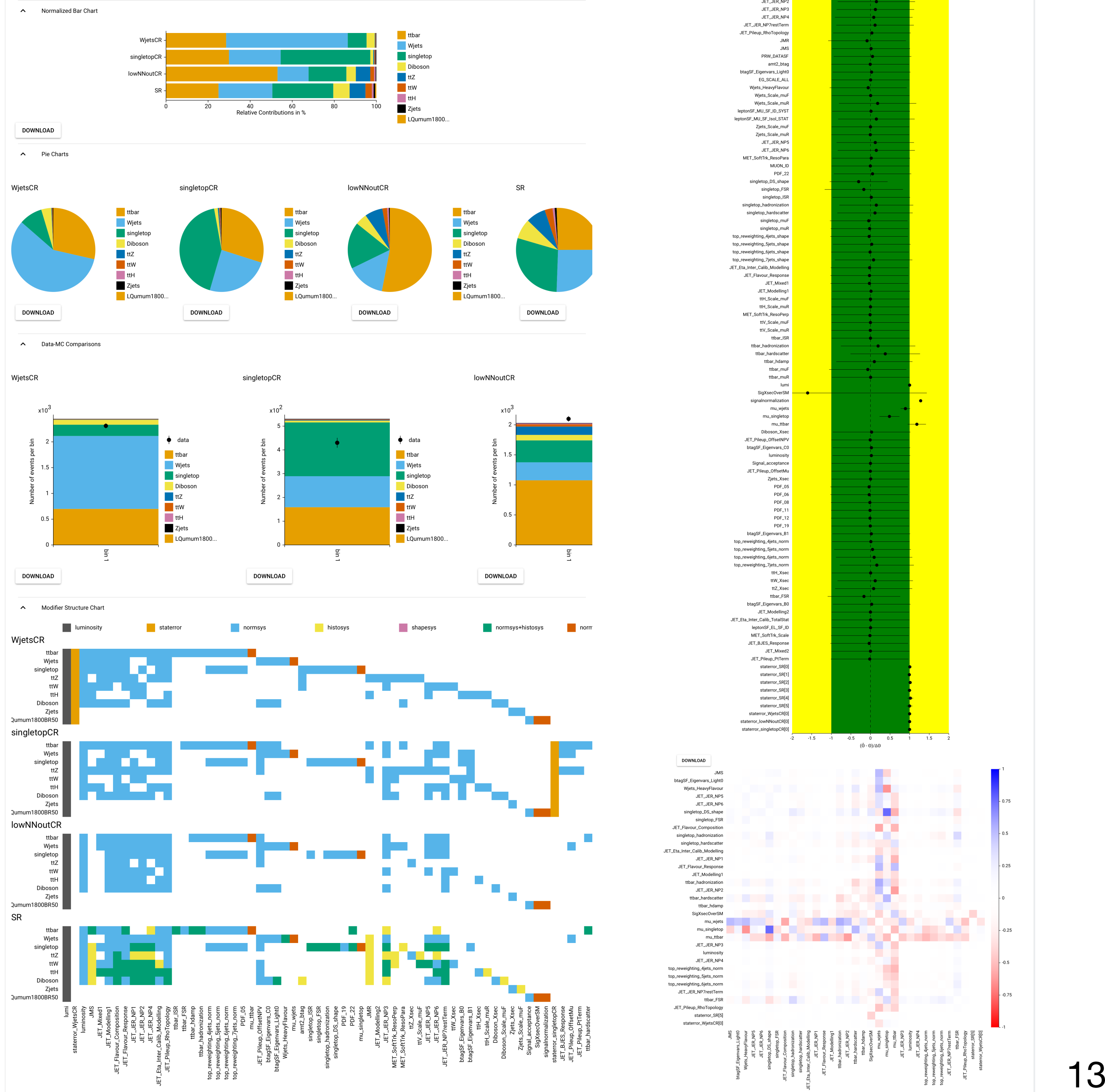
Conclusions & Outlook (I)

- ▶ Interactive visualisation of HistFactory workspaces in the web browser
- ▶ Read in local files or load directly from HEPdata
- ▶ Connection to backend running **pyhf + cabinetry** for profile-likelihood fits
- ▶ Useful for understanding statistical models, but also for sharing them





Conclusions & Outlook (II)

- ▶ In Progress:
 - ▶ interactive pulls
 - ▶ plots of systematic variations
 - ▶ Ideas for the future
 - ▶ Cache workspaces and fit results on server to allow sharing via permalink
 - ▶ Customisable fit parameters (e.g. exclude certain NPs)
 - ▶ ...



SimpleCombination

- ▶ Recently made code for combination effort public: [SimpleCombination](#)
- ▶ Based on  and 
- ▶ Easily combine workspaces by providing configurations for input analyses and overall combination

```
from common.analysisbase import AnalysisBase

class Analysis(AnalysisBase):
    def filename(self):
        return f"test/analysis1_M{self.parameters['mass']}GeV.json"

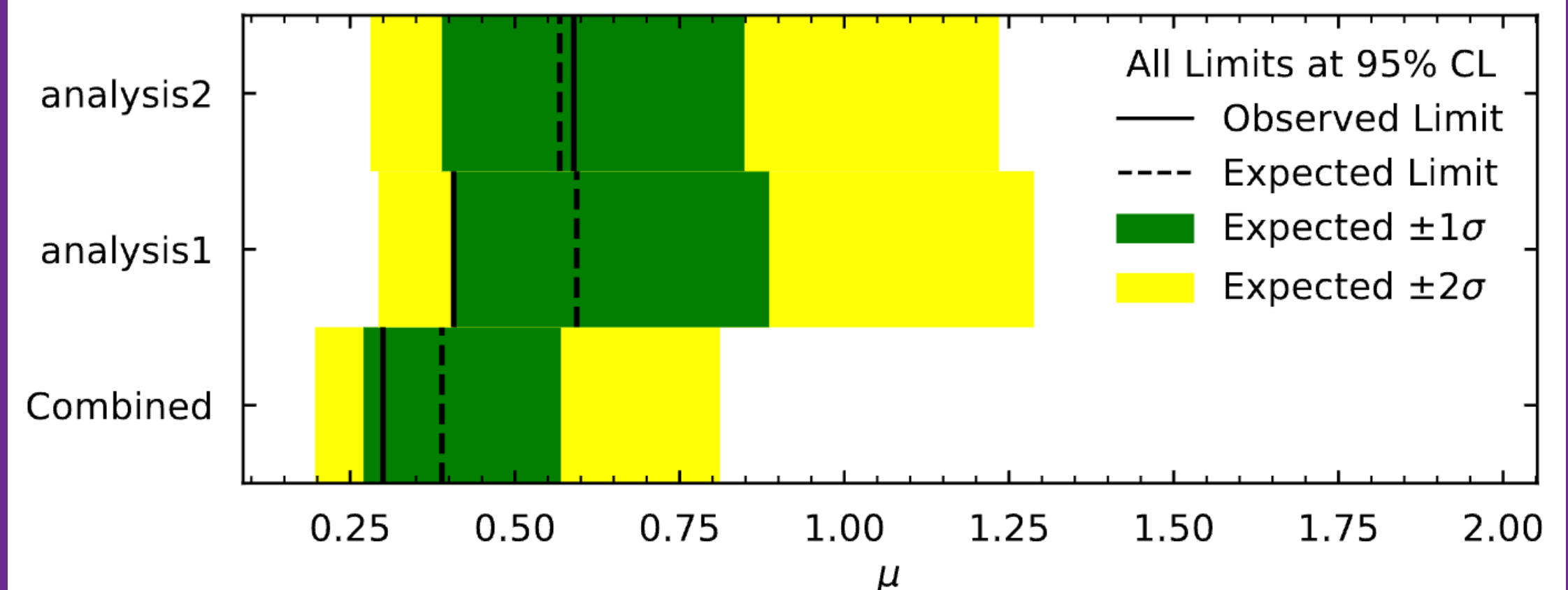
    def signalname(self):
        return f"signal_M{self.parameters['mass']}GeV"
```

```
from common.combinationbase import CombinationBase

class Combination(CombinationBase):
    channels = {
        "analysis1": {
            "SR": "SR",
            "CR1": "CR bkg 1",
            "CR2": "CR bkg 2",
        },
        "analysis2": {
            "SR": "SR",
            "CR1": "CR bkg 1",
            "CR2": "CR bkg 2",
        },
    }

    measurement_parameters = {
        "SigXsecOverSM": {"bounds": [[0, 5]], "inits": [0.0], "fixed": False},
        "lumi": {"sigmas": [0.017], "bounds": [[0.915, 1.085]], "fixed": False},
    }

    correlated_NPs = {
        "normsys2": {"analysis1": "normsys2", "analysis2": "normsys2"},
        "histosys1": {"analysis1": "histosys1", "analysis2": "histosys1"},
    }
```



Thank you!

