RECAST

Lukas Heinrich

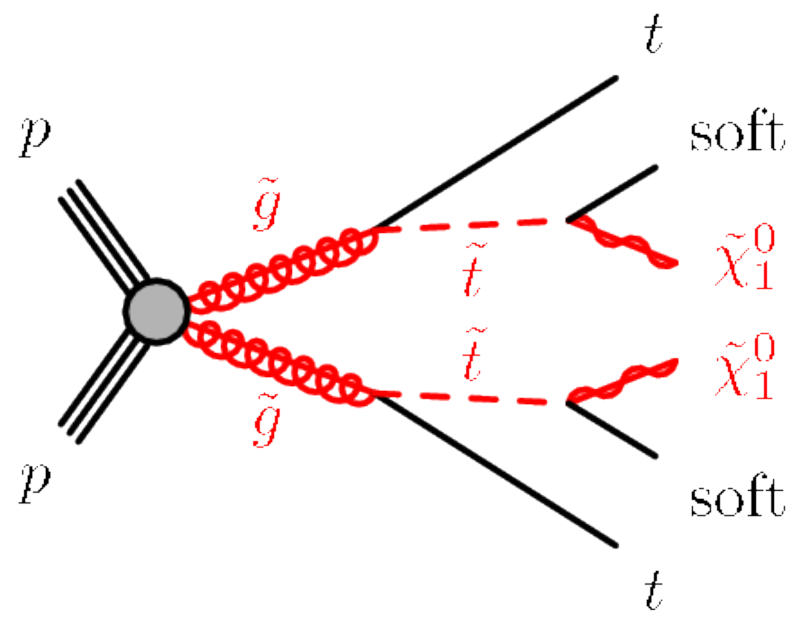
LHCb Computing Workshop

NEW YORK UNIVERSITY

Searching for New Physics in ATLAS

Typical ATLAS Search for BSM Physics

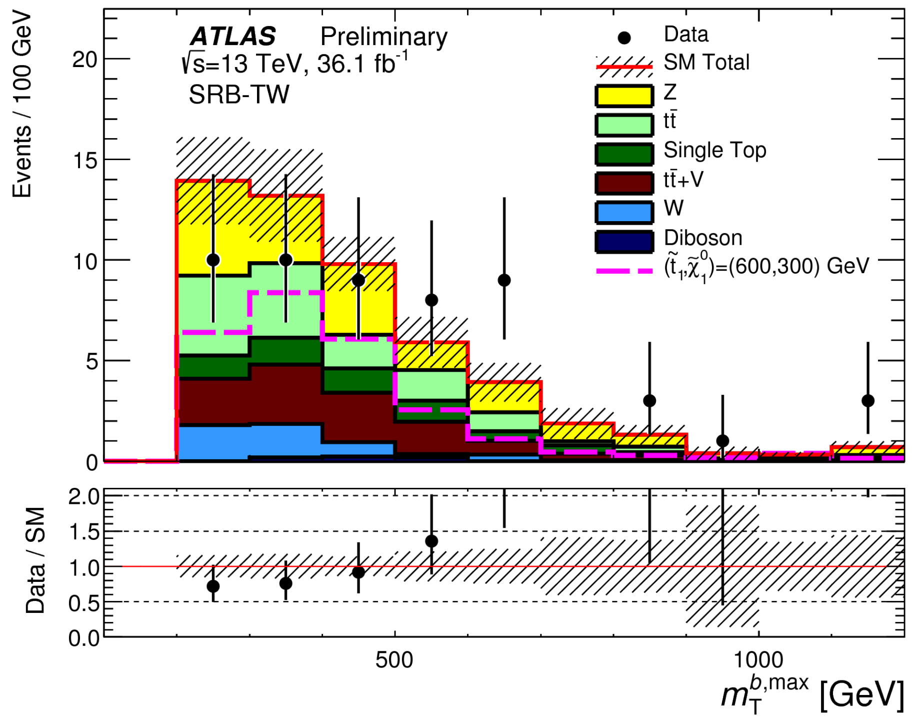
1) Take a Model of BSM Physics

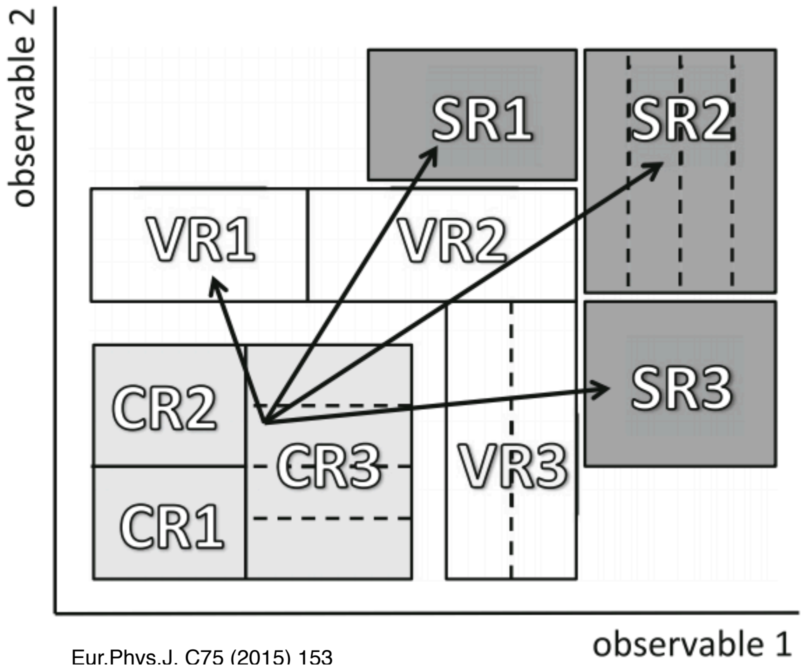


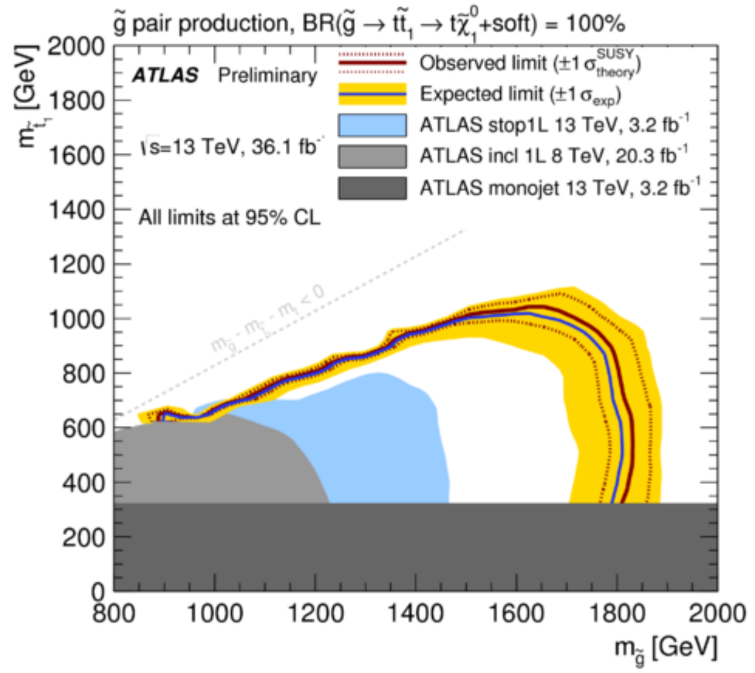
typically with some free parameters p1, … pn. Here gluino, **m_g̃**, and stop masses **m_t̃.**

2) Measure Data and estimate SM backgrounds

mix of data-driven and simulation-based (MC) background estimates. Signal from simulation.







3) set limits in space of free model parameters (m_g̃, m_t̃)

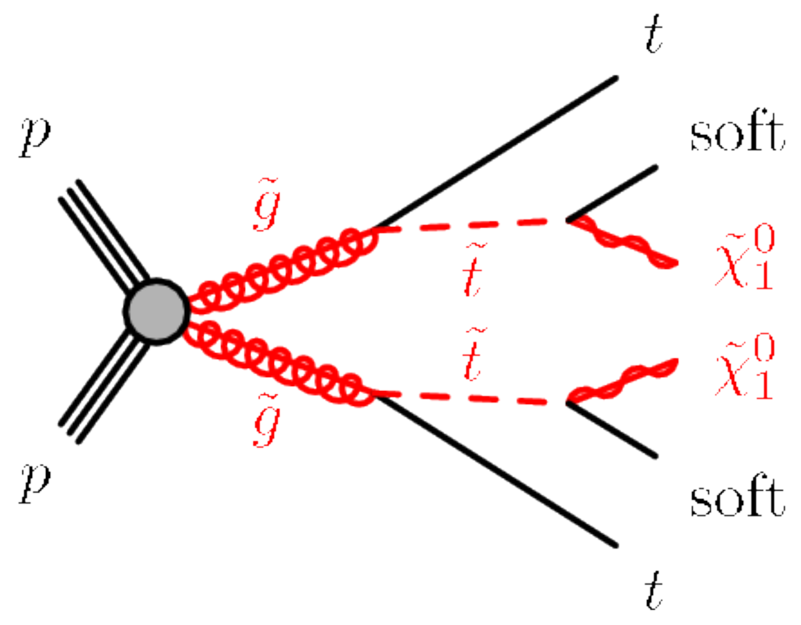
plots from ATLAS-CONF-2017-020

Typical ATLAS Search for BSM Physics

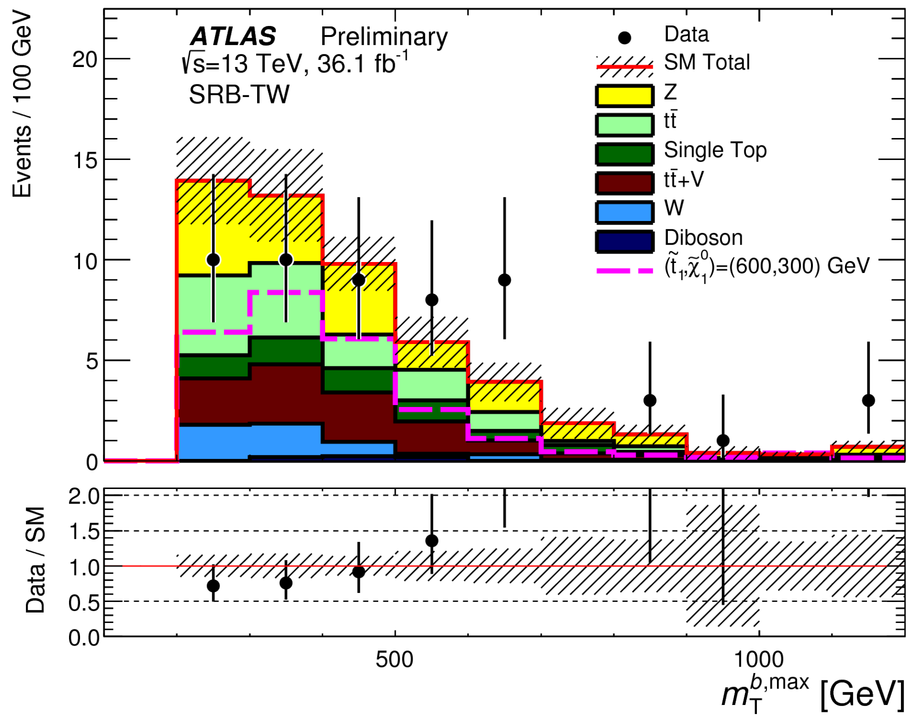
2) Measure Data and estimate SM backgrounds

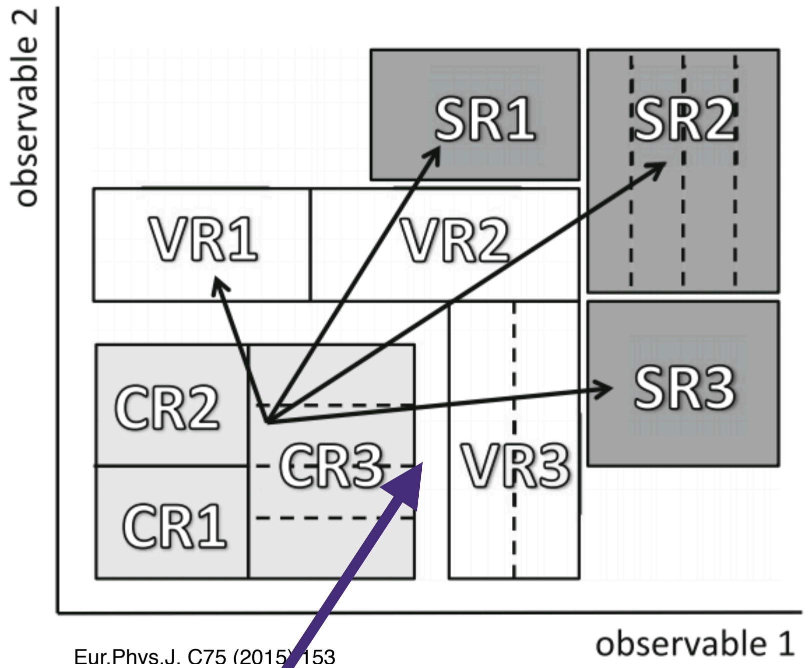
mix of data-driven and simulation-based (MC) background estimates. Signal from simulation.

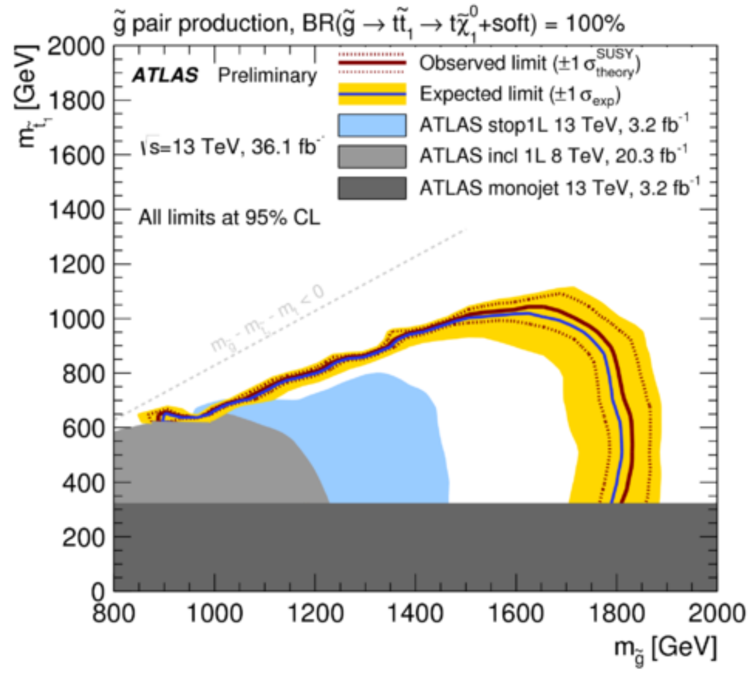
1) Take a Model of BSM Physics



typically with some free parameters p1, … pn. Here gluino, **m_g̃**, and stop masses **m_t̃.**







We are not measuring full phase space of final state particles. Acceptance into Signal Region is model-dependent

3) set limits in space of free model parameters (m_g̃, m_t̃)

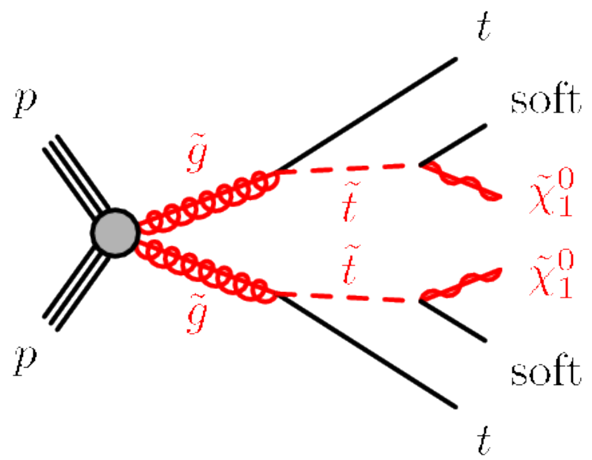
NEW YORK UNIVERSITY

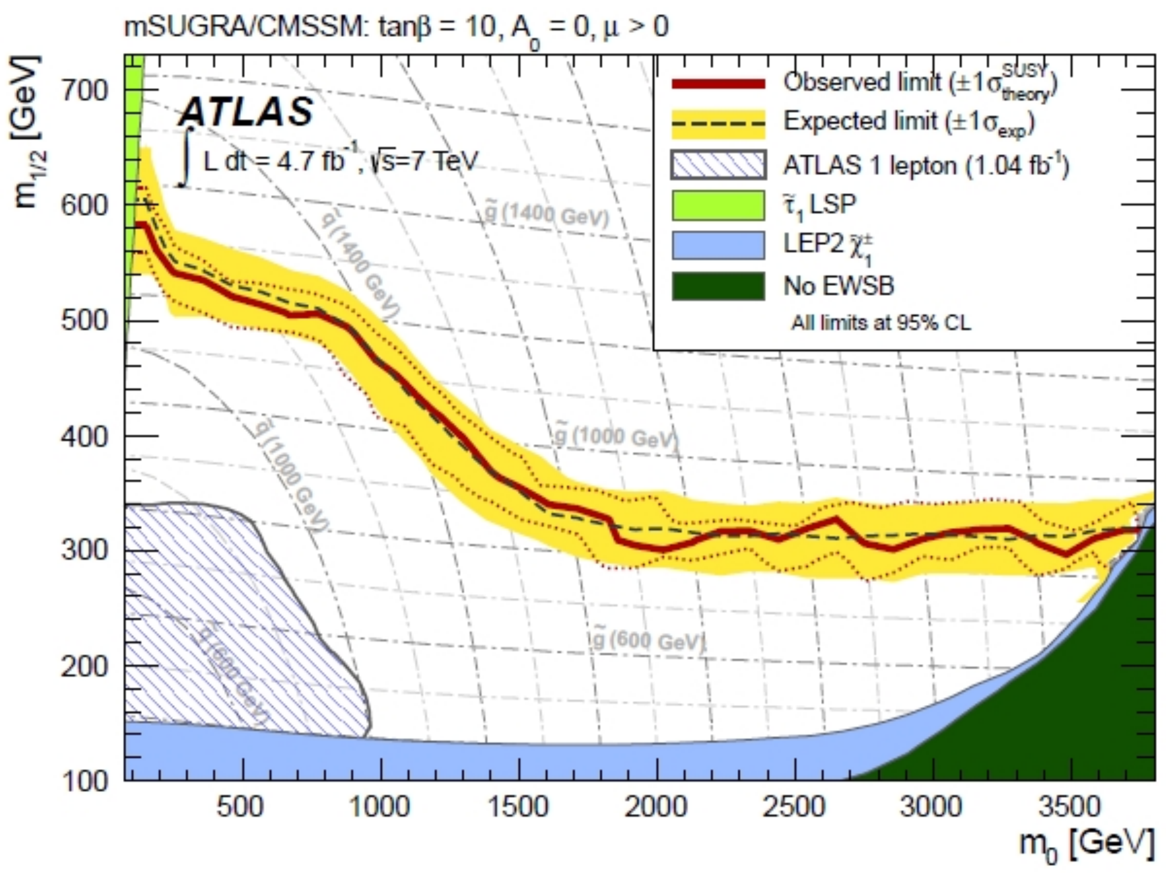
plots from ATLAS-CONF-2017-020

Typical ATLAS Search for BSM Physics

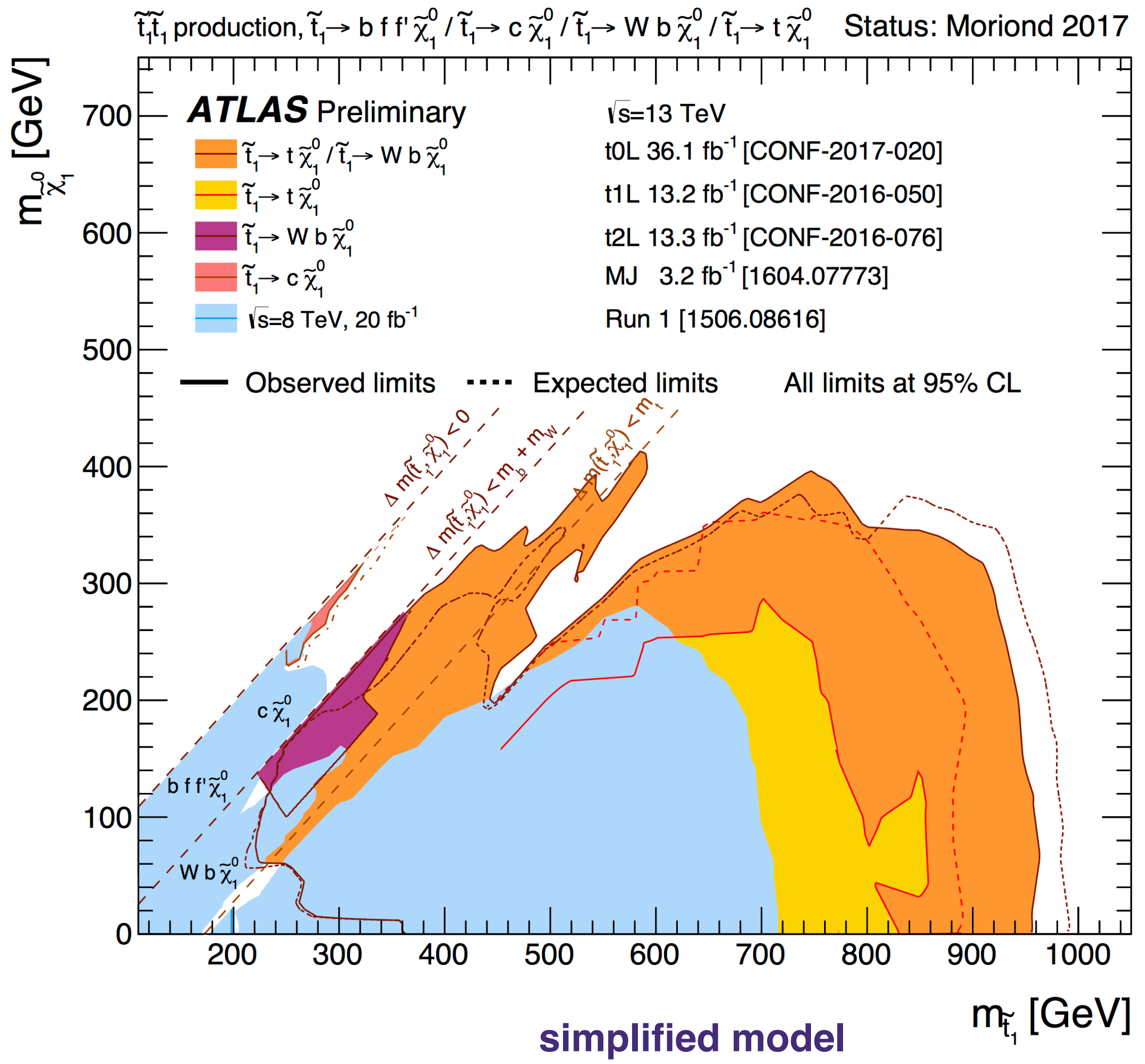
ATLAS has moved away from UV-complete models in favor of bottom-up ‘simplified models’:

- limited particle content focused on specific signature (e.g. tops + MET)
- reinterpretation for full model:
  - calculate BR into topology of simplified model
  - hard if not any one topology dominant. limits weaker for full theories than for simplified models





UV-complete model



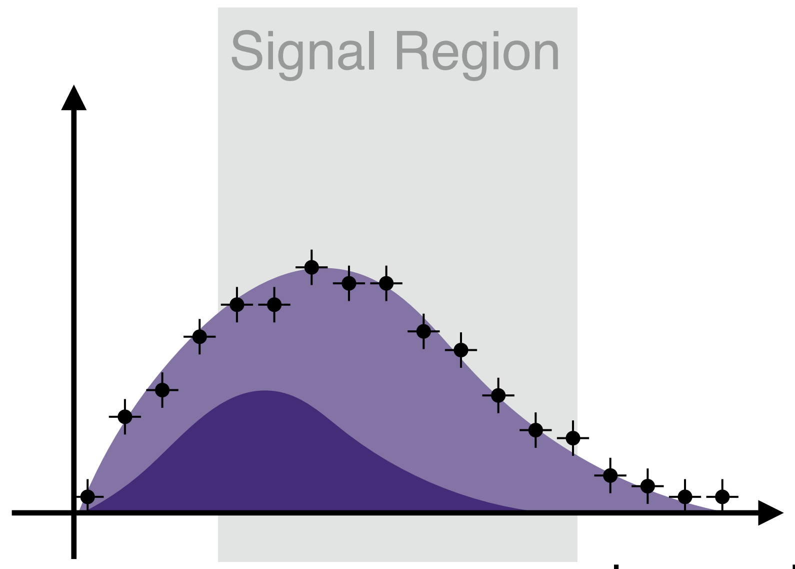
simplified model

The analyses we prepare at the LHC are **high-effort, expensive projects**: non-trivial amount of person-power, time, and computing resources devoted to achieving a publication-quality result.

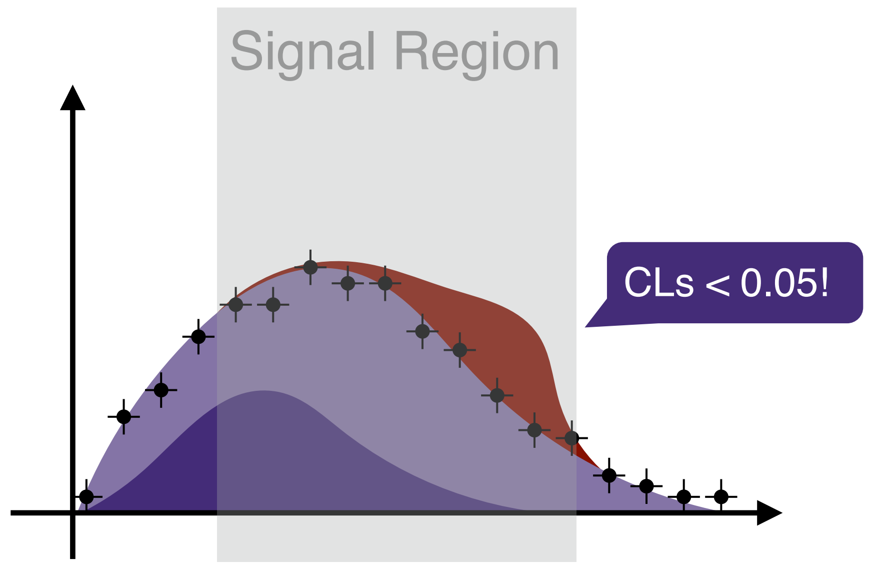
Most of the work goes into: **taking data, designing**, **validating** the analysis strategy, **understanding Standard Model backgrounds.** Effectively: a measurement of observed and backgrounds in interesting phase space regions.

Model interpretation come at the end, and are technically the **easiest part:** analysis pipeline is **fixed** after unblinding, MC dataset sizes small. Analysis teams routinely check hundreds of parameter points (of their favorite model).

But: most analyses only **interpreted once** within limited set of models.
- analysis team pushing for conference deadline
- interesting models proposed by hep-ph *after* they've seen the paper / note.



observed **data** and
estimated backgrounds



interpreted w.r.t **model A**

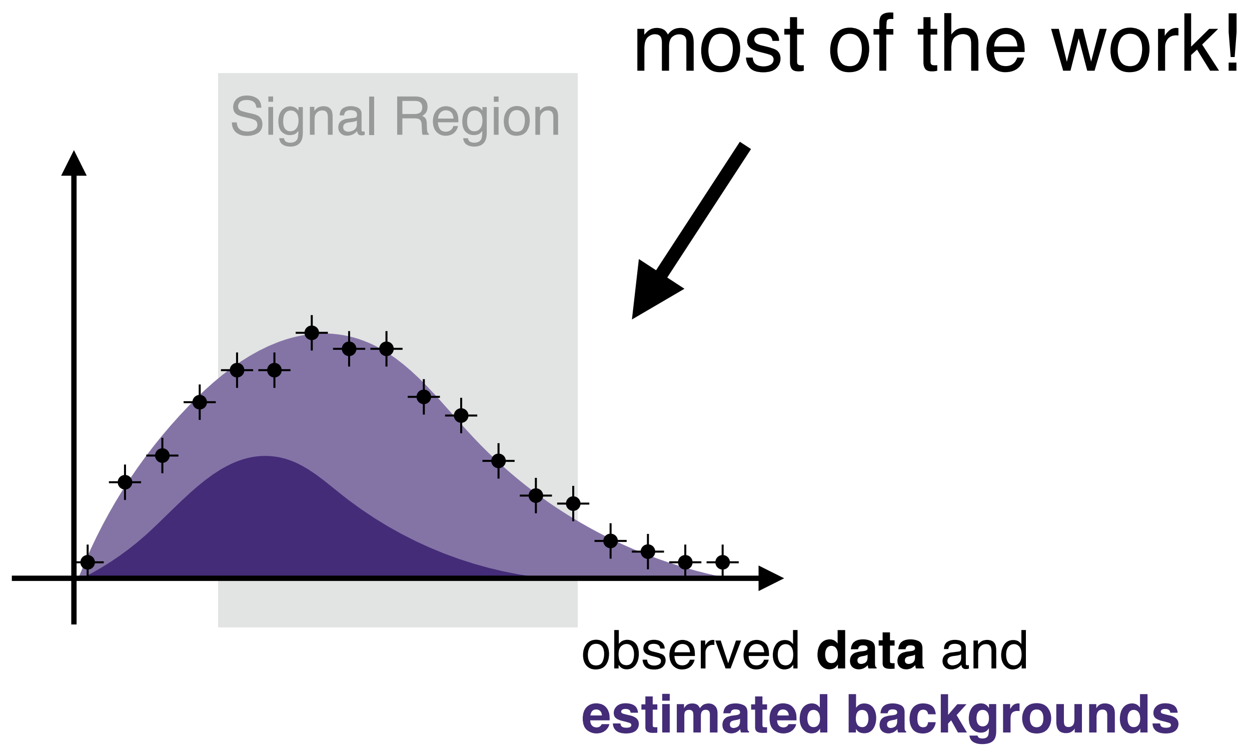
The analyses we prepare at the LHC are **high-effort, expensive projects**: non-trivial amount of person-power, time, and computing resources devoted to achieving a publication-quality result.

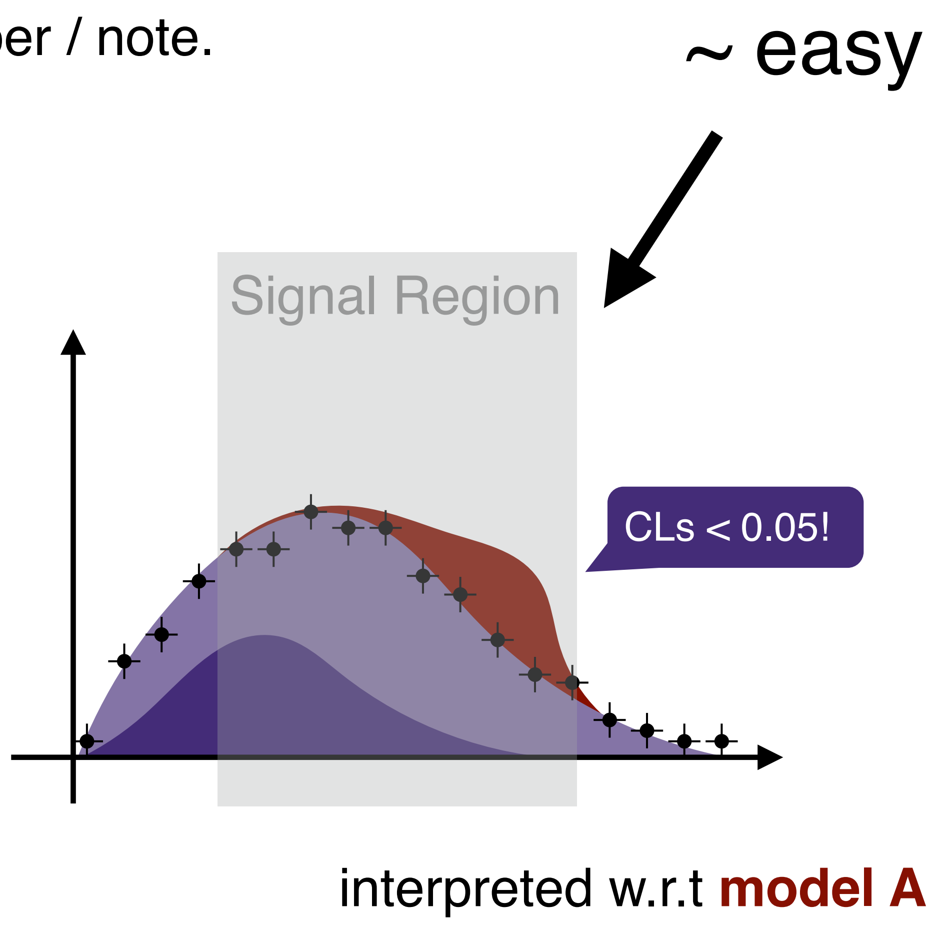
Most of the work goes into: **taking data, designing**, **validating** the analysis strategy, **understanding Standard Model backgrounds.** Effectively: a measurement of observed and backgrounds in interesting phase space regions.

Model interpretation come at the end, and are technically the **easiest part:** analysis pipeline is **fixed** after unblinding, MC dataset sizes small. Analysis teams routinely check hundreds of parameter points (of their favorite model).

But: most analyses only **interpreted once** within limited set of models.

- analysis team pushing for conference deadline
- interesting models proposed by hep-ph *after* they've seen the paper / note.



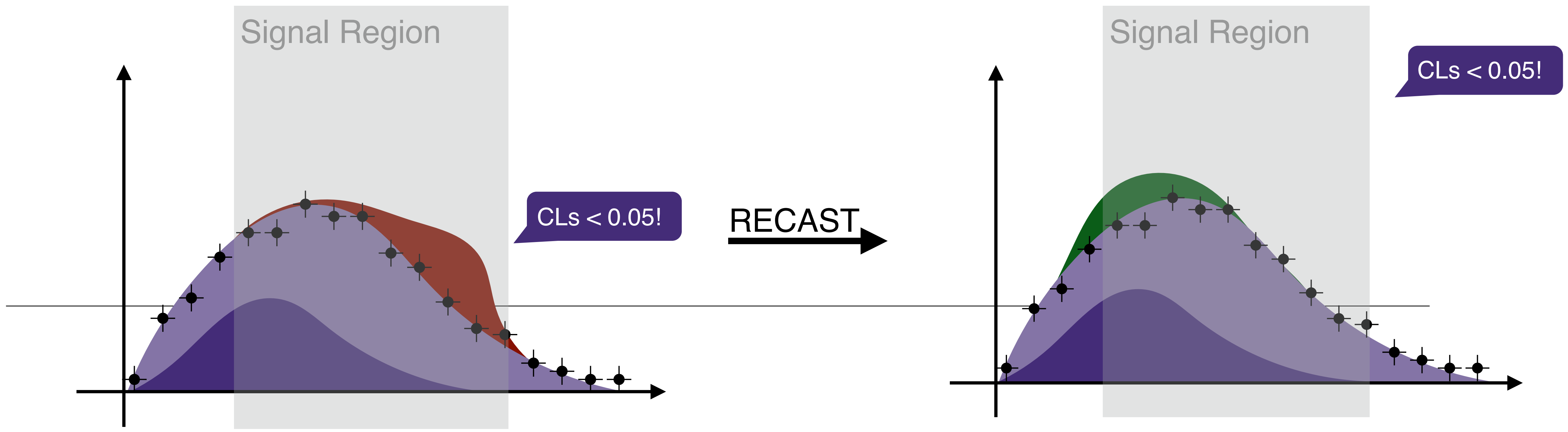


how to maximize scientific output of an analysis?

how to maximize scientific output of an analysis?

make analyses (rapidly) reusable.

reinterpret analyses w.r.t multiple models after publication



original analysis (w.r.t **model A**)

original analysis (recast to **model B**)

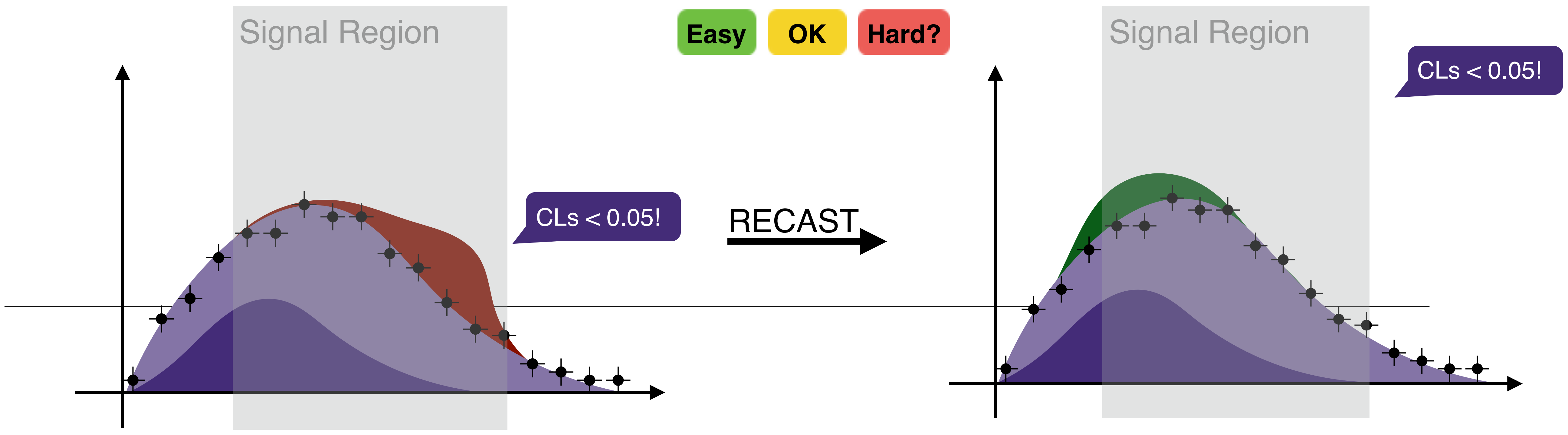
Reinterpretation Recipe:

At publication time:

1. archive data + background estimates in fully reduced form, i.e. as histograms / unbinned ntuples
2. **preserve original analysis pipeline** (at least such that we can run new signal sample). Not necessary to preserve ability to re-derive background estimates

Later:

1. Generate **new signal dataset** with same/compatible settings as original analysis (simulation, reconstruction, etc…)
   after sanity checks, e.g. DM abundance, cross-section, H mass, approx. SR acceptance)
2. Run dataset through **original analysis pipeline,** compare/fit against archived data and backgrounds.



original analysis (w.r.t **model A**)

original analysis (recast to **model B**)

Reinterpretation Recipe:

At publication time:

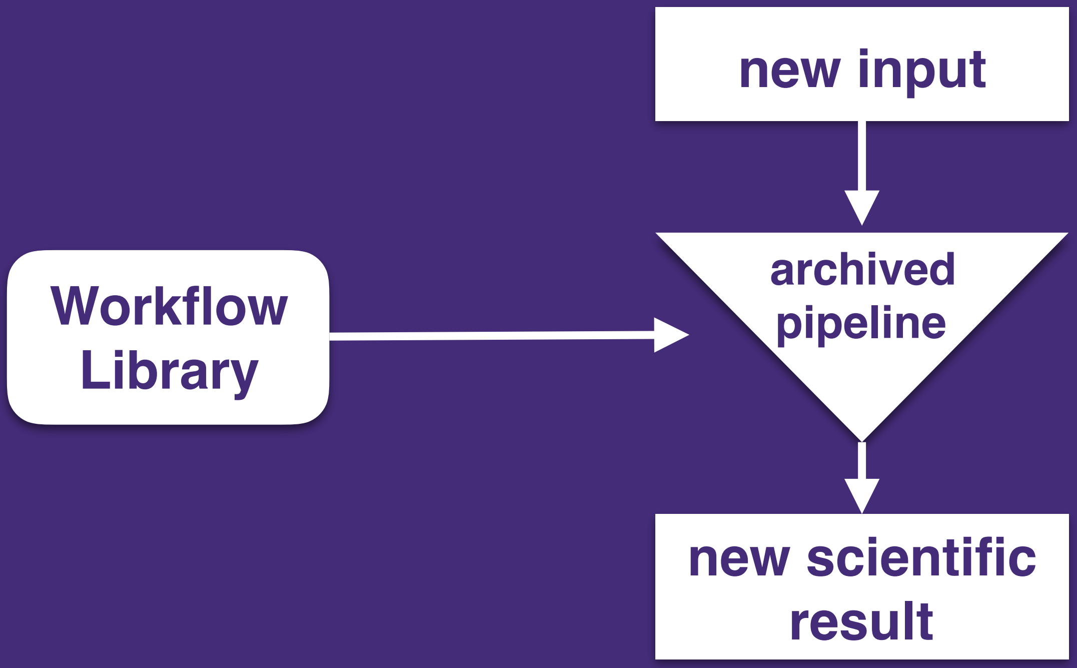
1. archive data + background estimates in fully reduced form, i.e. as histograms / unbinned ntuples
2. **preserve original analysis pipeline** (at least such that we can run new signal sample). Not necessary to preserve ability to re-derive background estimates

Later:

1. Generate **new signal dataset** with same/compatible settings as original analysis (simulation, reconstruction, etc…)
   after sanity checks, e.g. DM abundance, cross-section, H mass, approx. SR acceptance)
2. Run dataset through **original analysis pipeline,** compare/fit against archived data and backgrounds.

need a way to capture analysis pipelines such that they can be re-executed on *new input*

Reusability not Reproducibility



Complexity:

capturing an analysis for RECAST is easier and harder than *reproducing the original result*

Easier:

- **do not need to reproduce all aspects of analysis. certain parts of analysis are fixed**
  - discriminant training, cut optimization
  - background estimation techniques
  - reconstruction algorithms
  - …
- **can save certain aspects of invariant data**
  - observed data and background distributions in final reduced form (histograms)
  - no need for preservation of large background Monte Carlo datasets

Harder:

- **need to be prepared for “arbitrary” input data**
  - remove unnecessary hard-coded knowledge of model under consideration
- **need to preserve process, not just audit/trace of what software was run on what data**
  - need to know how to construct new jobs on new inputs based on e.g. templates

Analysis Preservation: two-step process

Modern HEP analysis:

- Multiple steps/code-bases, possibly developed by independent teams, with differing software requirements. Example: one team developing the event selection, another team developing the statistical analysis

Need to capture:

1. Individual processing steps

- code bases
- software environments
- identify binaries, scripts in code base
- templates how to run binaries (semantic description of arguments, naming etc..)
- description of step output, what are the relevant data fragments

2. How to connect these steps

- How to wire individual steps together
- What outputs of which steps, are used as inputs for other steps, …

Analysis Preservation:

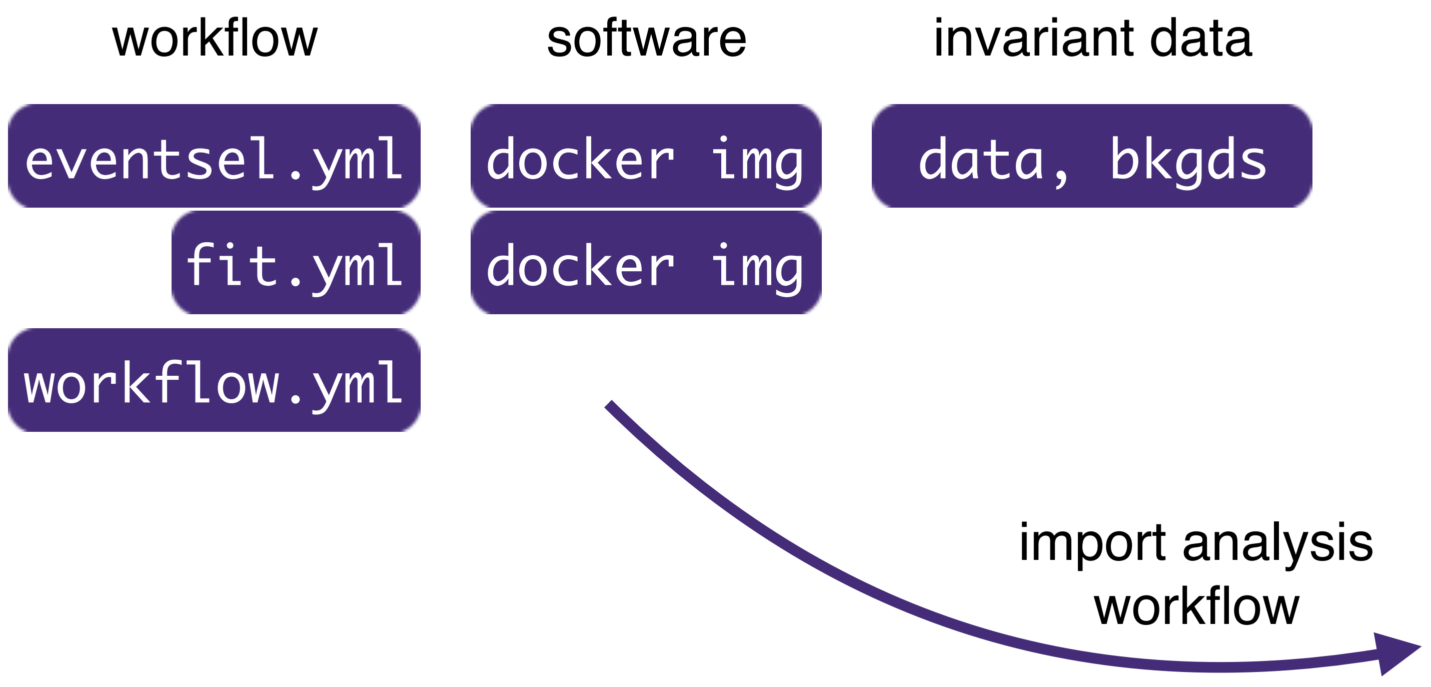
Some Goals

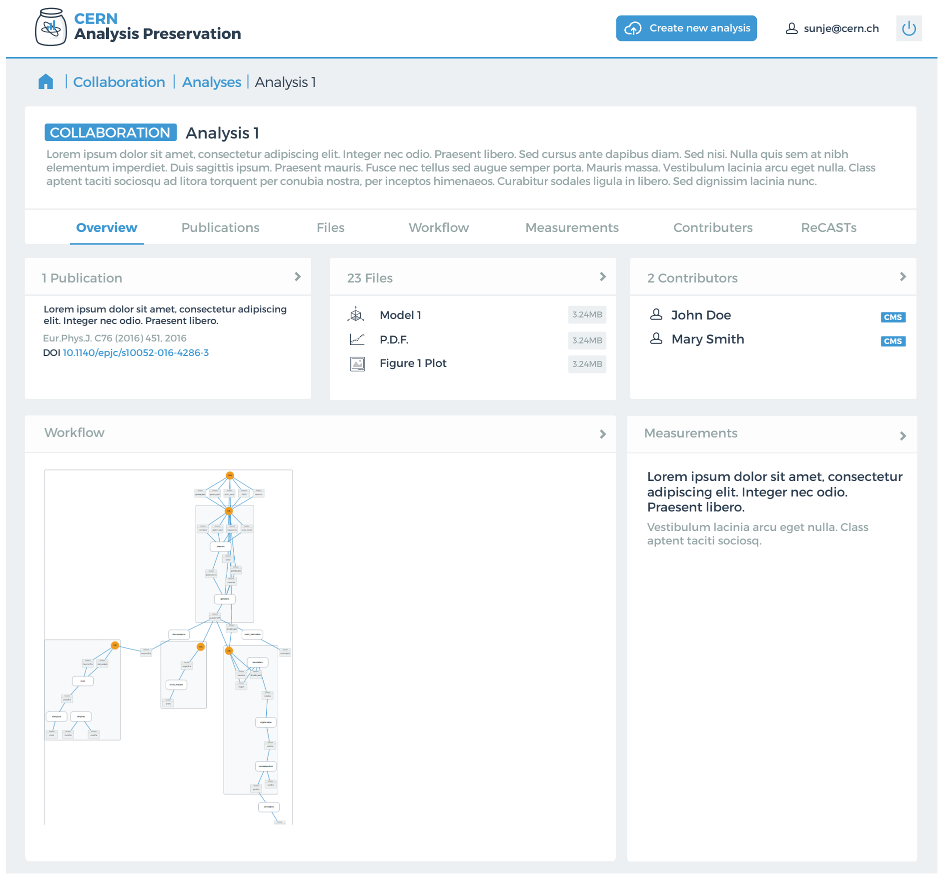
- **composable:** be able to re-use parts of pipeline from others, without changing yours
- **run on generic compute:** don't tie workflow description to specific technology, preserve relevant info, not how or where to run (no ganga / grid submission scripts please). assumption: Given some CPU, storage, memory, re-run you workflow
- **machine-readable and -writable:** make it easy to script e.g. compositions etc.
- **invariant:** capture moving parts once, do not depend on outside state
- **easy:** re-use well-known formats, technologies, make it easy to write for analysis teams
- **unintrusive:** adapt to you, not the other way around.
- **fast:** capturing an analysis should not take more than a few days. capture while developing

Analysis Workflow Preservation via yadage — rough outline

1. **Individual processing steps (packtivity):** docker images to capture software, job templates as YAML files
2. **Workflow (yadage):** YAML file of recipe how to build graph of packtivities

Since it's all JSON based, it's easily shared, imported, processed by machines. Fully integrated into CERN Analysis Preservation





Analysis 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet. Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa. Vestibulum lacinia arcu eget nulla. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc.

1 Publication

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero.
Eur.Phys.J. C76 (2016) 451, 2016
DOI 10.1140/epjc/s10052-016-4286-3

23 Files

Model 1	3.24MB
P.D.F.	3.24MB
Figure 1 Plot	3.24MB

2 Contributors

John Doe	CMS
Mary Smith	CMS

Workflow

Measurements

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero.
Vestibulum lacinia arcu eget nulla. Class aptent taciti sociosq.

Examples: packtivities

three pieces:

parametrized process:

template job from which we can produce concrete job

template: `“./DelphesHepMC <input file> <output file>”`

concrete: `“./DelphesHepMC /input/file/path.hepmc /output/file.root”`

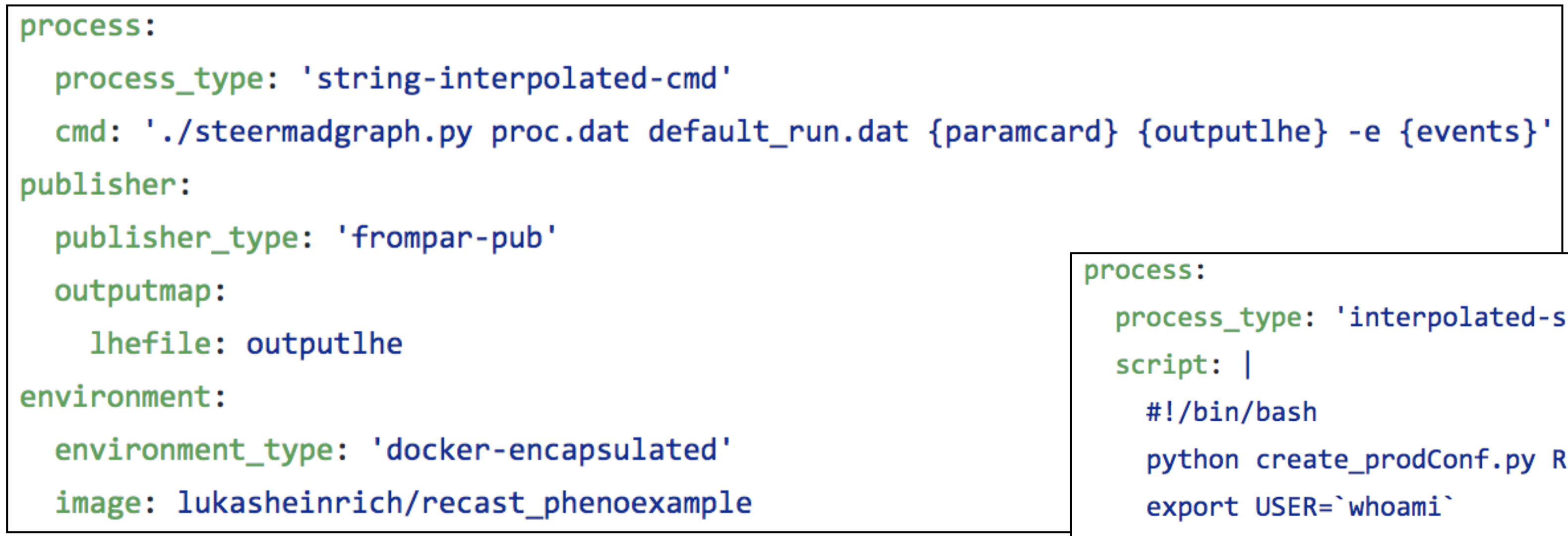
environment:

description of computing env in which above job can run. Best: docker image

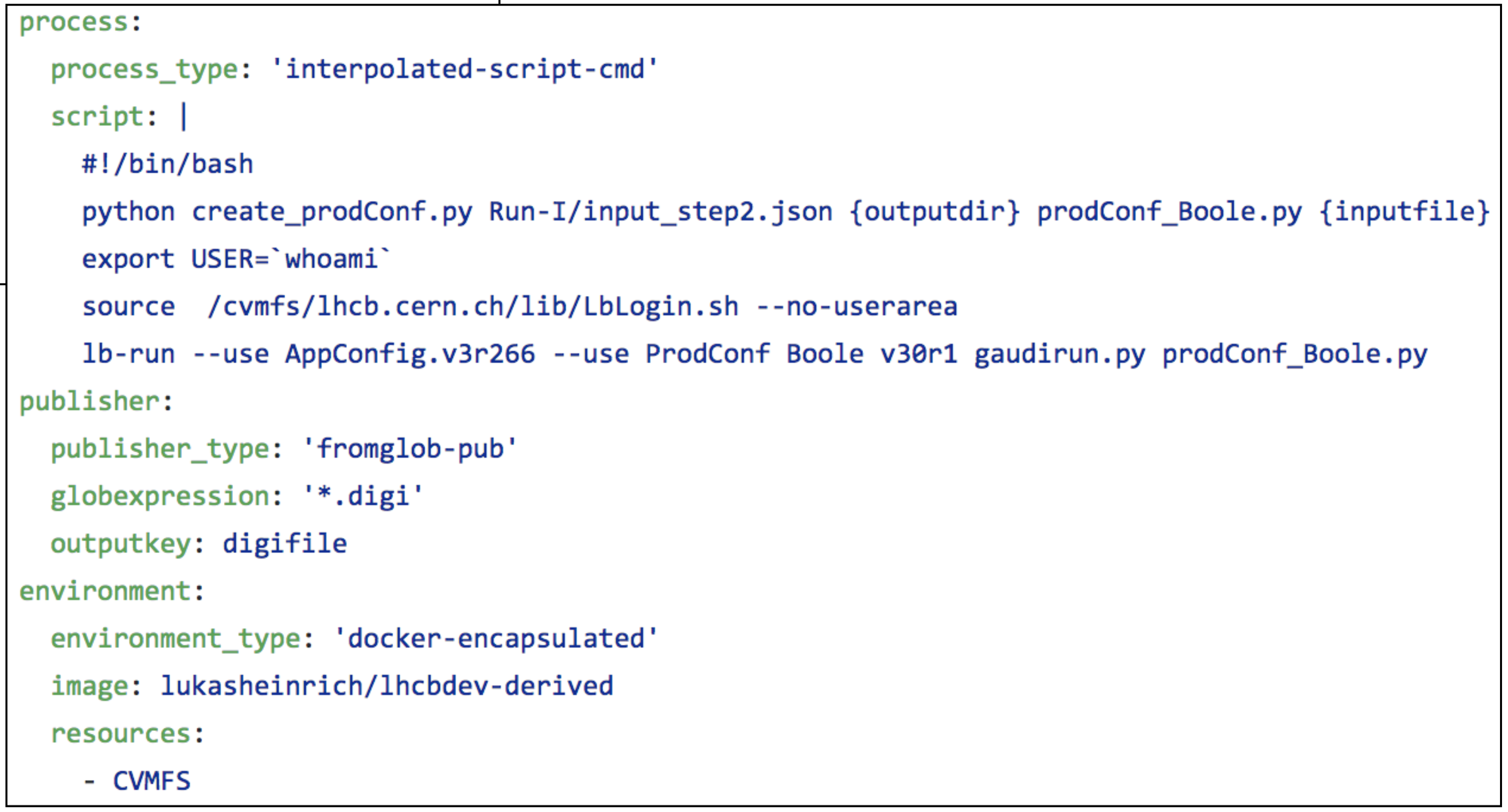
publisher:

recipe how to extract JSON result data after job completion

e.g. globbing files in a work directory, just declaring some of the input parameters as outputs

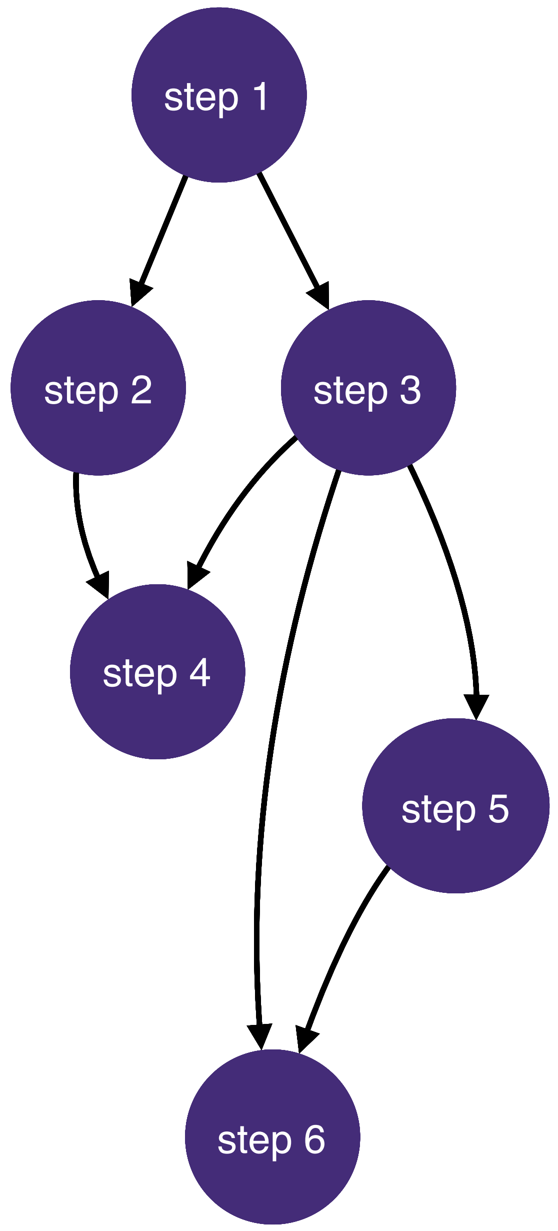


```
process:
  process_type: 'string-interpolated-cmd'
  cmd: './steermadgraph.py proc.dat default_run.dat {paramcard} {outputlhe} -e {events}'
publisher:
  publisher_type: 'frompar-pub'
  outputmap:
    lhefile: outputlhe
environment:
  environment_type: 'docker-encapsulated'
  image: lukasheinrich/recast_phenoexample
```



```
process:
  process_type: 'interpolated-script-cmd'
  script: |
    #!/bin/bash
    python create_prodConf.py Run-I/input_step2.json {outputdir} prodConf_Boole.py {inputfile}
    export USER=`whoami`
    source  /cvmfs/lhcb.cern.ch/lib/LbLogin.sh --no-userarea
    lb-run --use AppConfig.v3r266 --use ProdConf Boole v30r1 gaudirun.py prodConf_Boole.py
publisher:
  publisher_type: 'fromglob-pub'
  globexpression: '*.digi'
  outputkey: digifile
environment:
  environment_type: 'docker-encapsulated'
  image: lukasheinrich/lhcbdev-derived
  resources:
    - CVMFS
```

yadage workflows: dynamic topologies built at run-time



Natural Data Model: *directed acyclic graphs (DAGs)*

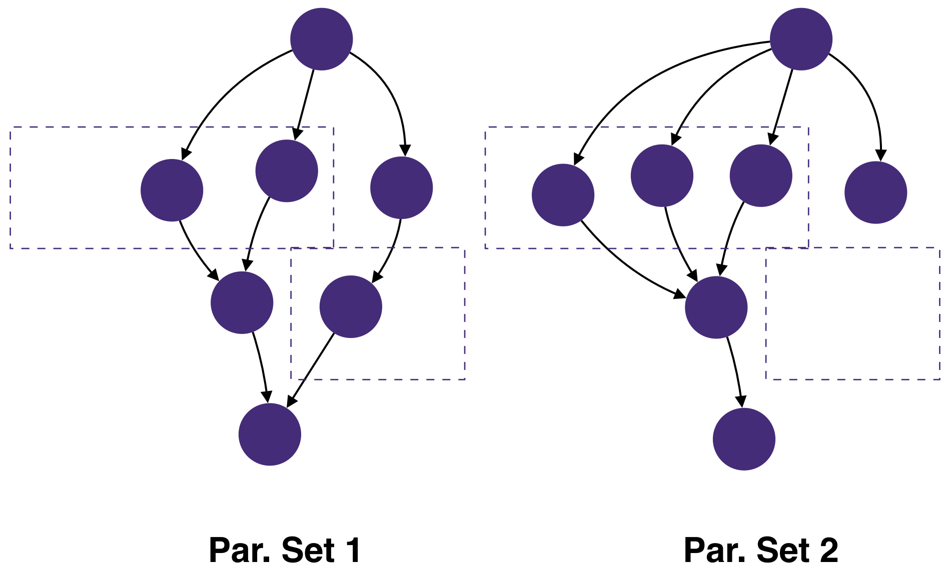
- **nodes**: individual steps
- **edges**: dependency relations

Two place where parametrization enter:

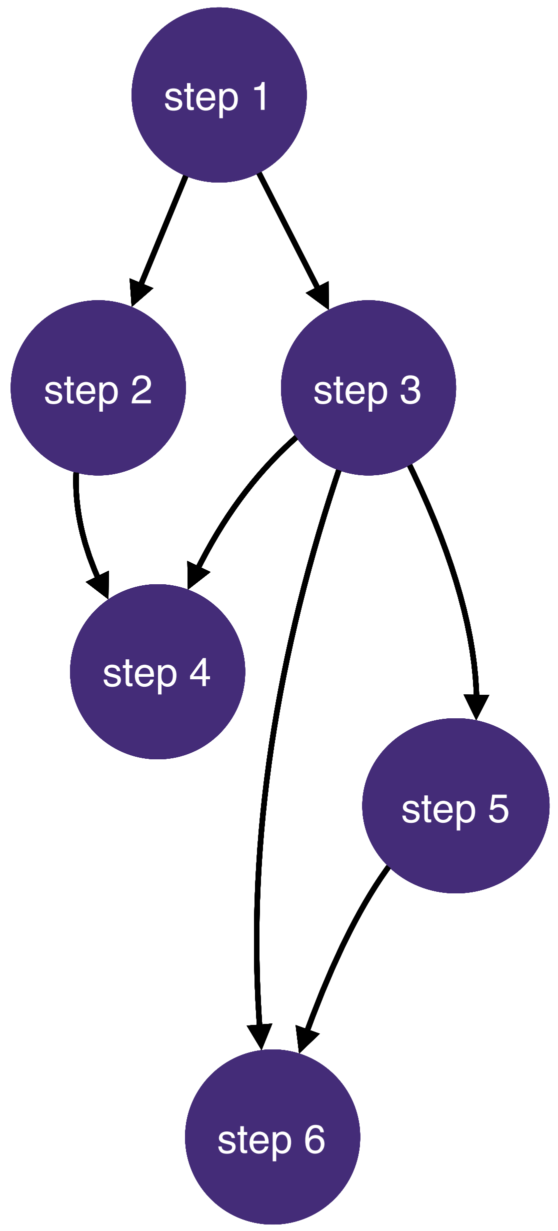
1. individual steps parametrized: covered by “packtivities”
2. graph topology may *depend on the parameters* of the analysis and only emerge during run-time

Examples:

- variable number of created files during execution,
- conditional choices (if/else)/flags do enable/disable steps, e.g. run systematics / not



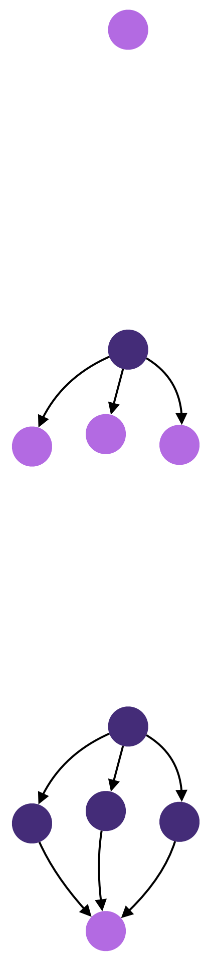
yadage workflows: dynamic topolgies built at run-time



Therefore: Sequentially build up graph, as sufficient information becomes available, using a number of stages that add nodes and edges

To capture analysis workflow, capture the stages.

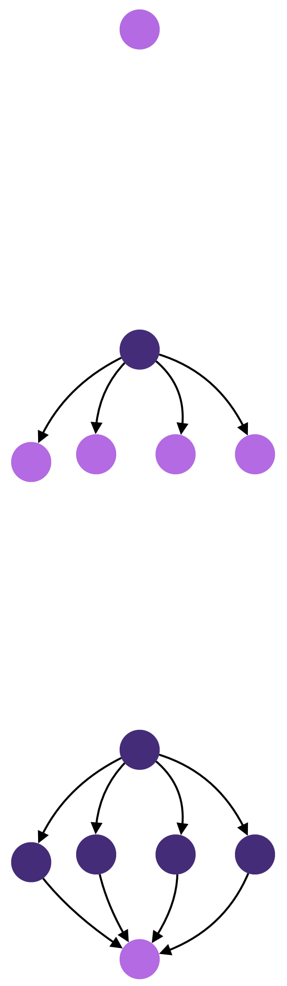
Example: Parametrized Map-Reduce



Stage 1:
unknown number of files. e.g. download & unpack archive with a priori unknown # of files

Stage 2:
for each file in the archive, add node to process it
(**only possible after first node done)**

Stage 3:
add a node that merges results of the map nodes
node/edge can be added before execution of map nodes



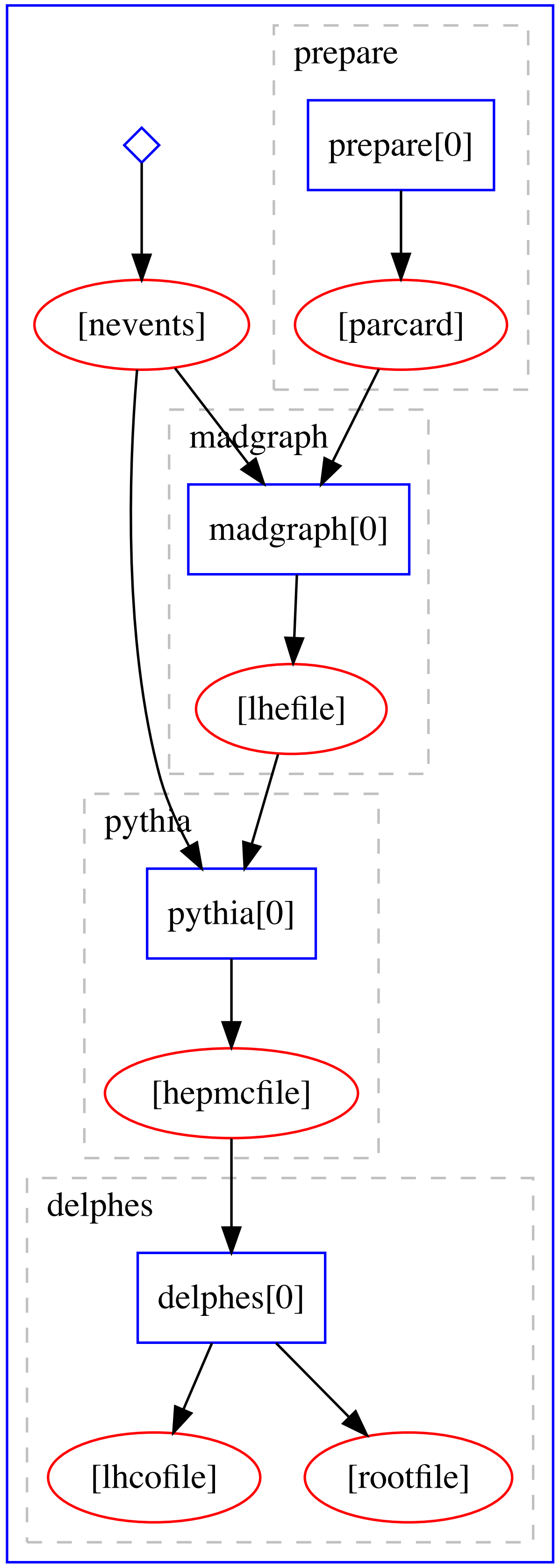
Par. Set 1

Par. Set 2

Examples: yadage workflows

three pieces:

```
stages:
  - name: prepare
    dependencies: []
    scheduler:
      scheduler_type: 'singlestep-stage'
      parameters:
        model: sm
        parametercard: '{workdir}/param.dat'
        inputpars: defaultparam.yml
      step: {$ref: 'preparestep.yml'}
  - name: madgraph
    dependencies: ['prepare','init']
    scheduler:
      scheduler_type: 'singlestep-stage'
      parameters:
        outputlhe: '{workdir}/output.lhe'
        events: {stages: init, output: nevents, unwrap: true}
        paramcard: {stages: prepare, output: parcard, unwrap: true}
      step: {$ref: 'madgraph.yml'}
  - name: pythia
    dependencies: ['madgraph']
    scheduler:
      scheduler_type: 'singlestep-stage'
      parameters:
        outputhepmc: '{workdir}/output.hepmc'
        events: {stages: init, output: nevents, unwrap: true}
        lhefile: {stages: madgraph, output: lhefile, unwrap: true}
      step: {$ref: 'pythia.yml'}
  - name: delphes
    dependencies:
      - pythia
    scheduler:
      scheduler_type: 'singlestep-stage'
      step: {$ref: 'delphes.yml'}
      parameters:
        outputroot: '{workdir}/output.root'
        outputlhco: '{workdir}/output.lhco'
        delphes_card: 'delphes/cards/delphes_card_ATLAS.tcl'
        inputhepmc: {stages: pythia, output: hepmcfile}
```



Executing yadage workflows

from command line:

```
$> pip install yadage
$> yadage-run gitlabwork workflow.yml -t gitlab-cern:lheinric/yadage-cern-workflows:busybox-helloworld
```

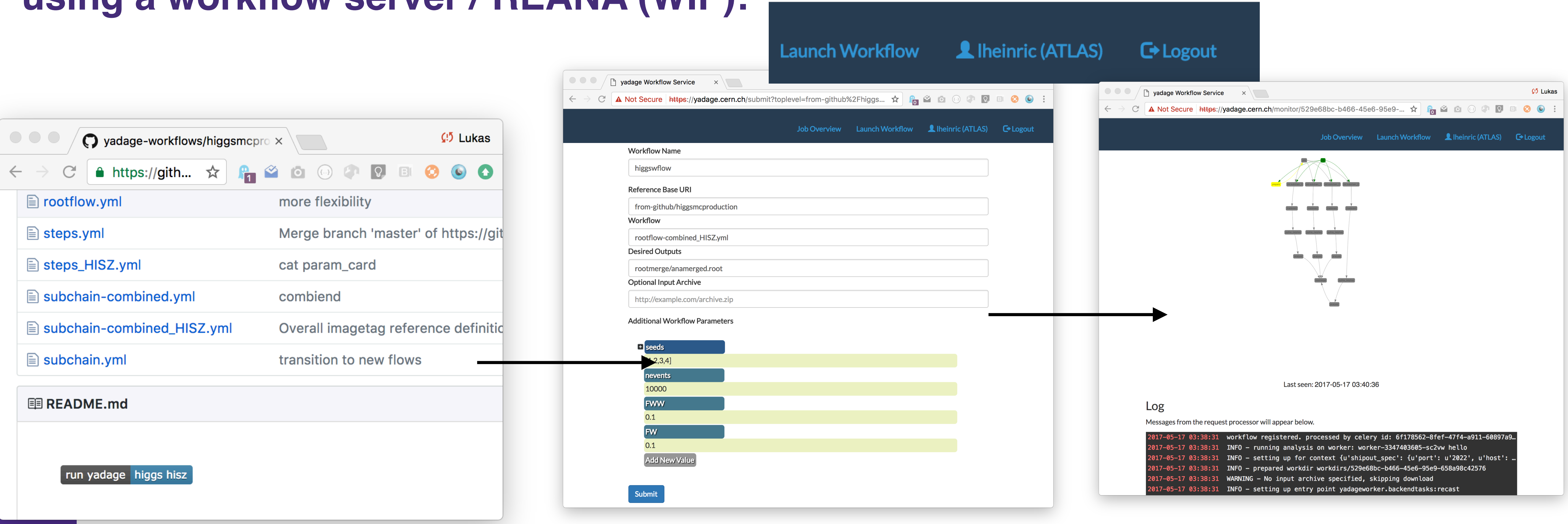
where to write data created by workflow

workflow

where to find workflow

using a workflow server / REANA (WIP):

CERN SSO protected



rootflow.yml	more flexibility
steps.yml	Merge branch 'master' of https://git
steps_HISZ.yml	cat param_card
subchain-combined.yml	combiend
subchain-combined_HISZ.yml	Overall imagetag reference definitic
subchain.yml	transition to new flows

README.md

Workflow Name: higgswflow

Reference Base URI: from-github/higgsmcproduction

Workflow: rootflow-combined_HISZ.yml

Desired Outputs: rootmerge/anamerged.root

Optional Input Archive: http://example.com/archive.zip

Additional Workflow Parameters

seeds	[1,2,3,4]
nevents	10000
FWW	0.1
FW	0.1

Last seen: 2017-05-17 03:40:36

Log

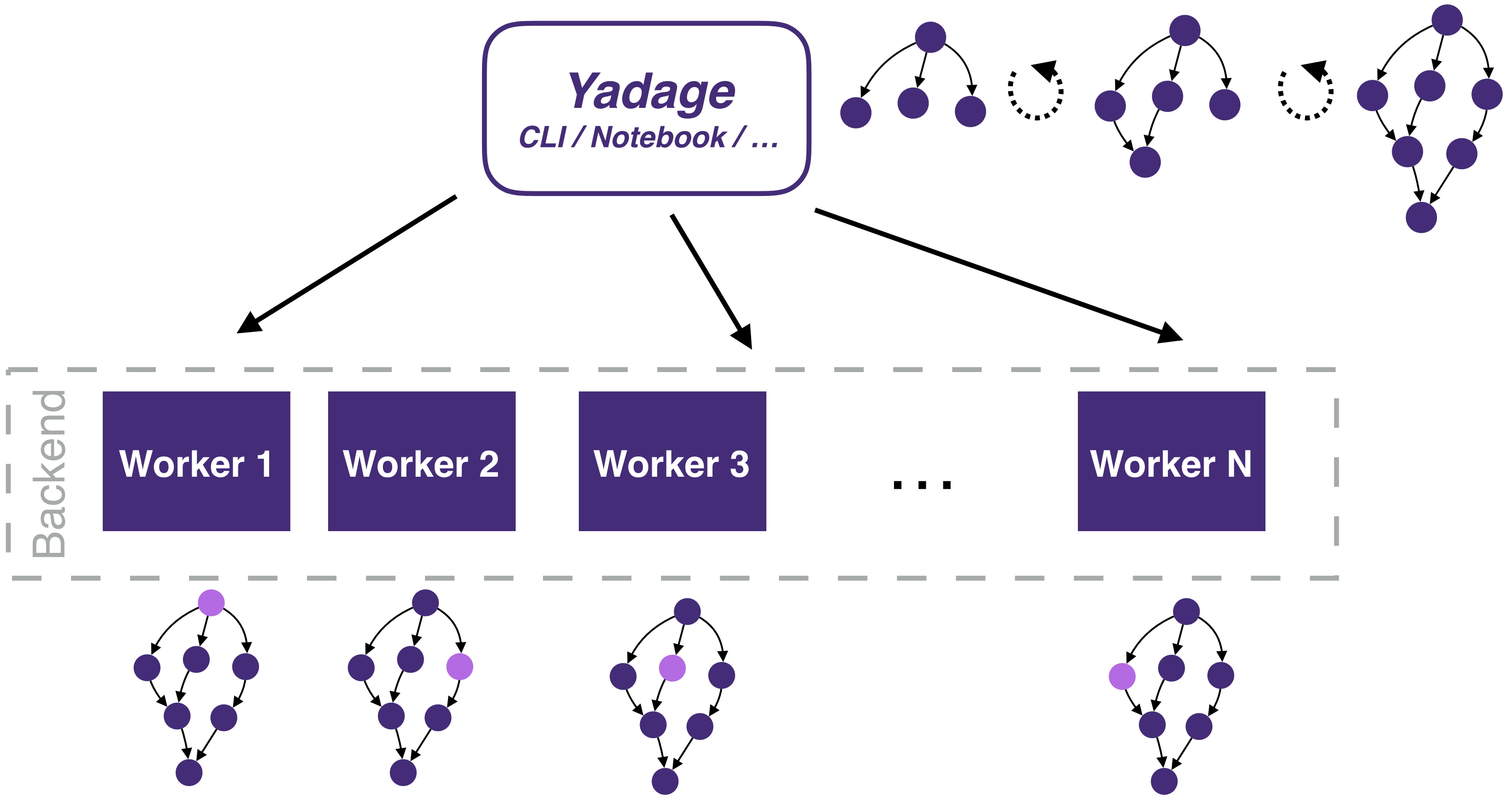
Messages from the request processor will appear below.

```
2017-05-17 03:38:31  workflow registered. processed by celery id: 6f178562-8fef-47f4-a911-60897a9…
2017-05-17 03:38:31  INFO - running analysis on worker: worker-3347403605-sc2vw hello
2017-05-17 03:38:31  INFO - setting up for context {u'shipout_spec': {u'port': u'2022', u'host': …
2017-05-17 03:38:31  INFO - prepared workdir workdirs/529e68bc-b466-45e6-95e9-658a98c42576
2017-05-17 03:38:31  WARNING - No input archive specified, skipping download
2017-05-17 03:38:31  INFO - setting up entry point yadageworker.backendtasks:recast
```

Distributing Workflows across multiple Hosts in a workflow server

multiple backends possible: ipython clusters, kubernetes, celery. Backend should be able to run containers, as they are most common runtime spec.

On REANA, we re-use native Kubernetes Job API to distribute Jobs across cluster. Shared Storage via CephFS, experiement / archived data via EOS/ XrootD, Authentication via Kerberos / VOMS proxy.



LHCb experience

captured both production-type and analysis workloads in yadage

1. Monte Carlo Simulation/Reconstruction workflow:
   Gauss → Boole → MooreL0 → Moore → Brunel → DaVinci

2. Analysis:

   Analysis separately developed in LHCb captured via Docker Container + internal workflow tool snakelike (simple workflow for yadage)

Towards a streamlined RECAST service

RECAST Infrastructure Overview:

With archived analysis workflows it becomes feasible to streamline the reinterpretation efforts.
Reinterpretations as a collaboration- (or community-) wide service.
Originally suggested by Cranmer, Yavin [arXiv:1010.2506]

recast

Idea:

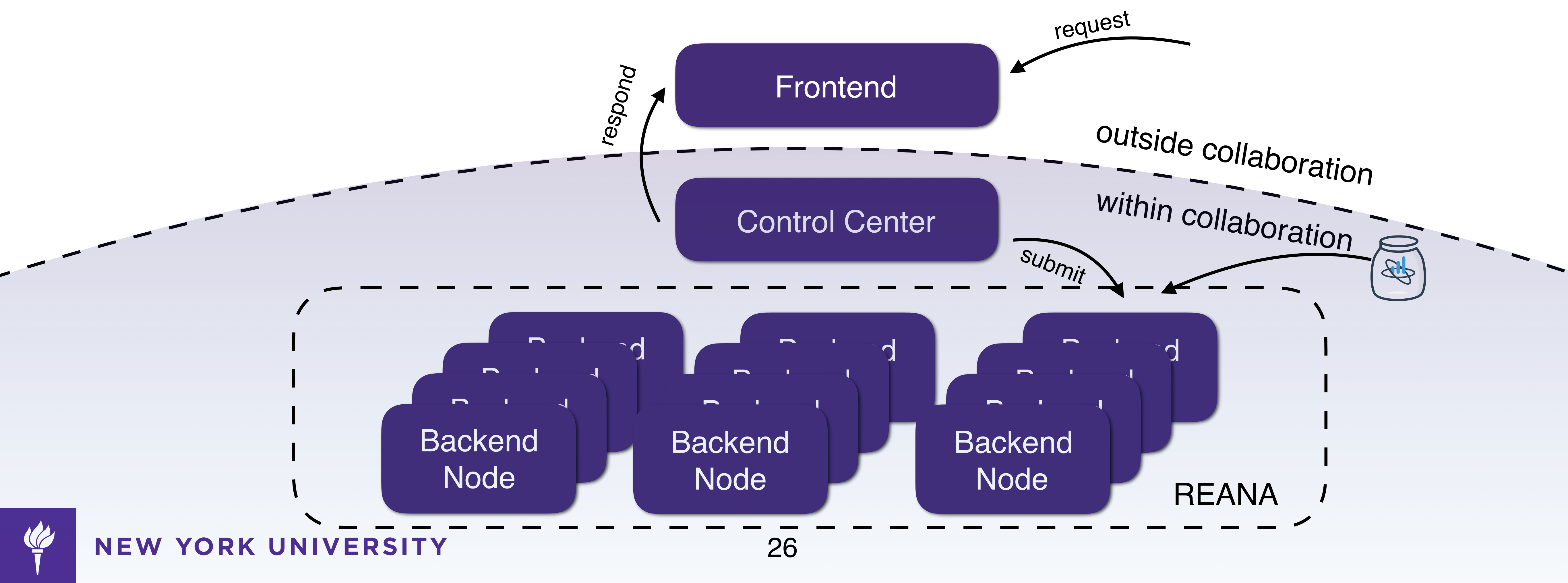
- Produce reinterpretations of same fidelity as original result (not just approximations)
- Allow hep-ph community to *suggest* reinterpretations through a standard (web) interface. They provide most interesting points / scans to do. Auxiliary information such as run cards, SLHA spectra, UFO models
- LHC collaborations review suggestions and choose which to fulfill (based on scale of request, availability of a preserved analysis, physics case)
- Use archived analysis to (semi-) automatically run reinterpretation. Review results, approve (possibly on accelerated track, since analysis already approved).
- Publish and/or append original analysis HEPDATA record.
- Allows us to decouple original publication from reinterpretations. Publish early using benchmark signals, continuously re-interpret as samples become available

REANA: HEP workflows as a service

generic service offered at CERN, agnostic to type of workflow being run, semantics of why workflows are run

Based on REANA infrastructure / capability, we can enable physics applications / web services

RECAST: reinterpretations as a service

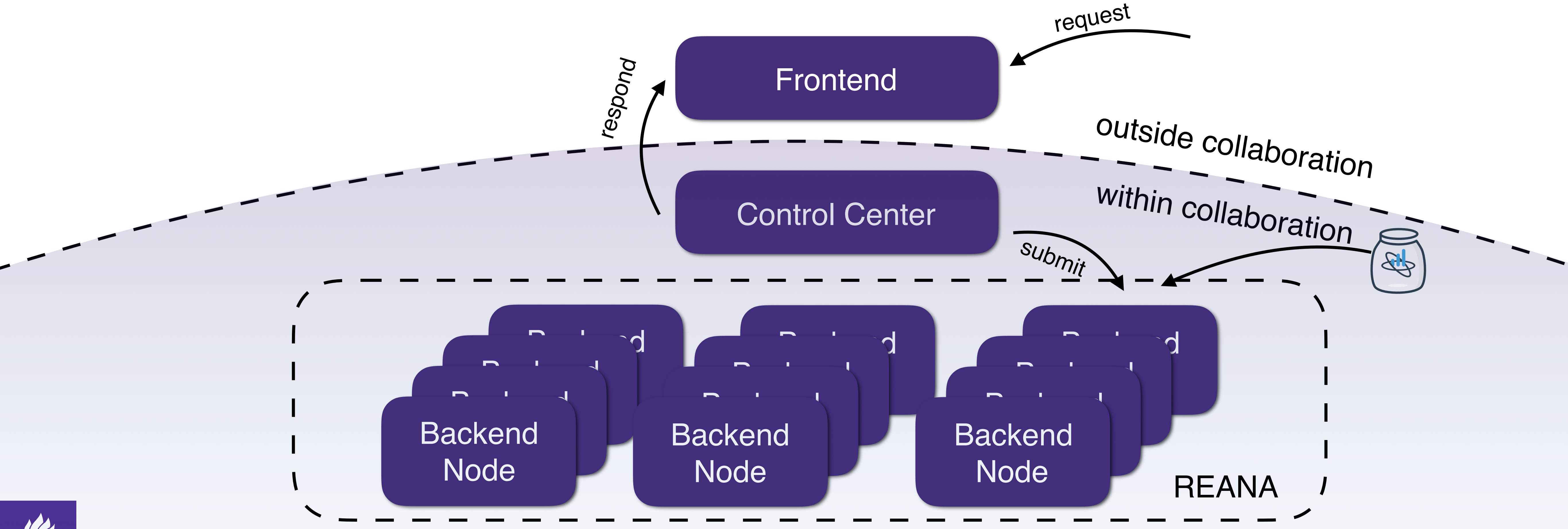


RECAST Overview:

Idea: Structured Interface between theory and experimental communities to assess BSM models

Theorists: propose models / parameter scans w.r.t. which to reinterpret a given analysis
Experimentalists: asses proposals, if worthwhile;

1. Generate new Signal
2. Run archived analysis pipeline on new signal
3. review / approve results
4. publish results / append original paper



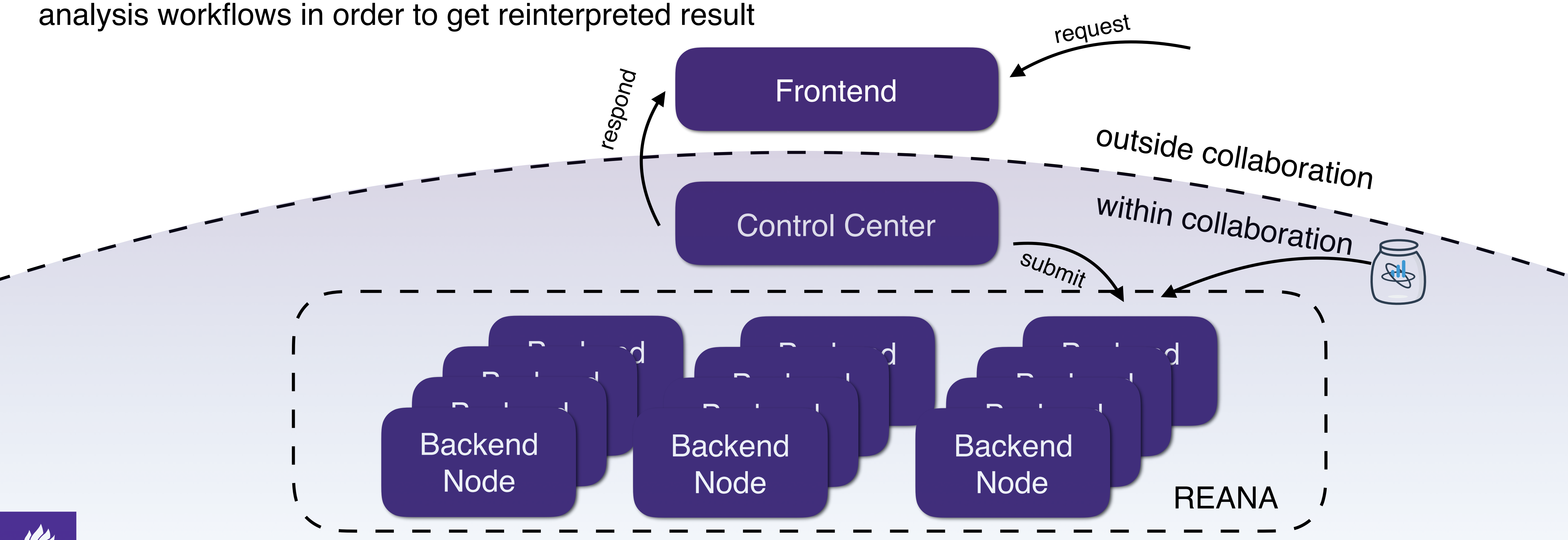
RECAST Infrastructure Overview:

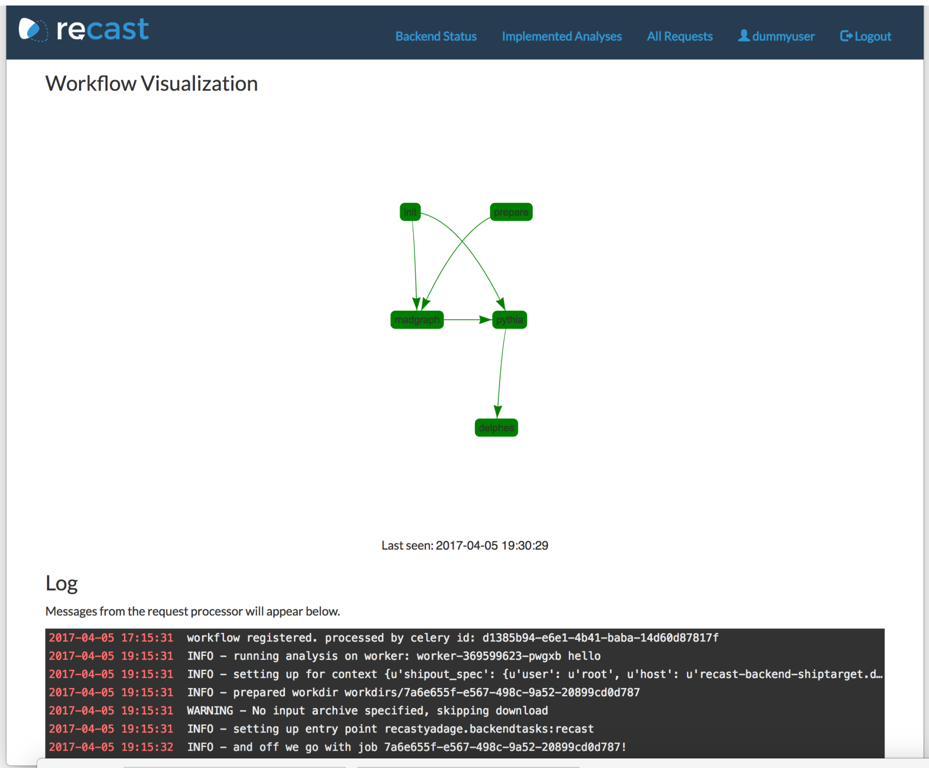
With archived analysis workflows it becomes feasible to streamline reinterpretations-as-a-service.

Frontend: public-facing web-service (+ API). Let's e.g. phenomenologists register interest in reinterpretations for specific published analyses. Allows them to provide auxiliary data (SLHA spectrum files, generator run cards, etc..). **Note:** No guarantee of fulfillment of request by collaborations.

Control Center: collaboration-internal web-service (+ API) to inspect incoming requests, compare against catalogue of archived analysis, allows submission to backend to actually perform reinterpretation. Can push "RECAST response" back to frontend.

Backend Cluster: distributed compute resource running on CERN OpenStack infrastructure to execute analysis workflows in order to get reinterpreted result





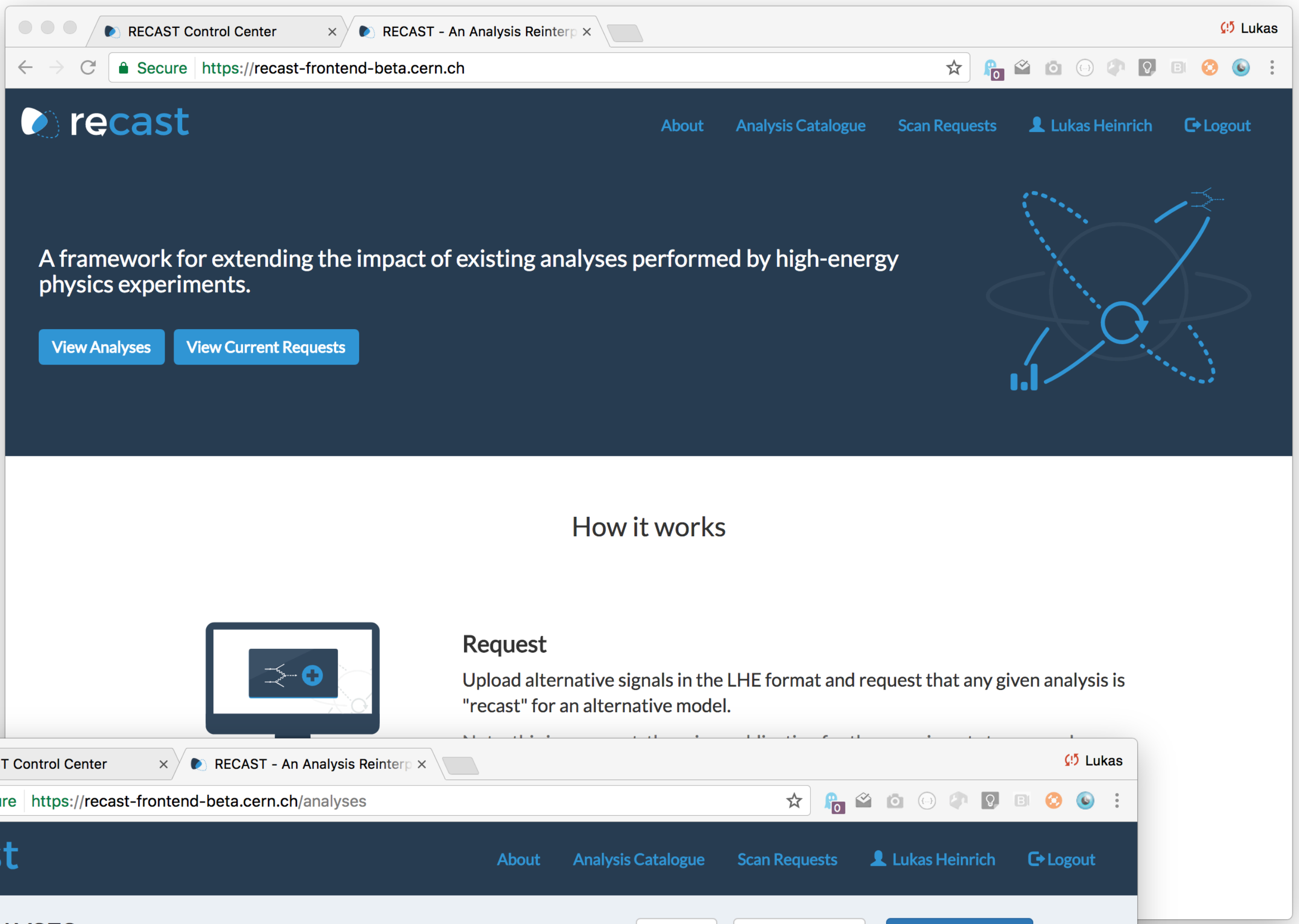
Workflow Visualization

Last seen: 2017-04-05 19:30:29

Log

Messages from the request processor will appear below.

```
2017-04-05 17:15:31  workflow registered. processed by celery id: d1385b94-e6e1-4b41-baba-14d60d87817f
2017-04-05 19:15:31  INFO - running analysis on worker: worker-369599623-pwgxb hello
2017-04-05 19:15:31  INFO - setting up for context {u'shipout_spec': {u'user': u'root', u'host': u'recast-backend-shiptarget.d…
2017-04-05 19:15:31  INFO - prepared workdir workdirs/7a6e655f-e567-498c-9a52-20899cd0d787
2017-04-05 19:15:31  WARNING - No input archive specified, skipping download
2017-04-05 19:15:31  INFO - setting up entry point recastyadage.backendtasks:recast
2017-04-05 19:15:32  INFO - and off we go with job 7a6e655f-e567-498c-9a52-20899cd0d787!
```

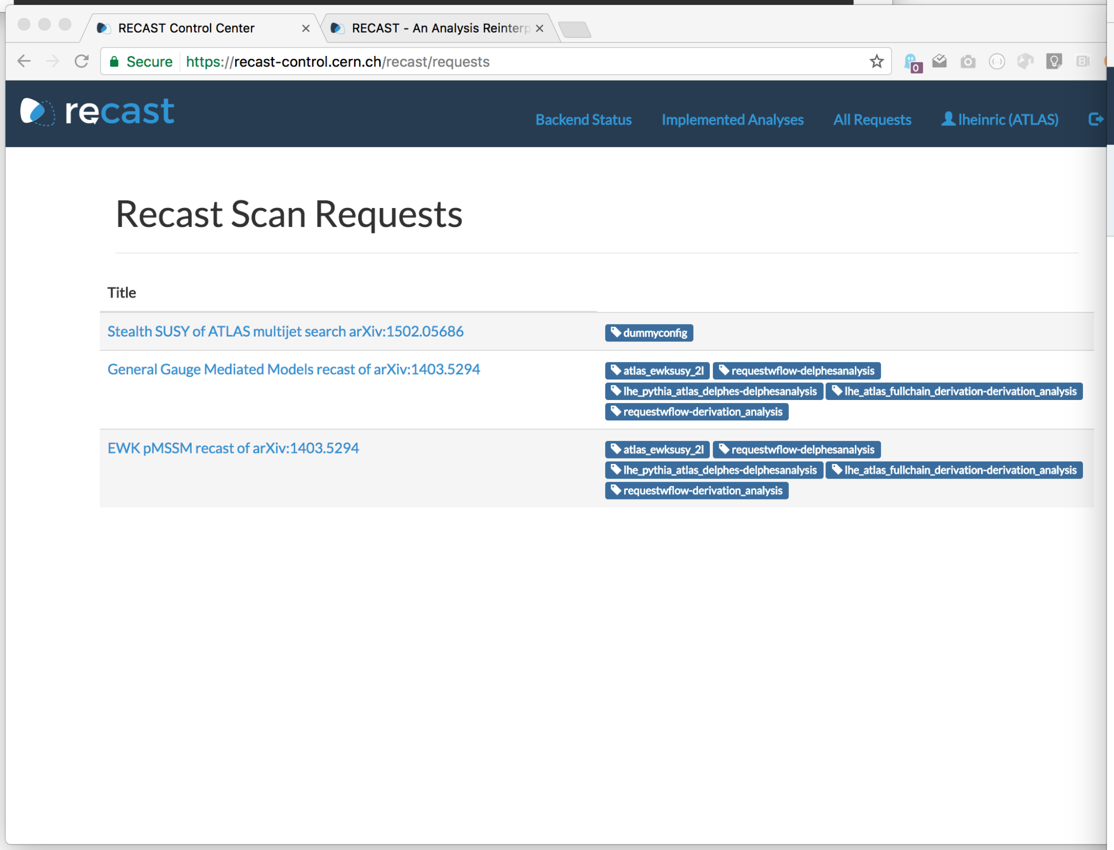


A framework for extending the impact of existing analyses performed by high-energy physics experiments.

How it works

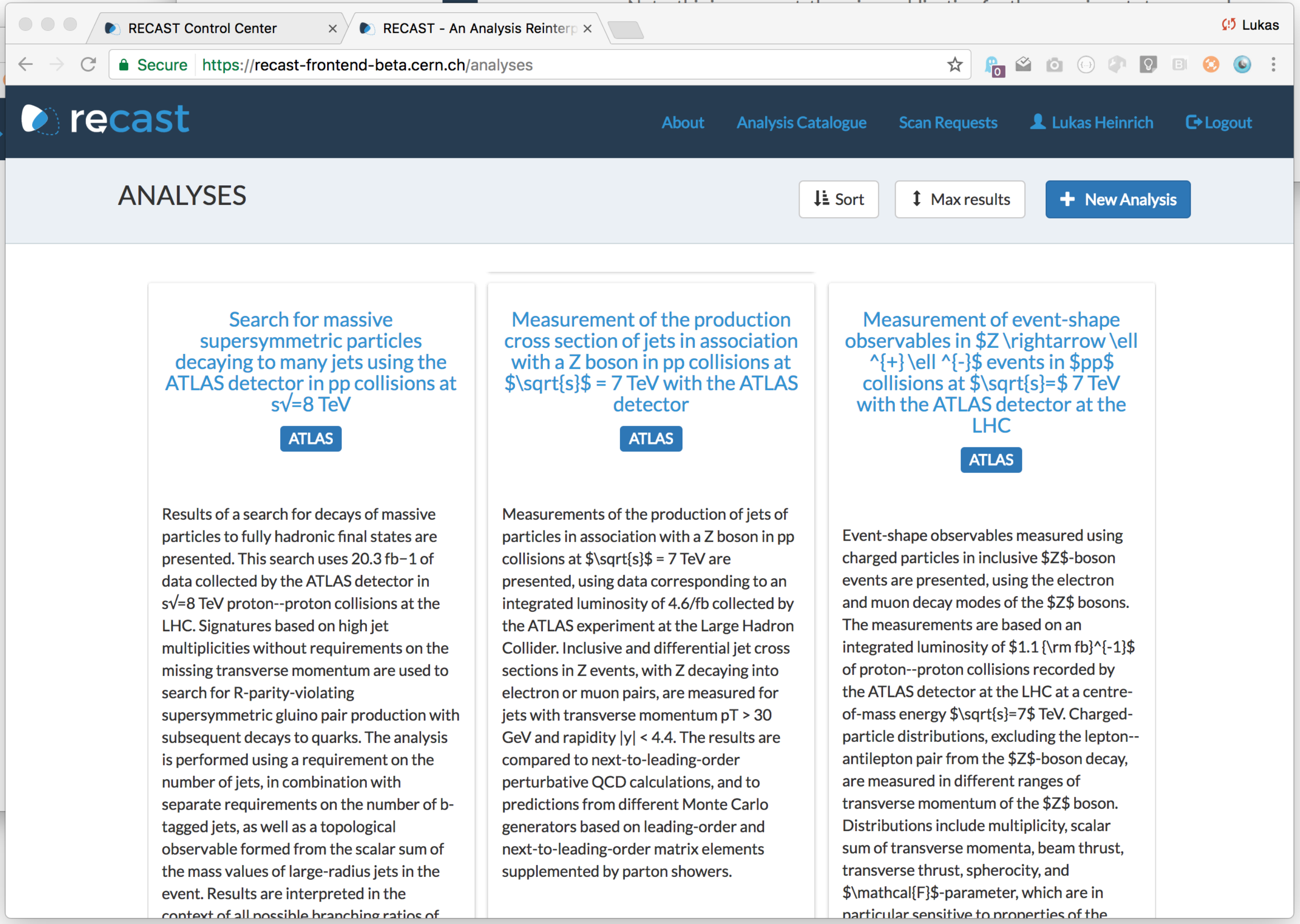
Request

Upload alternative signals in the LHE format and request that any given analysis is "recast" for an alternative model.



Recast Scan Requests

Title	
Stealth SUSY of ATLAS multijet search arXiv:1502.05686	dummyconfig
General Gauge Mediated Models recast of arXiv:1403.5294	atlas_ewksusy_2l, requestwflow-delphesanalysis, lhe_pythia_atlas_delphes-delphesanalysis, lhe_atlas_fullchain_derivation-derivation_analysis, requestwflow-derivation_analysis
EWK pMSSM recast of arXiv:1403.5294	atlas_ewksusy_2l, requestwflow-delphesanalysis, lhe_pythia_atlas_delphes-delphesanalysis, lhe_atlas_fullchain_derivation-derivation_analysis, requestwflow-derivation_analysis



ANALYSES

Search for massive supersymmetric particles decaying to many jets using the ATLAS detector in pp collisions at s√=8 TeV

Results of a search for decays of massive particles to fully hadronic final states are presented. This search uses 20.3 fb−1 of data collected by the ATLAS detector in s√=8 TeV proton--proton collisions at the LHC. Signatures based on high jet multiplicities without requirements on the missing transverse momentum are used to search for R-parity-violating supersymmetric gluino pair production with subsequent decays to quarks. The analysis is performed using a requirement on the number of jets, in combination with separate requirements on the number of b-tagged jets, as well as a topological observable formed from the scalar sum of the mass values of large-radius jets in the event. Results are interpreted in the

Measurement of the production cross section of jets in association with a Z boson in pp collisions at $\sqrt{s}$ = 7 TeV with the ATLAS detector

Measurements of the production of jets of particles in association with a Z boson in pp collisions at $\sqrt{s}$ = 7 TeV are presented, using data corresponding to an integrated luminosity of 4.6/fb collected by the ATLAS experiment at the Large Hadron Collider. Inclusive and differential jet cross sections in Z events, with Z decaying into electron or muon pairs, are measured for jets with transverse momentum pT > 30 GeV and rapidity |y| < 4.4. The results are compared to next-to-leading-order perturbative QCD calculations, and to predictions from different Monte Carlo generators based on leading-order and next-to-leading-order matrix elements supplemented by parton showers.

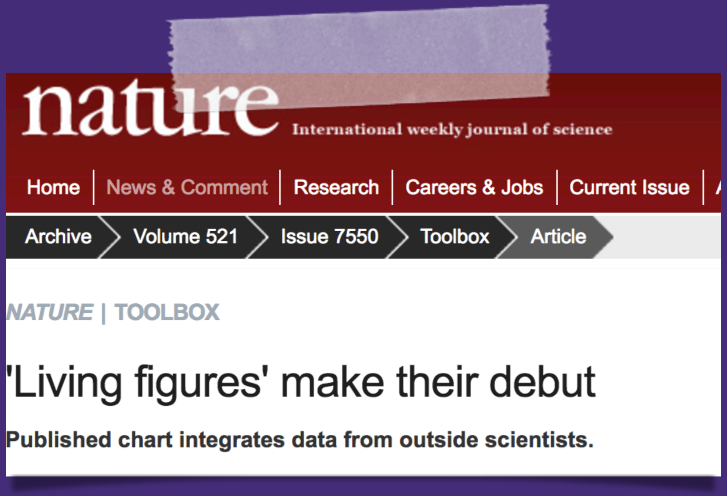
Measurement of event-shape observables in $Z \rightarrow \ell ^{+} \ell ^{-}$ events in pp collisions at $\sqrt{s}$=$ 7 TeV with the ATLAS detector at the LHC

Event-shape observables measured using charged particles in inclusive Z-boson events are presented, using the electron and muon decay modes of the Z bosons. The measurements are based on an integrated luminosity of $1.1 {\rm fb}^{-1}$ of proton--proton collisions recorded by the ATLAS detector at the LHC at a centre-of-mass energy $\sqrt{s}=7$ TeV. Charged-particle distributions, excluding the lepton--antilepton pair from the Z-boson decay, are measured in different ranges of transverse momentum of the Z boson. Distributions include multiplicity, scalar sum of transverse momenta, beam thrust, transverse thrust, spherocity, and $\mathcal{F}$-parameter, which are in

NEW YORK UNIVERSITY

Points of discussion for LHCb:

1. do you have use-case for parametrized pipelines? Adding data to a measurement? “Living Figure”?


nature
International weekly journal of science
Home | News & Comment | Research | Careers & Jobs | Current Issue
Archive > Volume 521 > Issue 7550 > Toolbox > Article
NATURE | TOOLBOX

'Living figures' make their debut

Published chart integrates data from outside scientists.

2. do you have use-case for structured scientific interface between theory/experiment which would benefit from computational workflow backend? PHASE network?

Analysis preservation is not only for reproducibility.

Also new science result by re-using algorithms of analysis on new inputs. (reinterpretation)

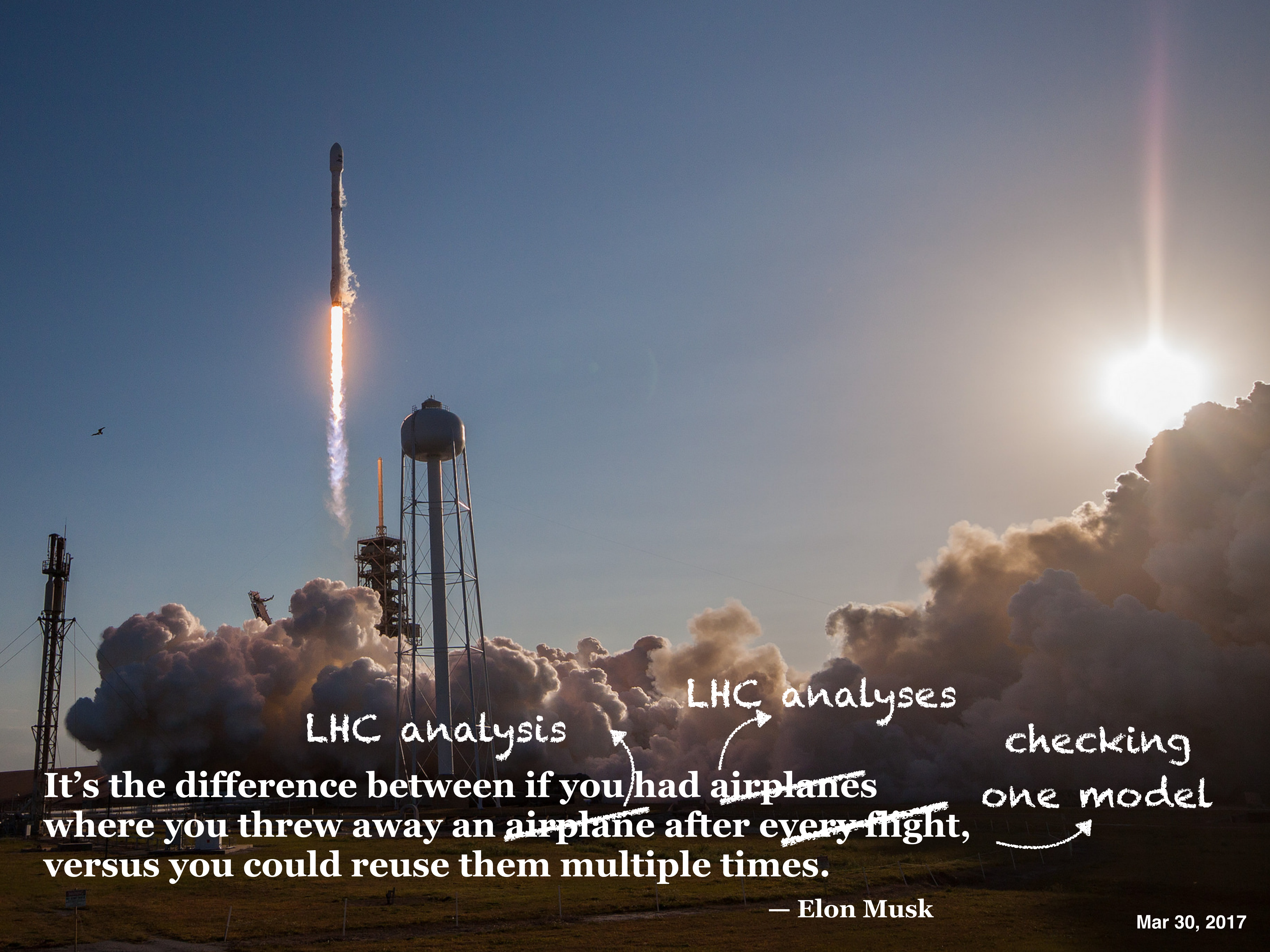
yadage / packtivity: backend-agnostic workflow description in YAML. Integrated into CAP and RECAST projects. Distributed Workflows based on Containers.

Recent Successes in Reusability…

It’s the difference between if you had airplanes
where you threw away an airplane after every flight,
versus you could reuse them multiple times.

— Elon Musk

Mar 30, 2017



LHC analyses

LHC analysis

checking one model

It’s the difference between if you had ~~airplanes~~ where you threw away an ~~airplane~~ after ~~every flight~~, versus you could reuse them multiple times.

— Elon Musk

Mar 30, 2017

Questions?

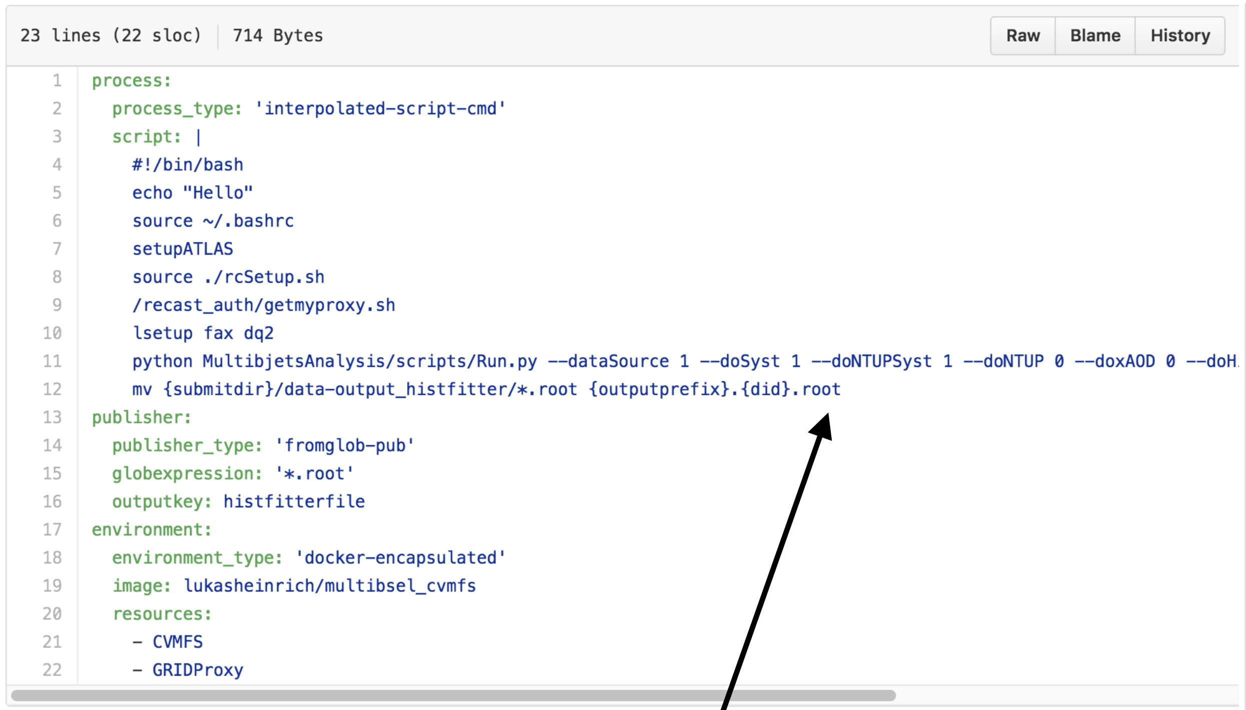
Appendix

Case Study: Multi B-jets analysis

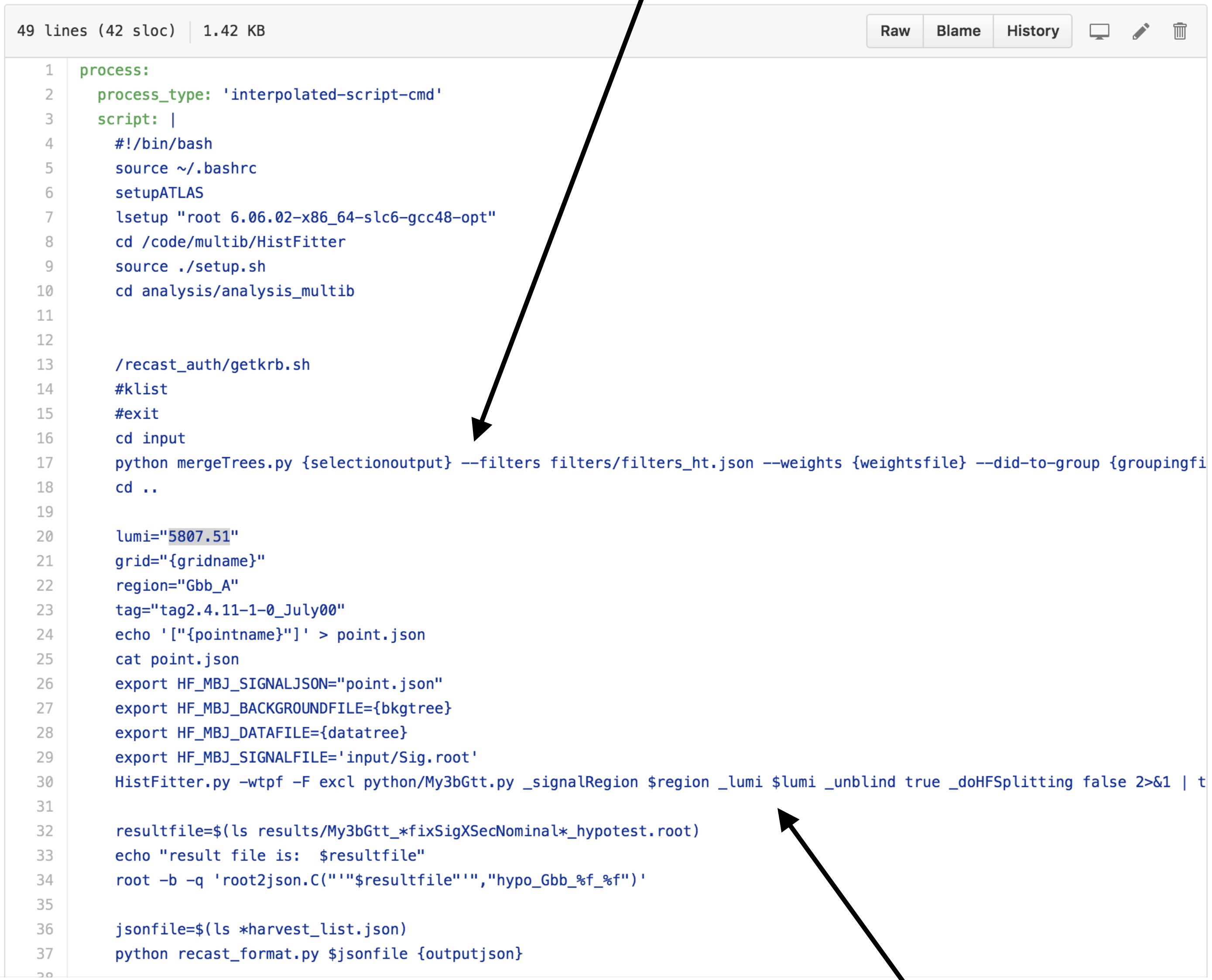
Defining the individual Workflow steps

- need script that tell us how to run the code once we are in the right environment. parametrized by a few variables (input file names etc)
- can use simple shell script, but also anything else

lumi/xsec/KF/FE weighting of HF tree


23 lines (22 sloc) | 714 Bytes

```
process:
  process_type: 'interpolated-script-cmd'
  script: |
    #!/bin/bash
    echo "Hello"
    source ~/.bashrc
    setupATLAS
    source ./rcSetup.sh
    /recast_auth/getmyproxy.sh
    lsetup fax dq2
    python MultibjetsAnalysis/scripts/Run.py --dataSource 1 --doSyst 1 --doNTUPSyst 1 --doNTUP 0 --doxAOD 0 --doH
    mv {submitdir}/data-output_histfitter/*.root {outputprefix}.{did}.root
publisher:
  publisher_type: 'fromglob-pub'
  globexpression: '*.root'
  outputkey: histfitterfile
environment:
  environment_type: 'docker-encapsulated'
  image: lukasheinrich/multibsel_cvmfs
  resources:
    - CVMFS
    - GRIDProxy
```


49 lines (42 sloc) | 1.42 KB

```
process:
  process_type: 'interpolated-script-cmd'
  script: |
    #!/bin/bash
    source ~/.bashrc
    setupATLAS
    lsetup "root 6.06.02-x86_64-slc6-gcc48-opt"
    cd /code/multib/HistFitter
    source ./setup.sh
    cd analysis/analysis_multib


    /recast_auth/getkrb.sh
    #klist
    #exit
    cd input
    python mergeTrees.py {selectionoutput} --filters filters/filters_ht.json --weights {weightsfile} --did-to-group {groupingfi
    cd ..

    lumi="5807.51"
    grid="{gridname}"
    region="Gbb_A"
    tag="tag2.4.11-1-0_July00"
    echo '["{pointname}"]' > point.json
    cat point.json
    export HF_MBJ_SIGNALJSON="point.json"
    export HF_MBJ_BACKGROUNDFILE={bkgtree}
    export HF_MBJ_DATAFILE={datatree}
    export HF_MBJ_SIGNALFILE='input/Sig.root'
    HistFitter.py -wtpf -F excl python/My3bGtt.py _signalRegion $region _lumi $lumi _unblind true _doHFSplitting false 2>&1 | t

    resultfile=$(ls results/My3bGtt_*fixSigXSecNominal*_hypotest.root)
    echo "result file is:  $resultfile"
    root -b -q 'root2json.C("'"$resultfile"'","hypo_Gbb_%f_%f")'

    jsonfile=$(ls *harvest_list.json)
    python recast_format.py $jsonfile {outputjson}
```

direct SH Driver reads signal dataset (a SUSY10 derivation) via XrootD writes out HistFitter tree

Run HF

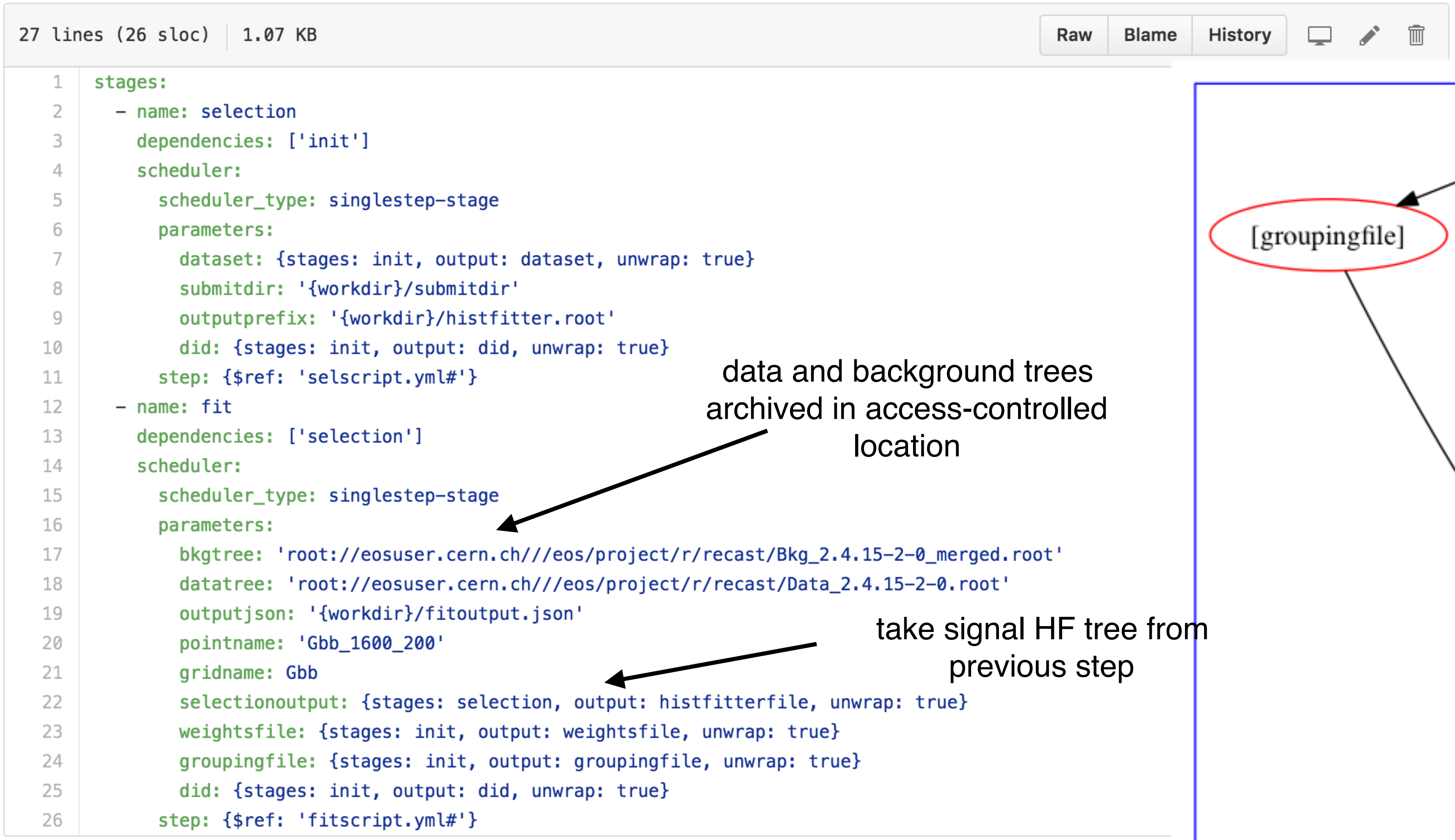
Extract Results into JSON format

NEW YORK UNIVERSITY

Case Study: Multi B-jets analysis

Stringing the workflow together

- small file on how the individual pieces fit together.
- Here: dataset, AMI info file etc provided as input parameters, define EOS location of signal and background trees, declare that signal histfitter tree comes from previous selection step etc

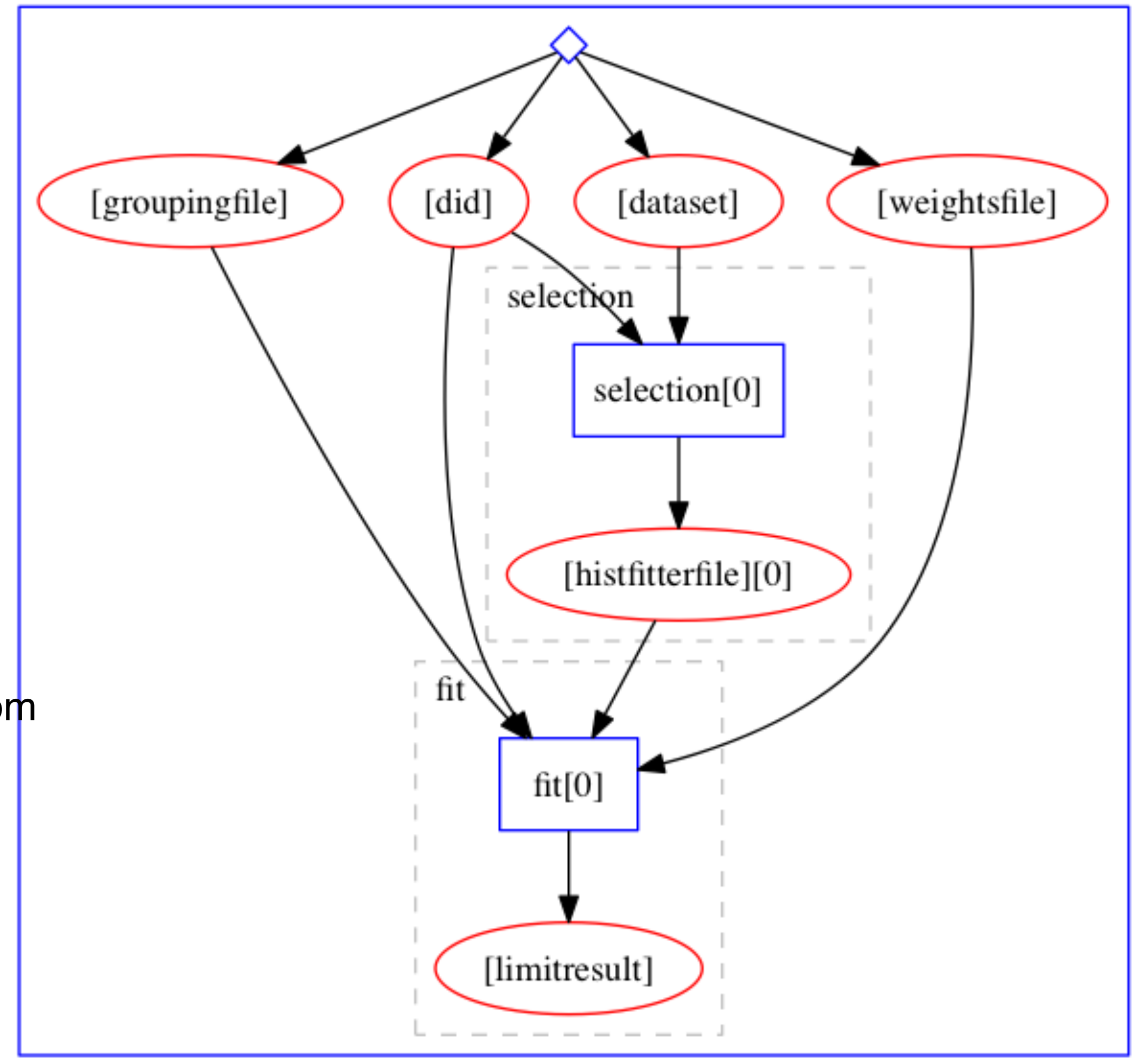


27 lines (26 sloc) | 1.07 KB

```
stages:
  - name: selection
    dependencies: ['init']
    scheduler:
      scheduler_type: singlestep-stage
      parameters:
        dataset: {stages: init, output: dataset, unwrap: true}
        submitdir: '{workdir}/submitdir'
        outputprefix: '{workdir}/histfitter.root'
        did: {stages: init, output: did, unwrap: true}
      step: {$ref: 'selscript.yml#'}
  - name: fit
    dependencies: ['selection']
    scheduler:
      scheduler_type: singlestep-stage
      parameters:
        bkgtree: 'root://eosuser.cern.ch///eos/project/r/recast/Bkg_2.4.15-2-0_merged.root'
        datatree: 'root://eosuser.cern.ch///eos/project/r/recast/Data_2.4.15-2-0.root'
        outputjson: '{workdir}/fitoutput.json'
        pointname: 'Gbb_1600_200'
        gridname: Gbb
        selectionoutput: {stages: selection, output: histfitterfile, unwrap: true}
        weightsfile: {stages: init, output: weightsfile, unwrap: true}
        groupingfile: {stages: init, output: groupingfile, unwrap: true}
        did: {stages: init, output: did, unwrap: true}
      step: {$ref: 'fitscript.yml#'}
```

data and background trees archived in access-controlled location

take signal HF tree from previous step



[groupingfile]
[did]
[dataset]
[weightsfile]
selection
selection[0]
[histfitterfile][0]
fit
fit[0]
[limitresult]