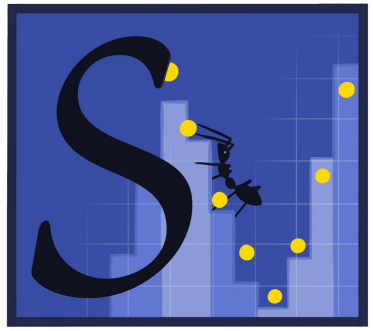


SysVar

Consistent and scalable treatment of systematic uncertainties in template-based analyses

Ilias Tsaklidis, Agrim Aggarwal*, Georgios Alexandris, Tristan Fillinger, Giacomo de Pietro, Daniel Ivanov, Melisa-Melek Akdag, Markus Prim, Florian Bernlochner

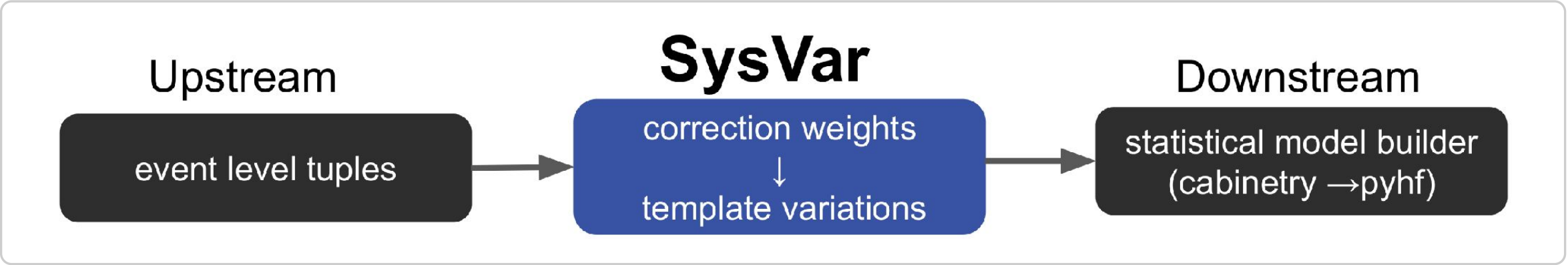
Physikalisches Institut, University of Bonn, 53115 Bonn, Germany
aaggarwa@uni-bonn.de

PyHEP.dev 2026 · 8th September 2026

DOI 10.5281/zenodo.20272326

Template fits with reweighted simulation

- **Binned maximum-likelihood fits** (template fits) to Monte Carlo templates are crucial to measurements where an analytic shape is not available.
- Simulation is corrected with **event-by-event weights** — detector efficiencies, physics modeling, background branching fractions …
- Every correction carries an uncertainty from its measurement that **must be propagated into the templates**.



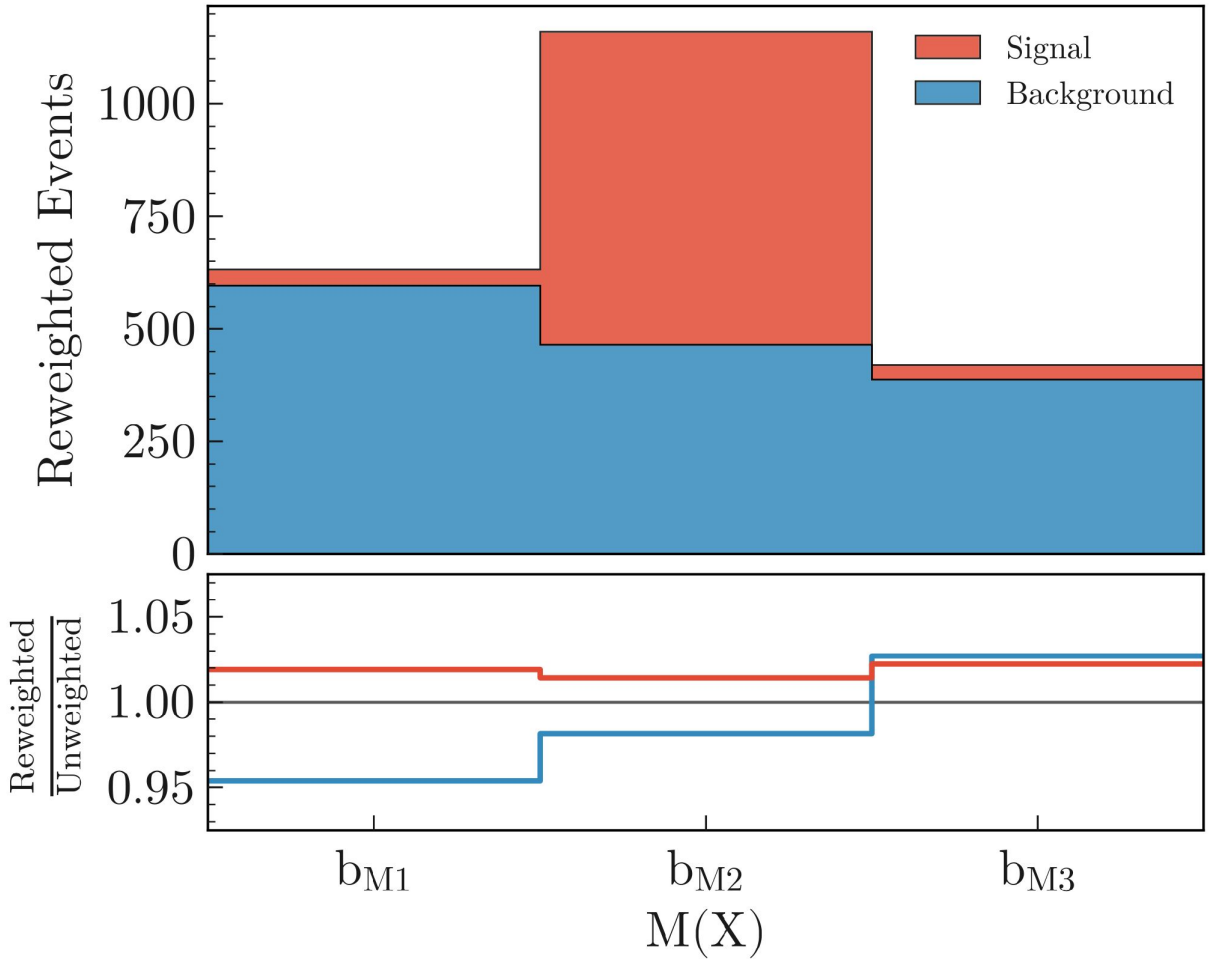
SysVar turns correction weights into template variations, ready for cabinetry → pyhf.

Corrections change normalisation and shape

- Reweighting shifts the **total yield** of each template …
- … and redistributes events across bins — a **shape change**.
- Signal and background respond **differently** to the same correction

A normalisation-only systematic misses this entirely.

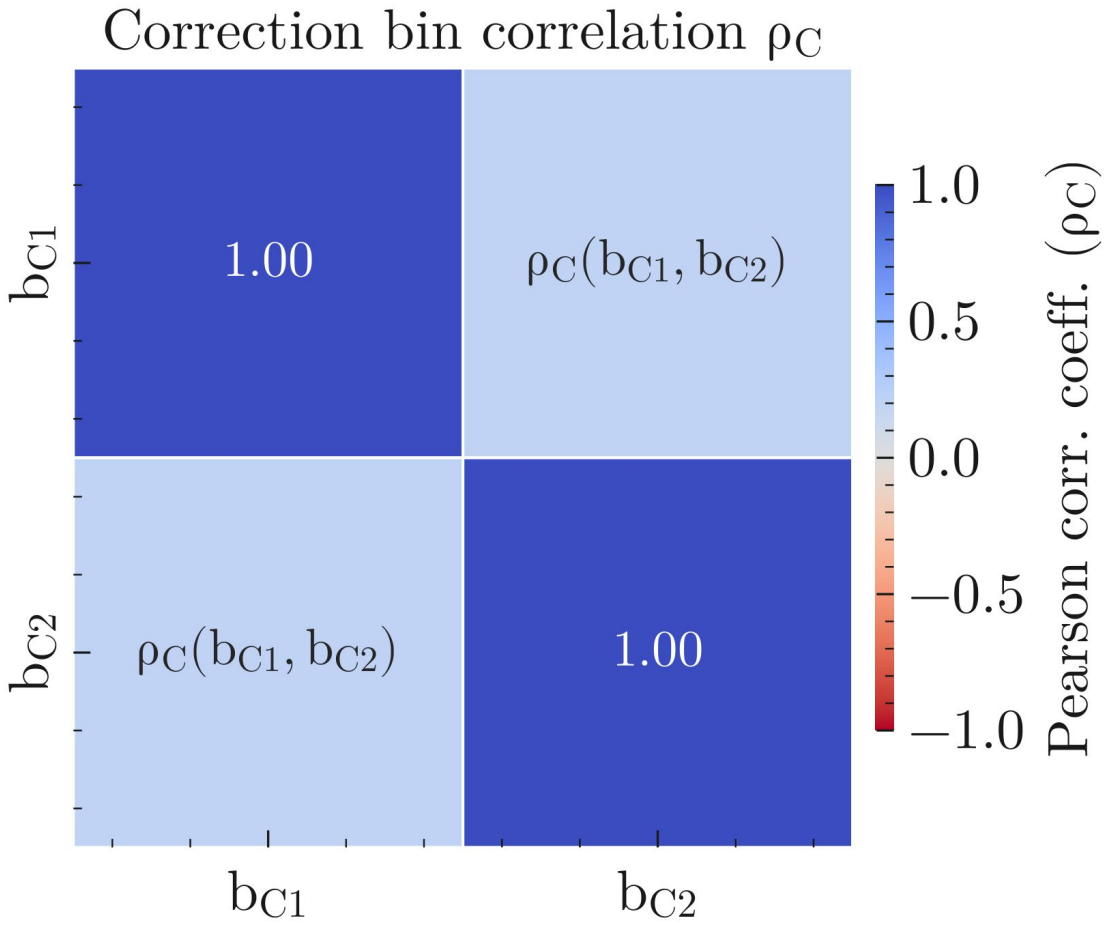
A non trivial shape systematic needed.



Correction uncertainties are correlated

- Corrections are derived from **auxiliary measurements** — e.g. an efficiency calibration binned in the momentum .
- Such measurements are typically **partially correlated** so their uncertainties are expressed with a covariance matrix with correlation:

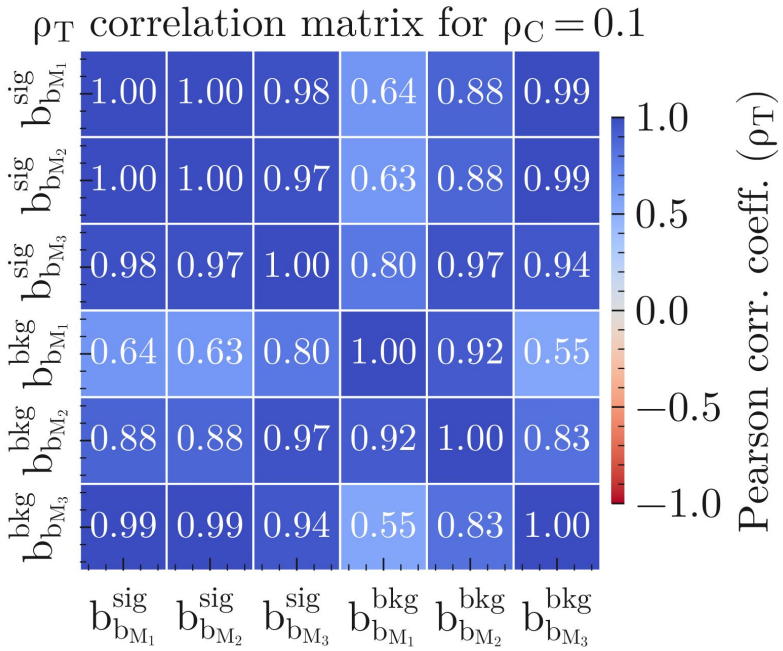
$$\rho_C = \rho\left(b_{C1},\, b_{C2}\right)$$



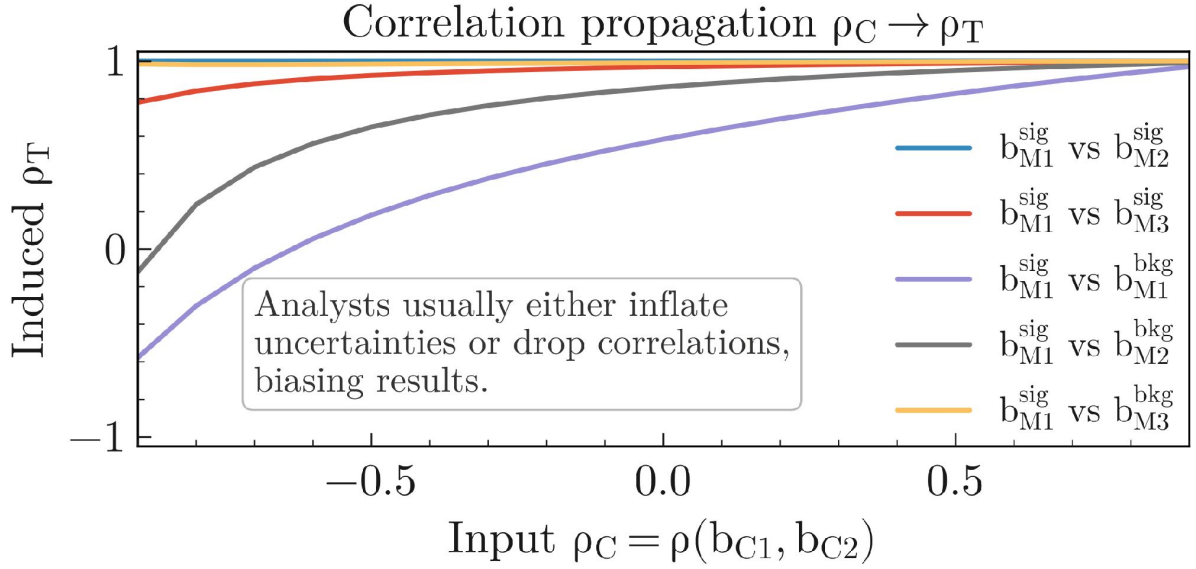
… and they propagate into the templates

- Propagating correlated correction uncertainties to the fit observable induces **partial bin-to-bin correlations** in the final templates — across bins, templates and channels:

$$\rho_T\left(b^{t_p}_{M_i},\ b^{t_q}_{M_j}\right), \qquad t \in \{\text{sig},\ \text{bkg}\}$$



Input ρC **= 0.1 already gives template correlations spanning 0.55 – 1.00.**



The map ρC → ρT **is non-linear and bin-pair dependent — shape specific for each case.**

Standard fitting frameworks cannot express this

Template fits need partially correlated shape systematics — but widely adopted fitters like pyhf only support **fully correlated** or **uncorrelated** nuisance parameters.

In practice analysts fall back on some workarounds (not exhaustive) :

- **Inflate the uncertainties** — safe, but conservative: you pay for it in sensitivity.
- **Drop the correlations** — cheap, but it can **bias the central value**.
- **Nuisance parameter per correction bin** — expensive.

- It also blocks proper **sharing** of a common systematic across experiments in a combination — both its normalisation **and** its shape.

Toys → varied templates → covariance

Generate toys

Sample N_v variations of the correction factors from their full covariance — correlations included.

Build templates

For every toy, rebuild every template, in every channel, for every process.

Form the covariance

Accumulate the relative shifts into the full template-space covariance.

$$C_{\text{syst}}^{ij} = \sum_{v=1}^{N_v} \sum_{c=1}^{N_c} \sum_{s=1}^{N_s} \Gamma_{vcsi}\, \Gamma_{vcsj}$$

Γ_{vcsi} : relative shift in bin i of template s, channel c, toy v (w.r.t. nominal)

C_{syst} lives in the full template space: i, j run over all bins × all templates × all channels.

Eigendecomposition → usable nuisance parameters

- Diagonalise $\mathbf{C_{syst}}$: each eigenvector is a **coherent shift pattern** across the whole template space.

$$C_{\text{syst}} = \sum_k \lambda_k \, \vec{e}_k \, \vec{e}_k^{\,T} \qquad \longrightarrow \qquad \delta_k = \sqrt{\lambda_k} \, \vec{e}_k$$

- Each eigenvariation **δk** is implemented as **one fully correlated shape nuisance parameter**.

SysVar streamlines and automates this entire workflow.

A book-keeper

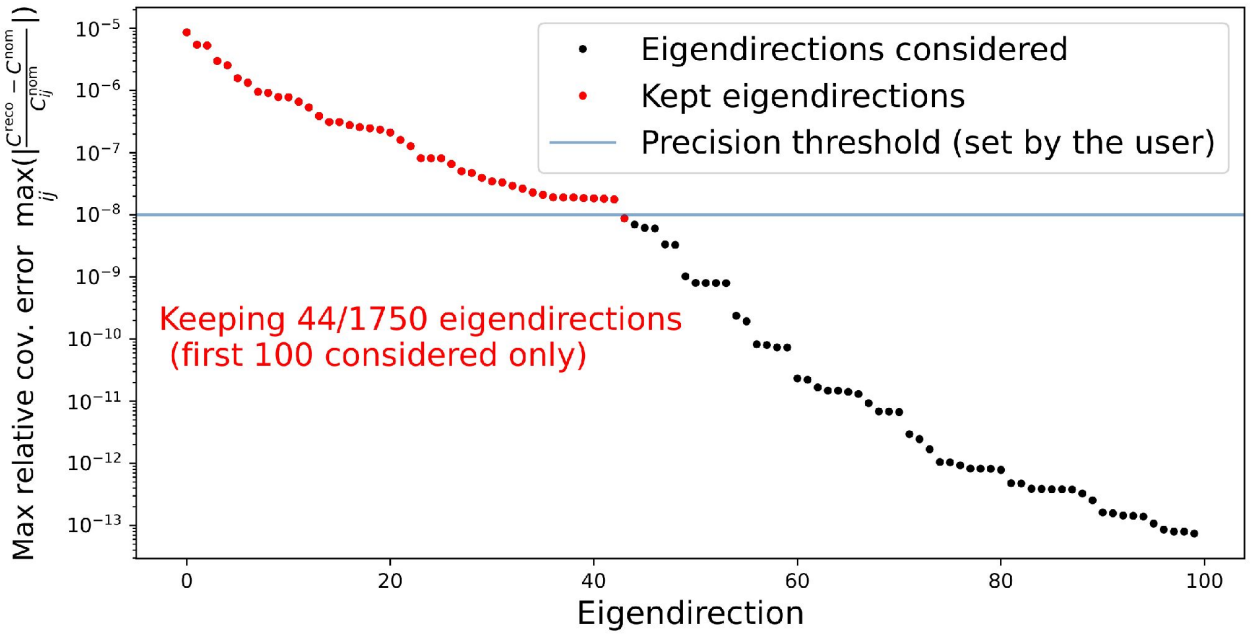
Bridges the gap between performance group providing corrections & analysts at Belle II

Truncation: keep only the modes that matter

- Keeping all eigendirections can be expensive.
- Order them by contribution and keep only those needed to reach a **user-set precision** on the covariance:

$$\max_{ij} \left(\left| \left(C^{\mathrm{reco}} - C^{\mathrm{nom}} \right)_{ij} / C^{\mathrm{nom}}_{ij} \right| \right) < \epsilon_{\mathrm{user}}$$

- Initial guess provided by default. Analysis dependant and different criterias are available



An example from an analysis with 1750 bins in total

Four steps from tuples to eigenvariations

```
from sysvar import add_weights_to_dataframe,
eigendecompose
```

1. Provide corrections

From a CSV file (recommended) or a Python dict.

2. Attach weights to your events

add_weights_to_dataframe(df, correction_source)

3. Configure the analysis declaratively

One config defines channels, templates, observables, binning, weight names and systematic sources.

4. Compress into eigenvariations

eigendecompose(df, config, systematic_source)

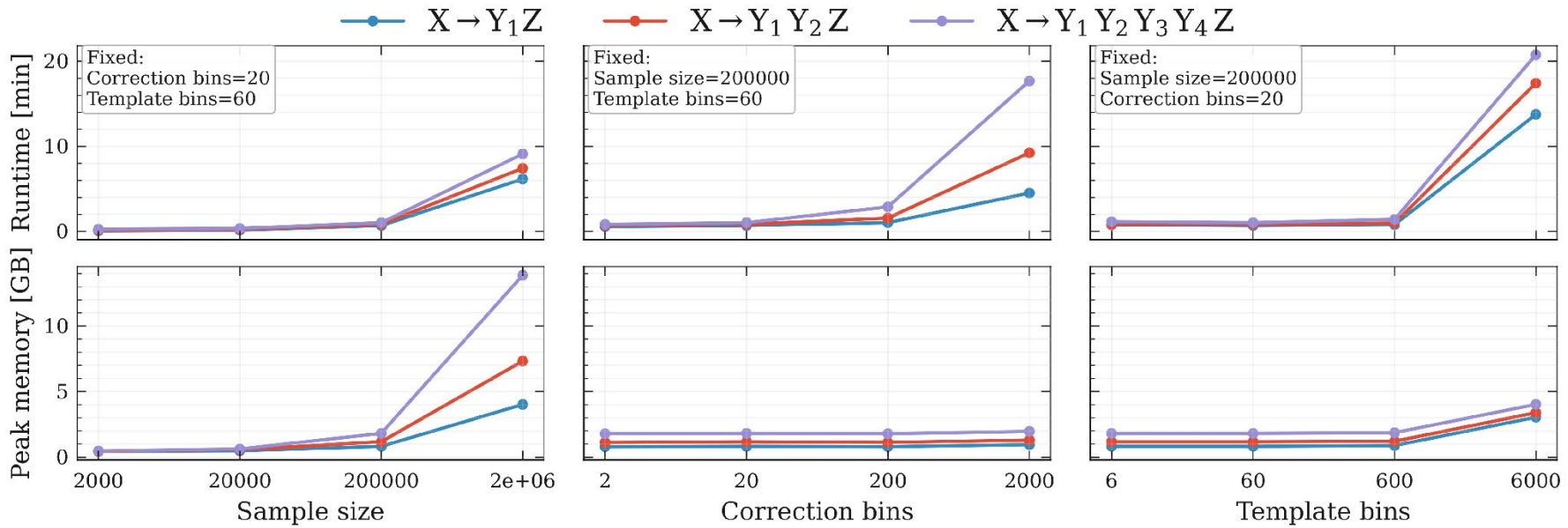
Writes a **ROOT** file in the layout **cabinetry** expects, for the downstream **pyhf** fit.

Full transparency with access to all internal objects and tons of visualization methods

```
config = {
    "output_filepath": "./test_output.root",
    "reco_channel_id_column": "channel",
    "reco_channels": {"channel1": [0], "channel2": [1],},
    "template_id_column": "template",
    "templates": ["signal", "bkg"],
    "total_weight": "total_weight",
    "Nvar": 500,
    "bins": {
        "channel1": {
            "fit_variable1": [1, 2, 3, 4],
        },
        "channel2": {
            "fit_variable1": [0, 0.5, 1],
            "fit_variable2": [1, 2.5, 4],
        },
    },
    "systematics": {
        "systematic_1": {
            "weight": "weight_1",
            "prefices": "particle_1",
            "reco_channels": {"include": ["channel1", "channel2"]},
            "templates": ["signal", "bkg"],
        },
        "systematic_2": {
            "weight": "weight_2",
            "prefices": None,
            "reco_channels": {"exclude": None},
            "templates": ["bkg"],
        },
    },
}
```

Runtime and memory scale manageably

- **Single thread**, scanning sample size, correction bins and template bins for decays with 1, 2 and 4 correction-carrying daughters.



Realistic working point
200k events, 20 correction bins, 60 template bins
~1 min · ~1–2 GB

Largest sample probed
2M events, 4-body decay
~9 min · ~13 GB

Finest binning probed
2000 correction bins / 6000 template bins
~18–21 min

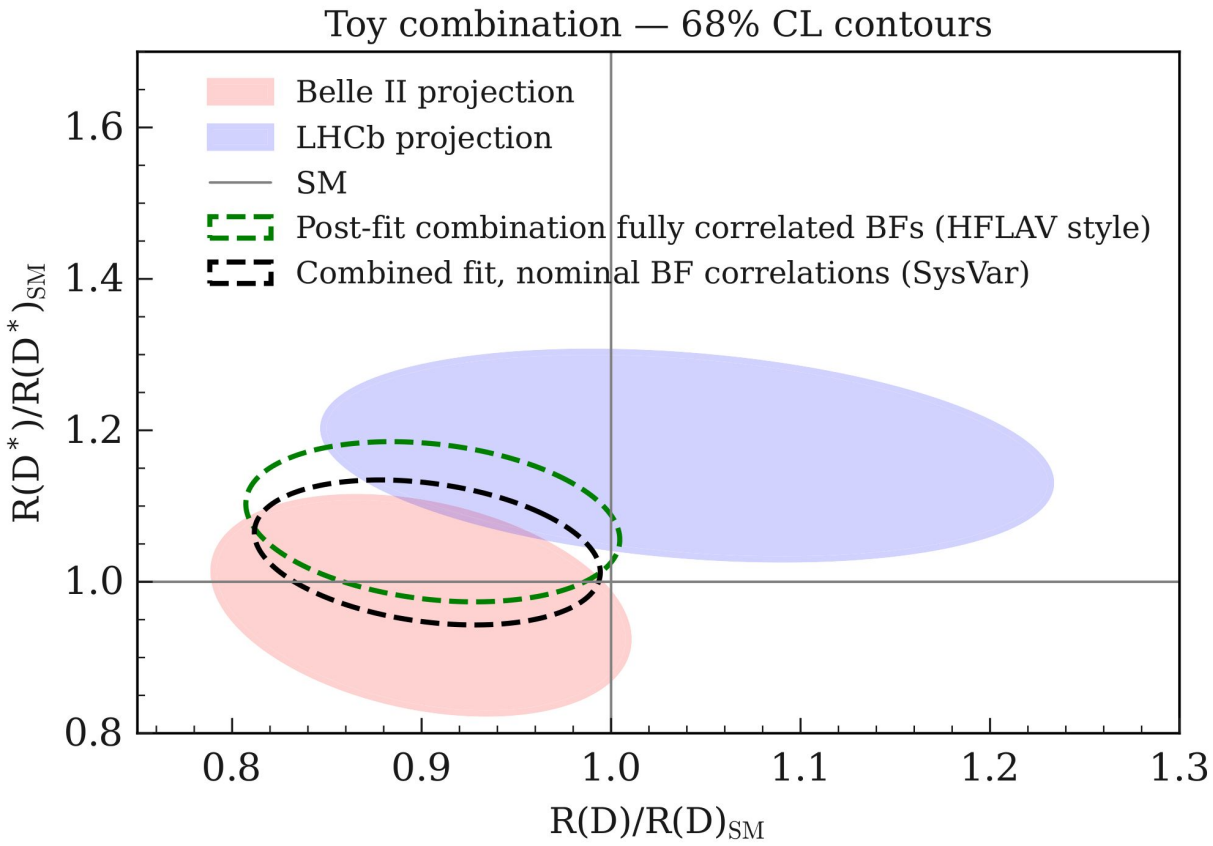
Precisely correlated systematics for precision measurements

- Toy combination of Belle II and LHCb pseudo-measurements of the LFU ratios, based on *Eur. Phys. J. C* **86**, 718 (2026).

$$R(D^{(*)}) = \mathcal{B}(B\to D^{(*)}\tau\nu) / \mathcal{B}(B\to D^{(*)}\ell\nu)$$

- Shared systematic: **common background branching fractions**, with a moderate assumed correlation model:

	BF_1	BF_2	BF_3
BF_1	1	0.05	−0.35
BF_2	0.05	1	0.10
BF_3	−0.35	0.10	1



Takeaway

Sharing and correlating the normalisation **and shape** of common systematics shifts both the combined **central value** and the **uncertainty** relative to ad-hoc correlation strategies.

Maintenance and future prospects

TODAY

We maintain SysVar for the Belle II collaboration.

Performance group
derives the corrections

↓

SysVar
one format · one consistent application

↓

Analysts
build templates and fit

Bridging the gap — one convention.

OUTLOOK

Correlate upstream

Diagonalise the correction covariance itself — provide analysts the eigenvariations.

Beyond Belle II

Feedback from other experiments to keep SysVar collaboration-agnostic.

Faster core

Architecture improvements for speed and memory.

Summary

The problem

Corrections to MC induce partially correlated shape systematics that HistFactory-style frameworks cannot express.

The method

Toys → varied templates → full template-space covariance → eigenvariations as fully correlated shape NPs → truncation.

The software

Two function calls and one declarative config; output is a cabinetry-ready ROOT file for pyhf.

The impact

On a realistic R(D)/R(D*) combination, correct treatment of shared systematics shifts both the central value and the uncertainty.

Collaboration agnostic

Built as part of the Belle II collaboration but it's a high energy physics analysis problem, therefore we try to build it as generic as possible

References: pyhf · cabinetry · *Eur. Phys. J. C* **86**, 718 (2026)

Get SysVar

Documentation

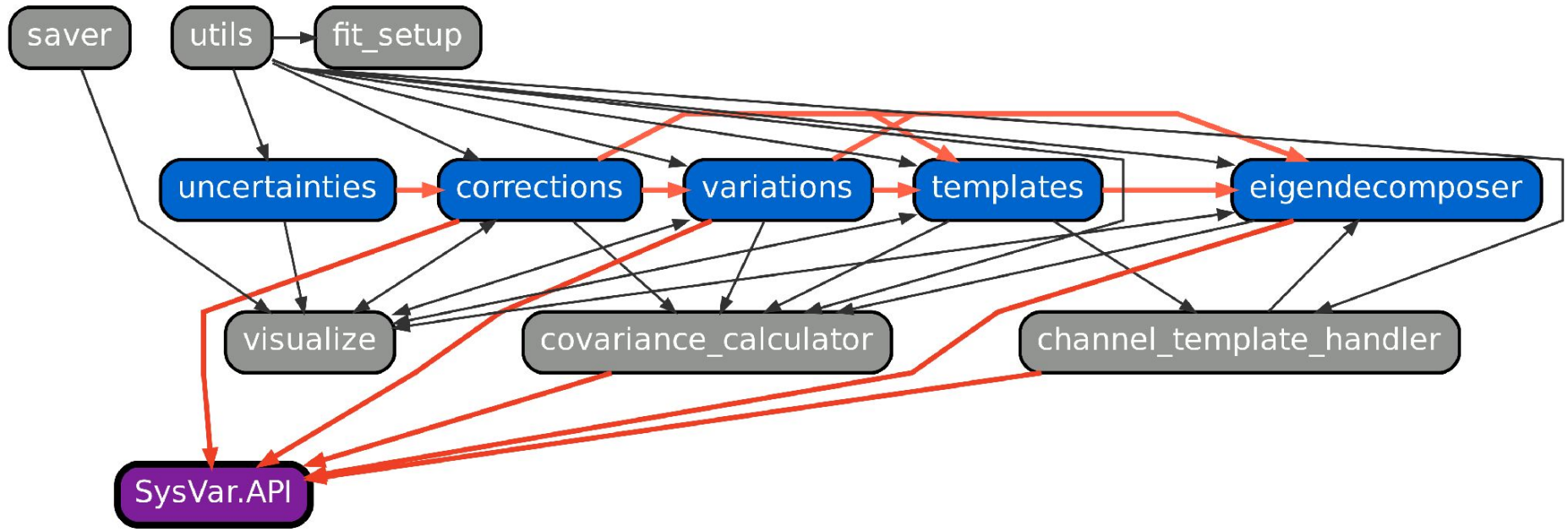
DOI 10.5281/zenodo.20272326

10.5281/zenodo.20272326

Backup

SysVar: a lightweight API over a modular core

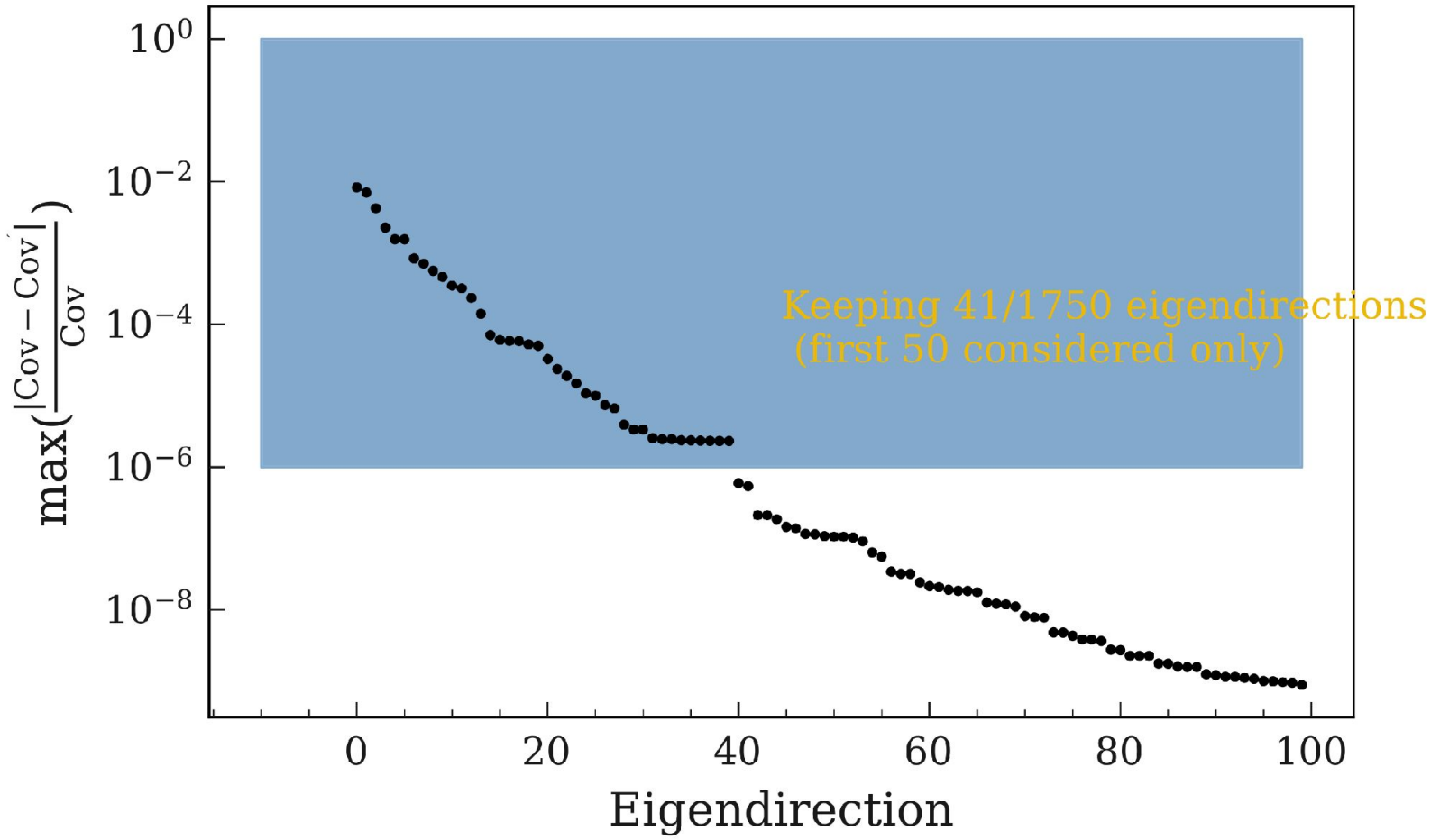
- A lightweight API abstracts away the book-keeping; internal components communicate via **class composition**.



Blue: the main pipeline — **uncertainties → corrections → variations → templates → eigendecomposer**.
Purple: the single public entry point.

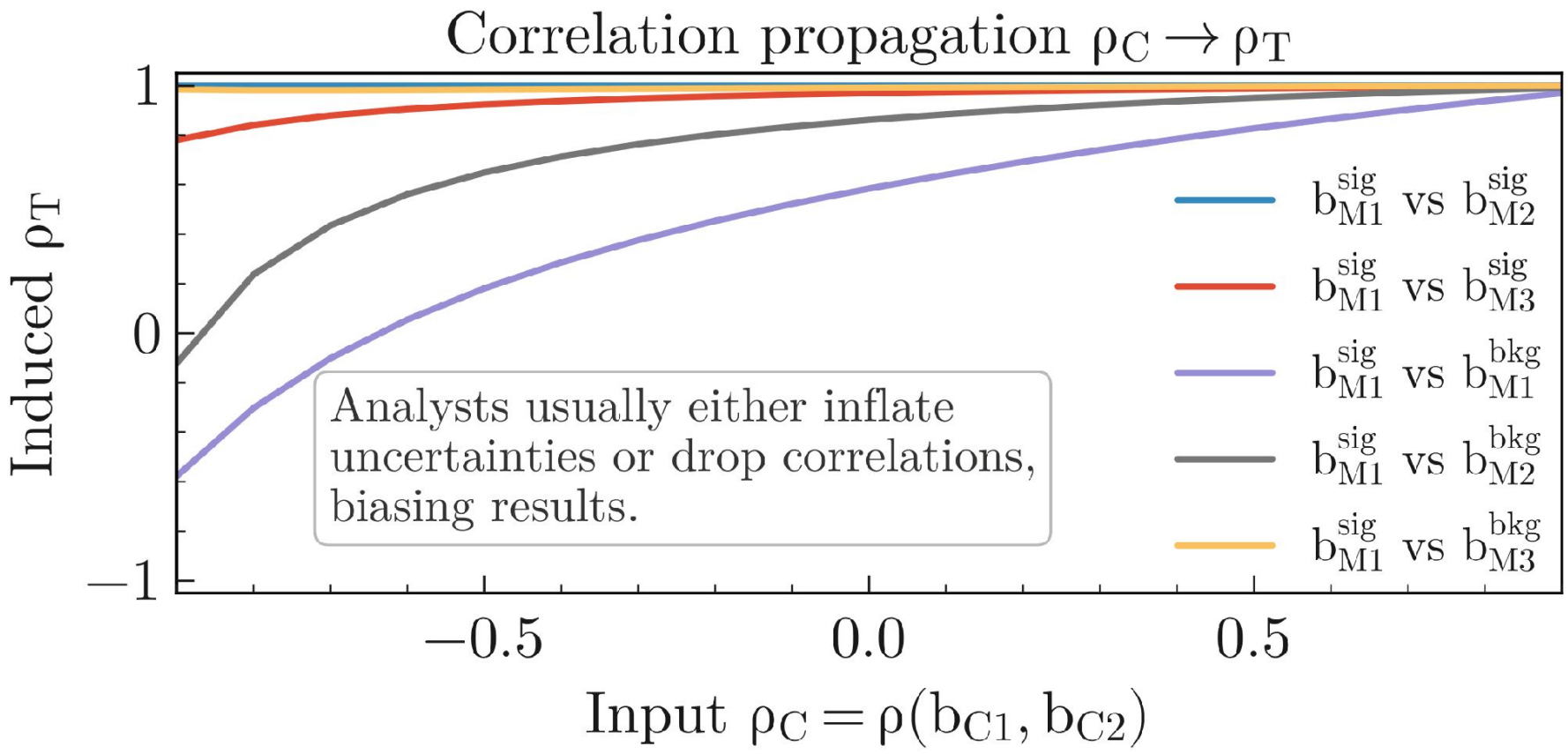
Output: a **ROOT** file following **cabinetry** conventions — straight into a **pyhf** fit.

Truncation — alternative working point



Looser threshold (10^{-6}, first 50 modes considered): **41 / 1750** kept — the retained count is insensitive to the threshold over orders of magnitude.

Correlation propagation ρ_C → ρ_T in detail



Signal–signal pairs saturate near +1; signal–background pairs swing from strongly negative to +1 — one ad-hoc correlation assumption cannot cover both.