SysVar:
A New Tool for Enhancing Consistency in the Treatment of Systematics

Agrim Aggarwal, Stefanie Meinert, **Ilias Tsaklidis**
Felix Metzner, Markus Prim, Slavomira Stefkova
Florian Bernlochner, Jochen Dingfelder

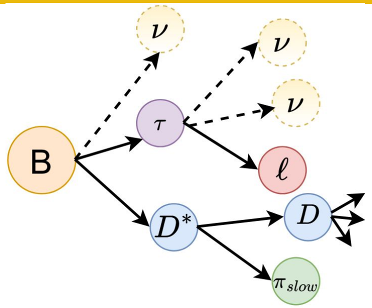
itsaklid@uni-bonn.de,
s6agagga@uni-bonn.de

09.09.2025
Belle II Germany meeting 2025

UNIVERSITÄT BONN
BLUB
Belle group
University of Bonn
Federal Ministry of Education and Research
Belle II

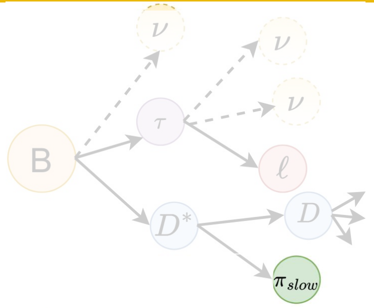
Is there a consistent way of handling partial correlations arising from systematic effects which affect shapes in simultaneous template fits ?

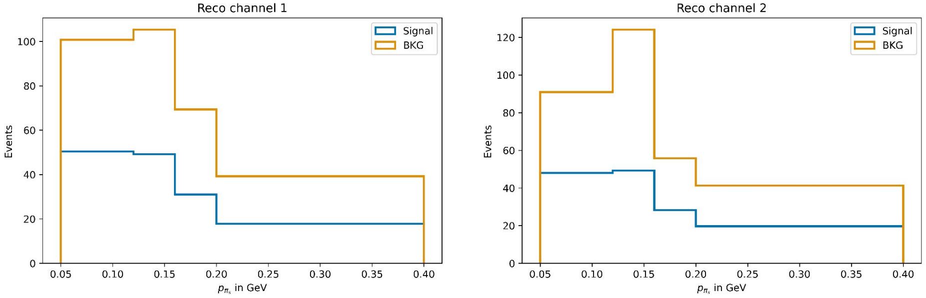
An example of π_{slow} efficiency systematics



The track finding efficiency of π_{slow} typically differs on Data and MC.
We correct those in bins of the momentum of π_{slow}

An example of π_{slow} efficiency systematics



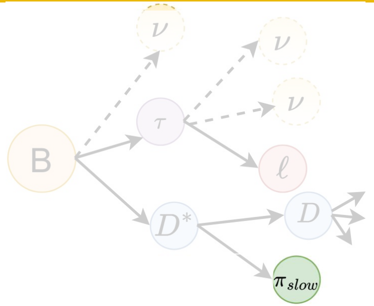


Apply $\frac{\epsilon^i_{\text{Data}}}{\epsilon^i_{\text{MC}}}$ factors as correction weights on MC with i the π_{slow} momentum bin

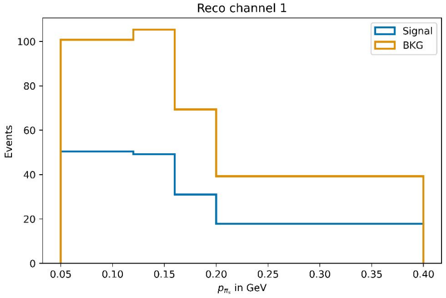
Table 18: Correction tables for $\pi_s^\pm$ efficiency corrections derived by the Tracking group.

$\pi_s^\pm$ momentum	0.05-0.12	0.12-0.16	0.16-0.20
value	0.996	0.990	0.987
stat. unc. correlated	0.015	0.015	0.015
stat. unc. uncorrelated	0.022	0.017	0.019
syst. unc.	0.003	0.003	0.003

An example of π_{slow} efficiency systematics



Apply $\frac{\epsilon^i_{\mathrm{Data}}}{\epsilon^i_{\mathrm{MC}}}$ factors as correction weights on MC with i the π_{slow} momentum bin



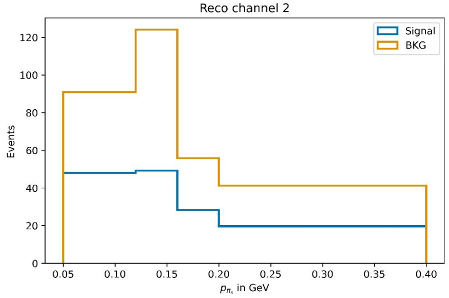
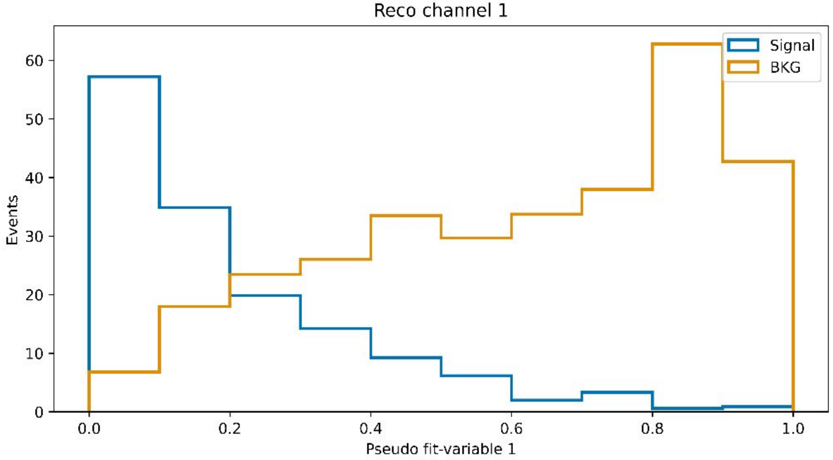


Table 18: Correction tables for $\pi_s^{\pm}$ efficiency corrections derived by the Tracking group.

$\pi_s^{\pm}$ momentum	0.05-0.12	0.12-0.16	0.16-0.20
value	0.996	0.990	0.987
stat. unc. correlated	0.015	0.015	0.015
stat. unc. uncorrelated	0.022	0.017	0.019
syst. unc.	0.003	0.003	0.003



How do these correction weights affect the signal extraction variable ?

Sysvar

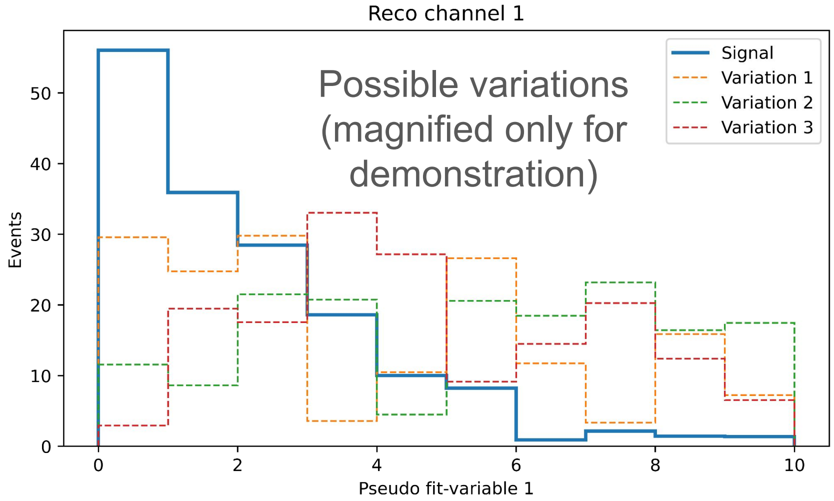
The MC corrections that we apply can have two effects on the signal extraction variable

1. Overall normalization (trivial to calculate)
2. Shape effect (non trivial especially in simultaneous fits)

Sysvar

The MC corrections that we apply can have two effects on the signal extraction variable

1. Overall normalization (trivial to calculate)
2. Shape effect (non trivial especially in simultaneous fits)



Sysvar

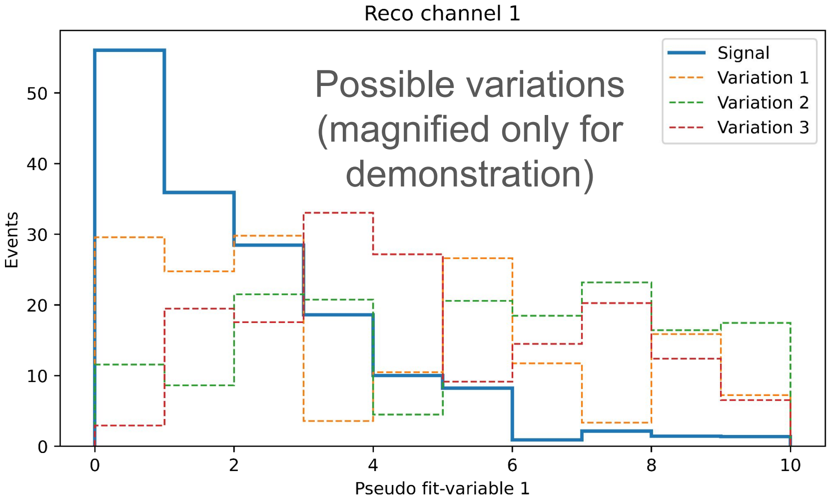
The MC corrections that we apply can have two effects on the signal extraction variable

1. Overall normalization (trivial to calculate)
2. Shape effect (non trivial especially in simultaneous fits)

In case no arbitrary correlation matrices can be used in the fitter to **properly correlate the different bins** (like in pyhf)
one has to

1. Build a **covariance matrix** between all the bins
   Dimensions: (channels x templates x bins)
2. **Diagonalize** it
3. Calculate the **eigenvariations** of those.

Every eigendirection is then implemented as
a fully correlated nuisance parameter across all bins.



Sysvar

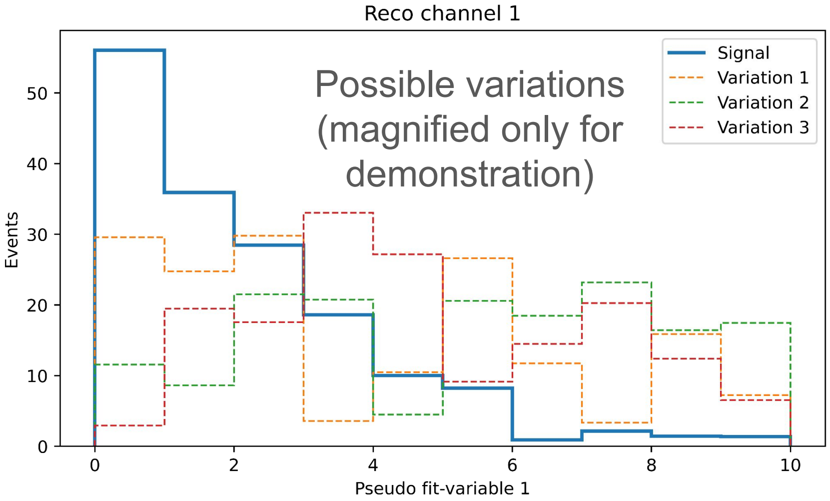
The MC corrections that we apply can have two effects on the signal extraction variable

1. Overall normalization (trivial to calculate)
2. Shape effect (non trivial especially in simultaneous fits)

In case no arbitrary correlation matrices can be used in the fitter to **properly correlate the different bins** (like in pyhf) one has to

1. Build a **covariance matrix** between all the bins
   Dimensions: (channels x templates x bins)
2. **Diagonalize** it
3. Calculate the **eigenvariations** of those.

Every eigendirection is then implemented as **a fully correlated nuisance parameter across all bins.**



$$C^{ij}_{syst} = \sum_{v=1}^{N_v} \sum_{c=1}^{N_c} \sum_{s=1}^{N_s} \Gamma_{vcsi}\Gamma_{vcsj} \tag{2}$$

where Γ_{vcsi} describes the variation v of bin i in template s in reconstruction channel c,

$$C_{syst} = VUV^T = (V\sqrt{U})(V\sqrt{U})^T = \Gamma'\Gamma'^T \tag{3}$$

where V is a matrix describing the eigenvectors and U is the diagonalized matrix with the eigenvalues. Now the eigenvariation is defined as

$$\Gamma' = V\sqrt{U} \tag{4}$$

Sysvar

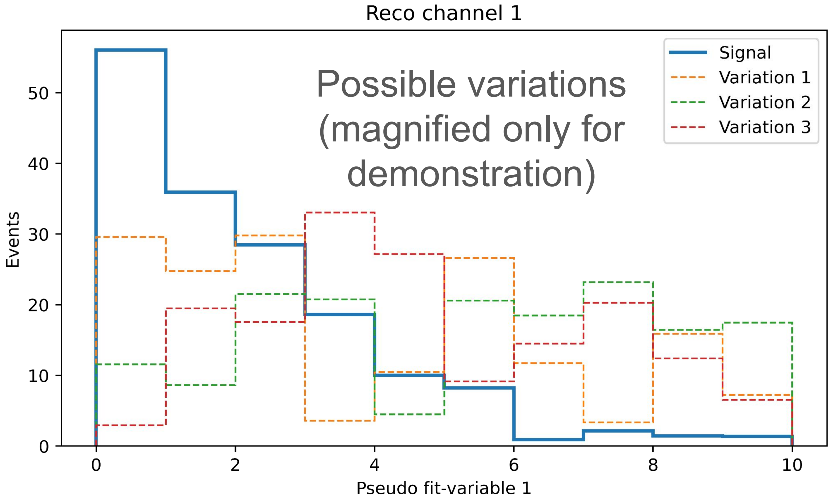
The MC corrections that we apply can have two effects on the signal extraction variable

1. Overall normalization (trivial to calculate)
2. Shape effect (non trivial especially in simultaneous fits)

In case no arbitrary correlation matrices can be used in the fitter to **properly correlate the different bins** (like in pyhf) one has to

1. Build a **covariance matrix** between all the bins
   Dimensions: (channels x templates x bins)
2. **Diagonalize** it
3. Calculate the **eigenvariations** of those.

Every eigendirection is then implemented as **a fully correlated nuisance parameter across all bins.**



$$C^{ij}_{syst} = \sum_{v=1}^{N_v} \sum_{c=1}^{N_c} \sum_{s=1}^{N_s} \Gamma_{vcsi} \Gamma_{vcsj} \tag{2}$$

where Γ_{vcsi} describes the variation v of bin i in template s in reconstruction channel c,

$$C_{syst} = VUV^T = (V\sqrt{U})(V\sqrt{U})^T = \Gamma' \Gamma'^T \tag{3}$$

where V is a matrix describing the eigenvectors and U is the diagonalized matrix with the eigenvalues. Now the eigenvariation is defined as

$$\Gamma' = V\sqrt{U} \tag{4}$$

SysVar is streamlining this procedure

SysVar

A New Tool for Enhancing Consistency in the Treatment of Systematic Uncertainties

SysVar: A tool enhancing consistency in the treatment of systematics

SysVar is a python based tool that allows to:

1. Apply Data/MC corrections to a DataFrame

SysVar: A tool enhancing consistency in the treatment of systematics

SysVar is a python based tool that allows to:

1. Apply Data/MC corrections to a DataFrame
2. Generate Variations of Data/MC Corrections

SysVar: A tool enhancing consistency in the treatment of systematics

SysVar is a python based tool that allows to:

1. Apply Data/MC corrections to a DataFrame
2. Generate Variations of Data/MC Corrections
3. Histogram MC to build templates and template variations for a non-parametric fit.

SysVar: A tool enhancing consistency in the treatment of systematics

SysVar is a python based tool that allows to:

1. Apply Data/MC corrections to a DataFrame
2. Generate Variations of Data/MC Corrections
3. Histogram MC to build templates and template variations for a non-parametric fit.
4. Diagonalize a cov matrix (channels x templates x bins) to produce Eigenvariations and save them to implement Correlated Shape Variations

SysVar: A tool enhancing consistency in the treatment of systematics

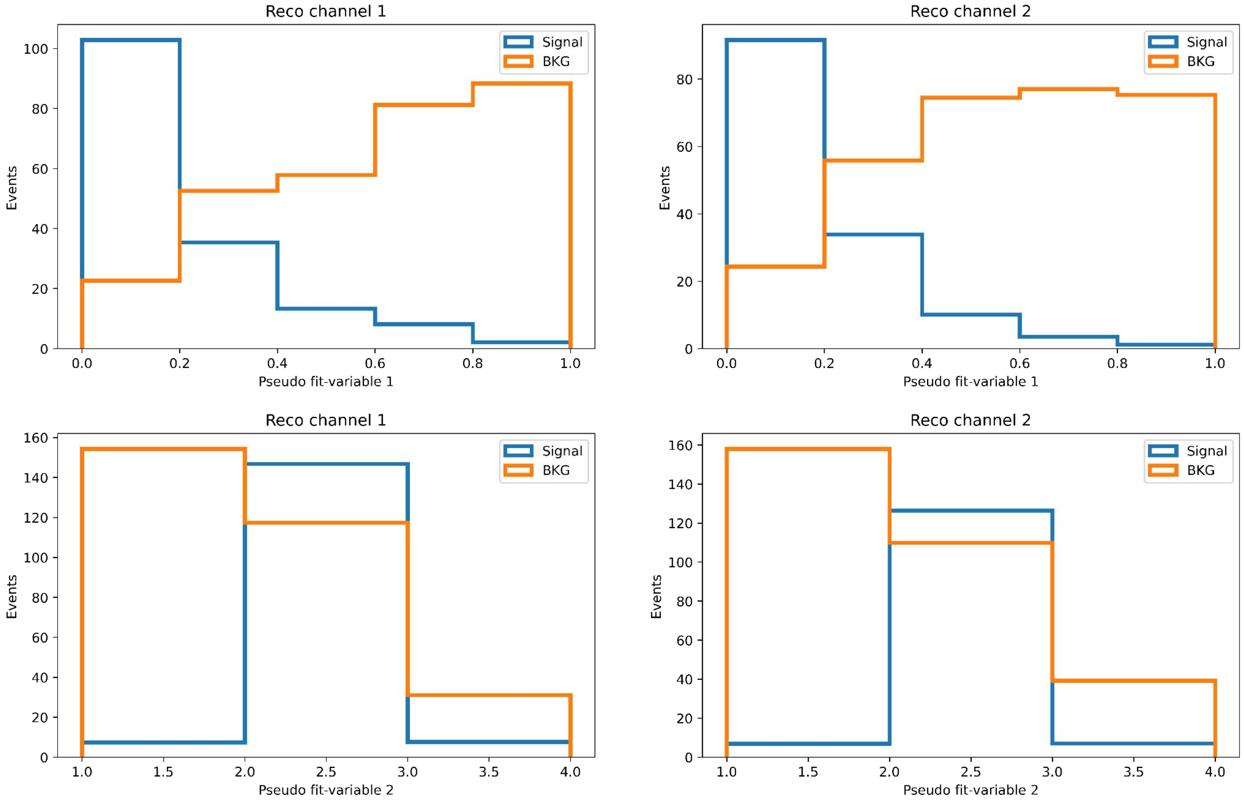
SysVar is a python based tool that allows to:

1. Apply Data/MC corrections to a DataFrame
2. Generate Variations of Data/MC Corrections
3. Histogram MC to build templates and template variations for a non-parametric fit.
4. Diagonalize a cov matrix (channels x templates x bins) to produce Eigenvariations and save them to implement Correlated Shape Variations
5. Identify the number of necessary Eigendirections that the user should consider for an accurate analysis (different criteria are available)

How to use SysVar

User interface example

For demonstration let's consider a 2D simultaneous fit in two reconstructions channels



Slow π efficiency systematics need to be considered

Adding weights to dataframe

	channel	template	slow_pi_p	fit_variable1	fit_variable2	other_weight
0	0	bkg	0.153206	0.663650	1.023131	0.653335
4	0	signal	0.181029	0.029307	2.753247	0.297398
5	1	signal	0.109792	0.020703	2.265813	0.263402
6	0	signal	0.117345	0.068399	2.646895	0.295782
7	1	signal	0.210761	0.047100	2.054302	0.324007
...	...	...	...	...	...	...
1995	0	bkg	0.223465	0.986080	2.976878	0.767080
1996	1	signal	0.205902	0.469632	2.136452	0.269193
1997	0	signal	0.144924	0.003711	2.630525	0.309213
1998	1	signal	0.176194	0.058186	2.847713	0.365680
1999	1	bkg	0.179354	0.050963	1.594758	0.700421

1767 rows × 8 columns

Adding weights to dataframe

```
from sysvar import add_weights_to_dataframe

add_weights_to_dataframe(
    df = toy_df,
    systematic= "charged_slow_pi",
    MC_production= "MC15rd",
    prefix= "slow_pi",
    weightname ="charged_weight",
    #overwrite: False,
    #Nvar: 0
)
toy_df
```

```
WARNING : load_covariance_matrix: 275 :  Explicit covariance matrix was not found in config under None or cov_matrix_path and will be built from the specified uncertainties.
INFO : read_corrections: 388 :  Loading correction values from config array.
INFO : add_weights_to_dataframe: 84 :  slow_pi_charged_weight does not exist. Adding it to dataframe
```

	channel	template	slow_pi_p	fit_variable1	fit_variable2	other_weight	slow_pi_weight
0	0	bkg	0.153206	0.663650	1.023131	0.653335	0.990
4	0	signal	0.181029	0.029307	2.753247	0.297398	0.987
5	1	signal	0.109792	0.020703	2.265813	0.263402	0.996
6	0	signal	0.117345	0.068399	2.646895	0.295782	0.996
7	1	signal	0.210761	0.047100	2.054302	0.324007	1.000
...	...	...	...	...	...	...	...
1995	0	bkg	0.223465	0.986080	2.976878	0.767080	1.000
1996	1	signal	0.205902	0.469632	2.136452	0.269193	1.000
1997	0	signal	0.144924	0.003711	2.630525	0.309213	0.990
1998	1	signal	0.176194	0.058186	2.847713	0.365680	0.987
1999	1	bkg	0.179354	0.050963	1.594758	0.700421	0.987

1767 rows × 8 columns

Performing the EigenDecomposition

```
from sysvar import eigendecompose
eigedecomposer_object = eigendecompose(
    df= df,
    settings= settings,
    syst_effect= "charged_slow_pi",
    #criterion: "cov_diff",
    #prc: 0.005,
    #save_variations: False
)
```

```
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel1 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel2 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
```

Performing the EigenDecomposition

```python
from sysvar import eigendecompose
eigedecomposer_object = eigendecompose(
    df= df,
    settings= settings,
    syst_effect= "charged_slow_pi",
    #criterion: "cov_diff",
    #prc: 0.005,
    #save_variations: False
)
```

```
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel1 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel2 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
```

```yaml
input_filepath: ./test_input.root
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
MC_prod: MC15rd
Nvar: 500

bins:
  channel1:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]
  channel2:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]

systematics:
  charged_slow_pi:
    weight: "charged_weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude:
    templates:

  neutral_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude:
    templates:
```

Performing the EigenDecomposition

```python
from sysvar import eigendecompose
eigedecomposer_object = eigendecompose(
    df= df,
    settings= settings,
    syst_effect= "charged_slow_pi",
    #criterion: "cov_diff",
    #prc: 0.005,
    #save_variations: False
)
```

```
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel1 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel2 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
```

```yaml
input_filepath: ./test_input.root
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
MC_prod: MC15rd
Nvar: 500

bins:
  channel1:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]
  channel2:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]

systematics:
  charged_slow_pi:
    weight: "charged_weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude:
    templates:

  neutral_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude:
    templates:
```

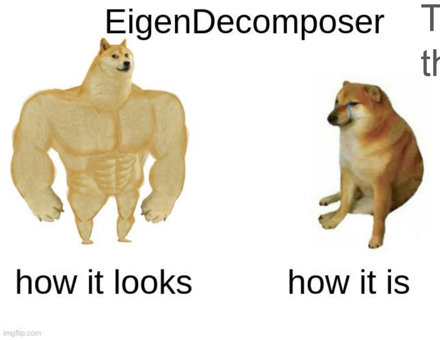
Templates are being saved in the format expected by cabinetry → pyhf ∂ifferentiable ℒikelihoods

Performing the EigenDecomposition

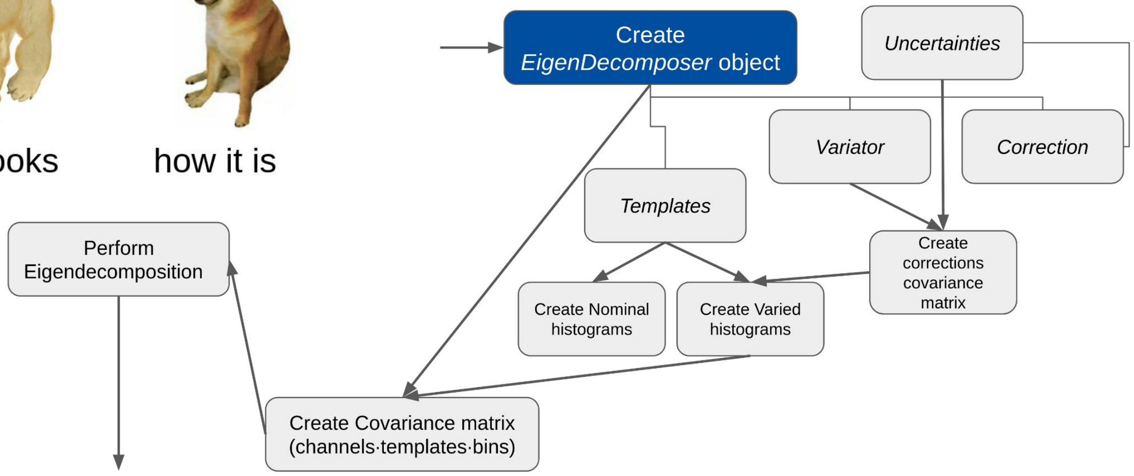
```python
from sysvar import eigendecompose
eigedecomposer_object = eigendecompose(
    df= df,
    settings= settings,
    syst_effect= "charged_slow_pi",
    #criterion: "cov_diff",
    #prc: 0.005,
    #save_variations: False
)
```

```
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel1 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
INFO : _create_varied_templates: 235 :  ########## Reco channel: channel2 ##########
INFO : _create_varied_templates: 266 :  Building Template2D for signal
INFO : _create_varied_templates: 266 :  Building Template2D for bkg
```

```yaml
input_filepath: ./test_input.root
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
MC_prod: MC15rd
Nvar: 500

bins:
  channel1:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]
  channel2:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]

systematics:
  charged_slow_pi:
    weight: "charged_weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude:
    templates:

  neutral_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude:
    templates:
```

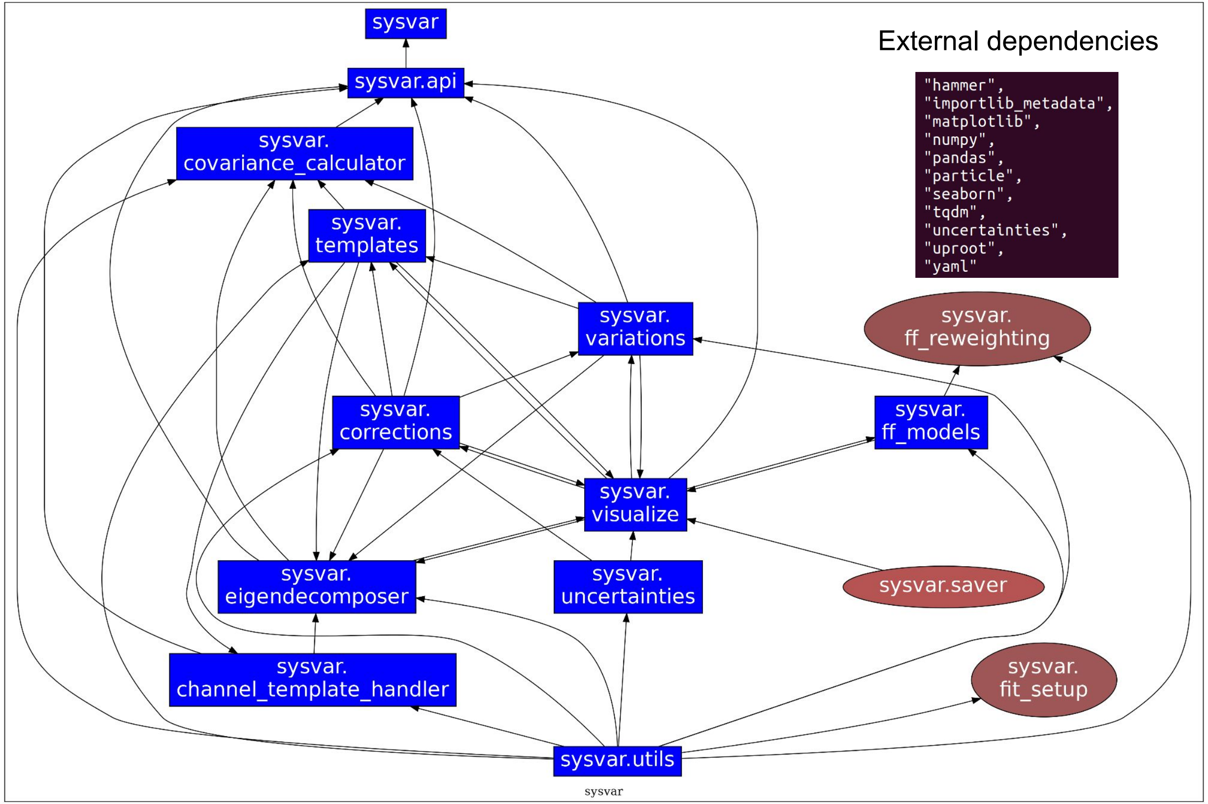


Templates are being saved in the format expected by cabinetry → pyhf



Modules overview

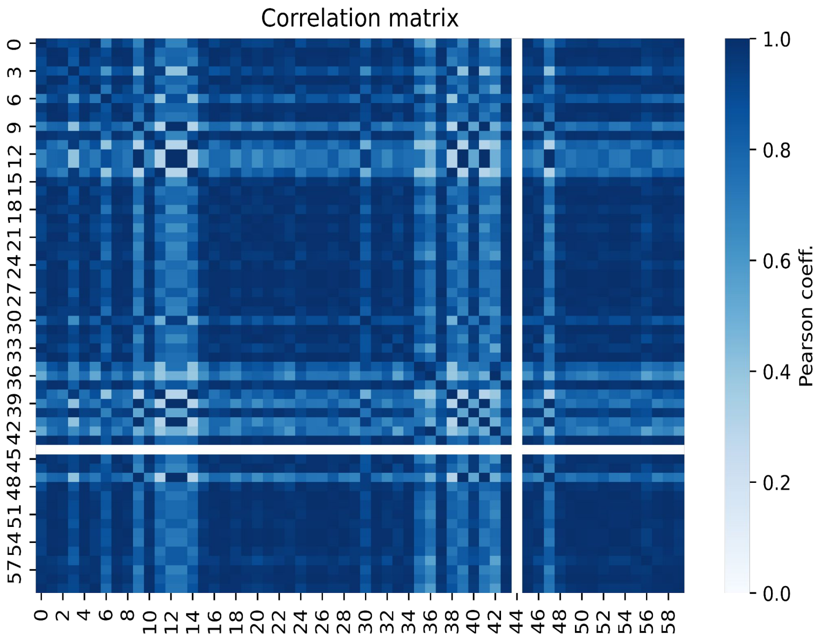
Language	files	blank	comment	code
Python	15	1385	1187	4667
SUM:	15	1385	1187	4667



Conceptually inspired by PIDVar but implemented using **class composition** and **shallow inheritance** design patterns

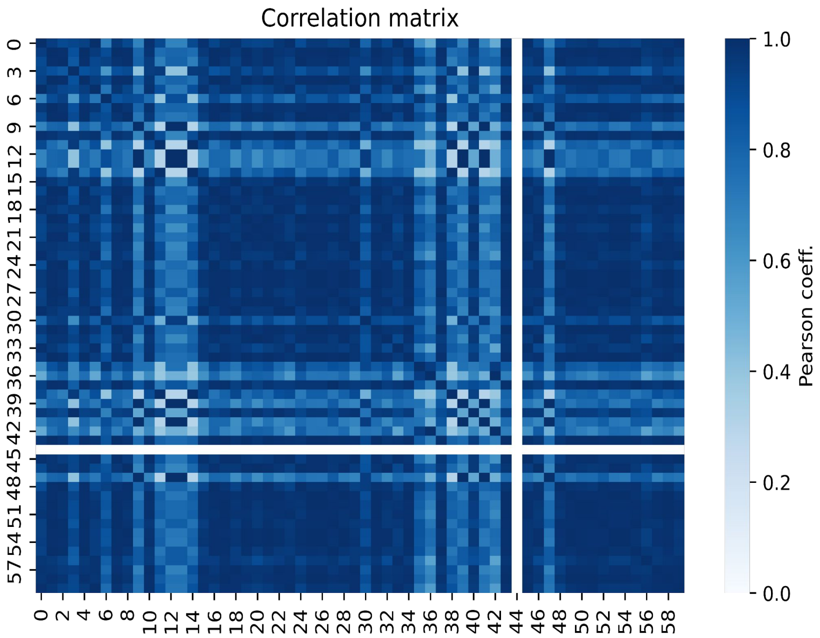
The API is hiding much of this complexity

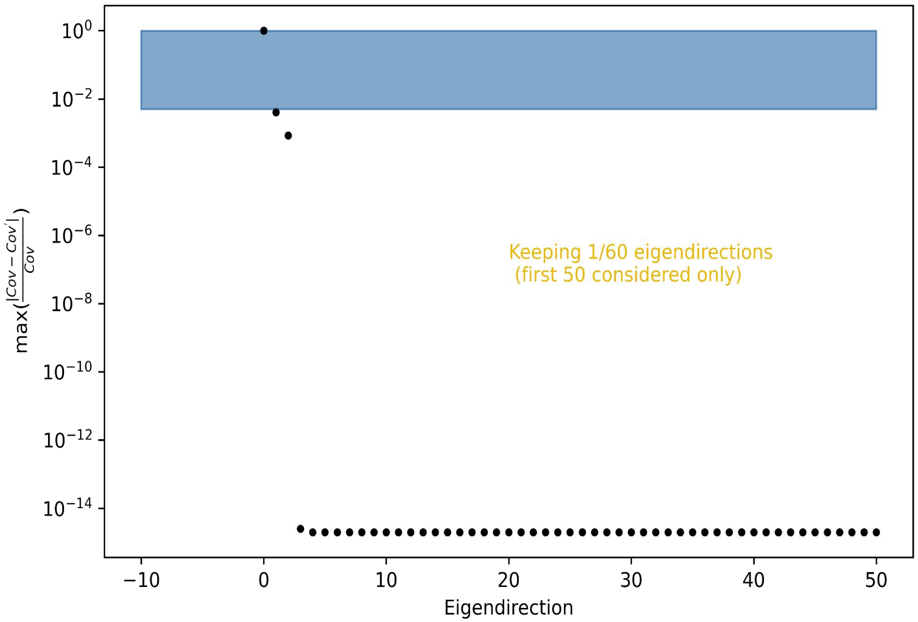
High Level Visualization API



from sysvar import **plot_analysis_corr_matrix**
plot_analysis_corr_matrix (eigendecomposer_object)

High Level Visualization API



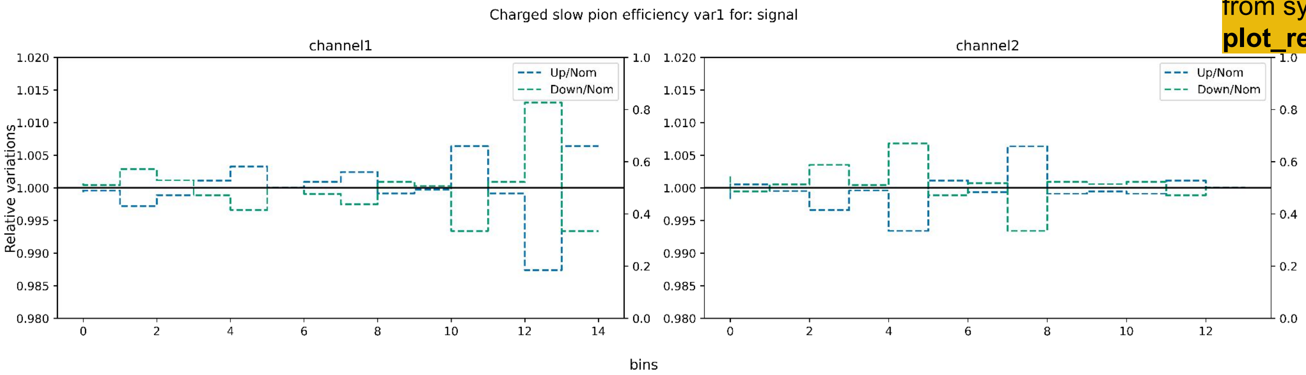


from sysvar import **plot_analysis_corr_matrix**
plot_analysis_corr_matrix (eigendecomposer_object)

from sysvar import **plot_cov_diff**
plot_cov_diff(eigendecomposer_object)

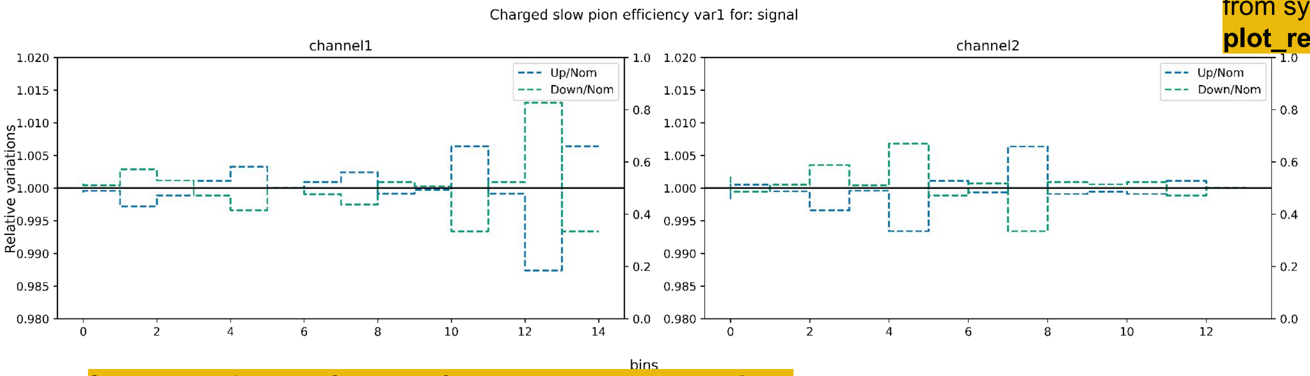
Diagnostic tools for templates

from sysvar import **plot_templates_relative_variations**
plot_relative_variations(eigendecomposer_object)

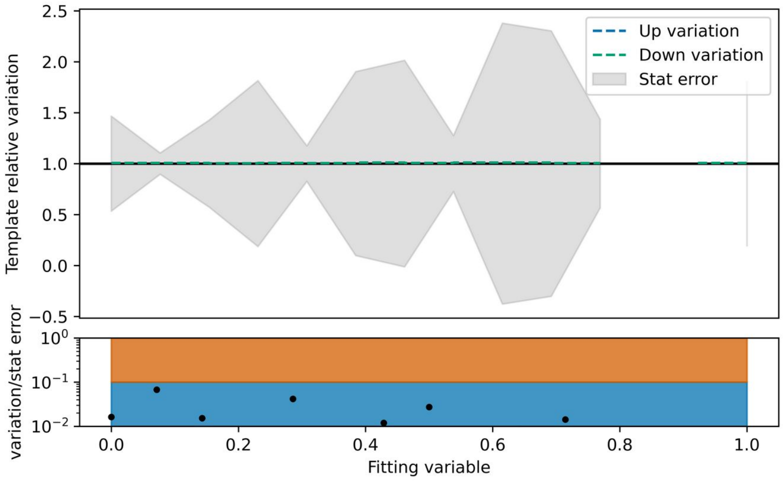


Diagnostic tools for templates

from sysvar import **plot_templates_relative_variations**
plot_relative_variations(eigendecomposer_object)

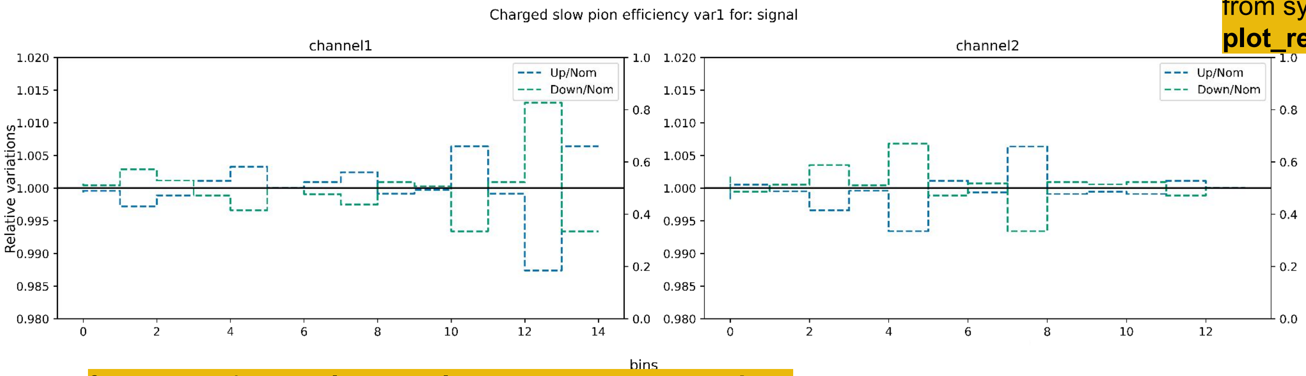


from sysvar import **plot_templates_staterror_comparison**
plot_templates_staterror_comparison(eigendecomposer_object)

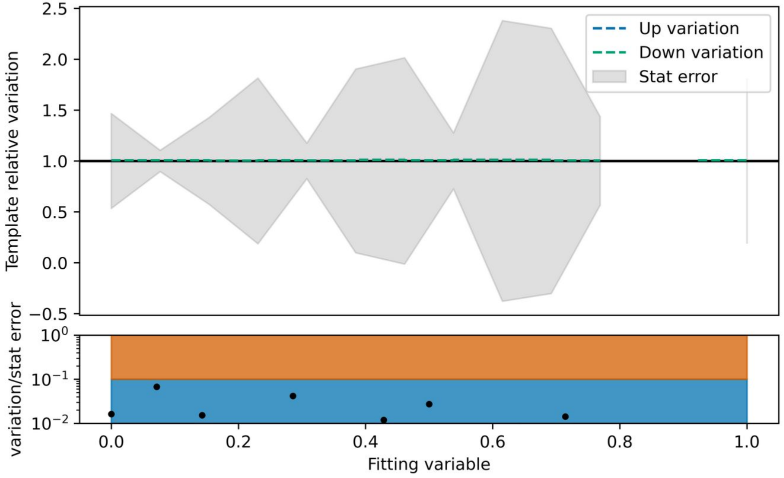


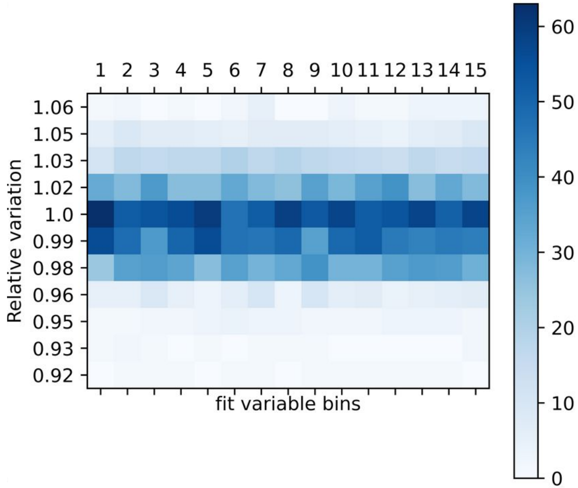
Diagnostic tools for templates

from sysvar import **plot_templates_relative_variations**
plot_relative_variations(eigendecomposer_object)



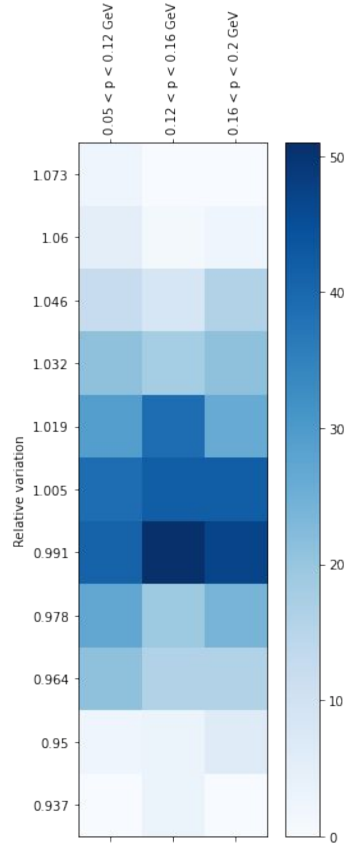
from sysvar import **plot_templates_staterror_comparison**
plot_templates_staterror_comparison(eigendecomposer_object)





from sysvar import **plot_templates_variations_distribution**
plot_templates_variations_distribution(eigendecomposer_object)

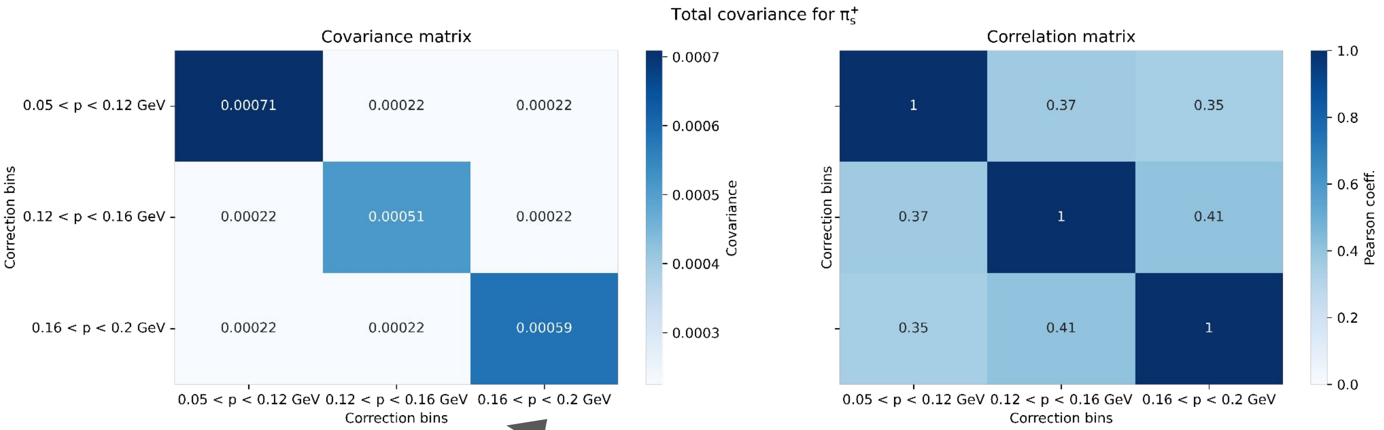
Correction variations



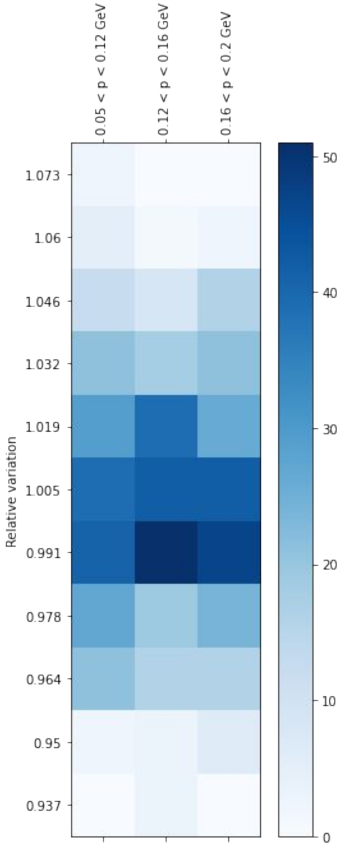
from sysvar import **plot_correction_variations_in_grid**
plot_correction_variations_in_grid(eigendecomposer_object)

Correction variations

from sysvar import **plot_corrections_cov_and_corr**
plot_corrections_cov_and_corr(eigendecomposer_object)



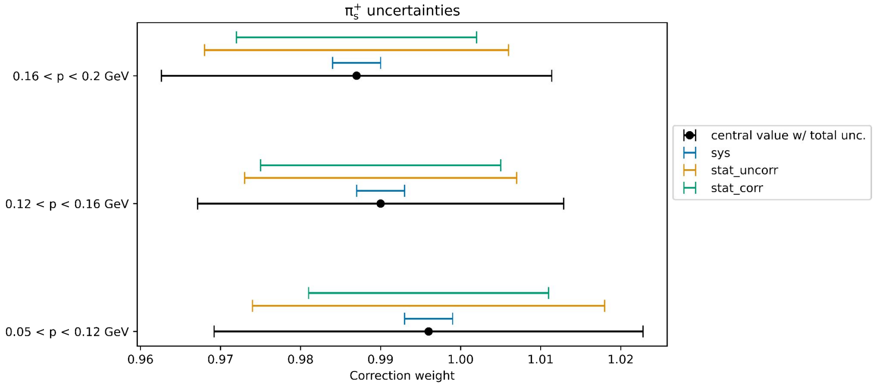
Sampled from a multidimensional gaussian

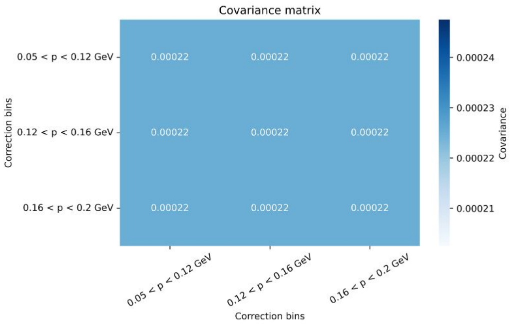


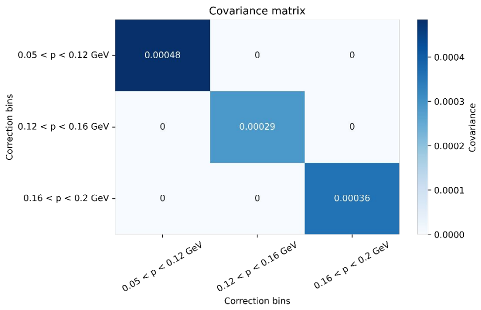
Current implementation ensures that the eigenvariations in all channels/templates are calculated from the same covariance matrix !

from sysvar import **plot_correction_variations_in_grid**
plot_correction_variations_in_grid(eigendecomposer_object)

High Level Visualization UI

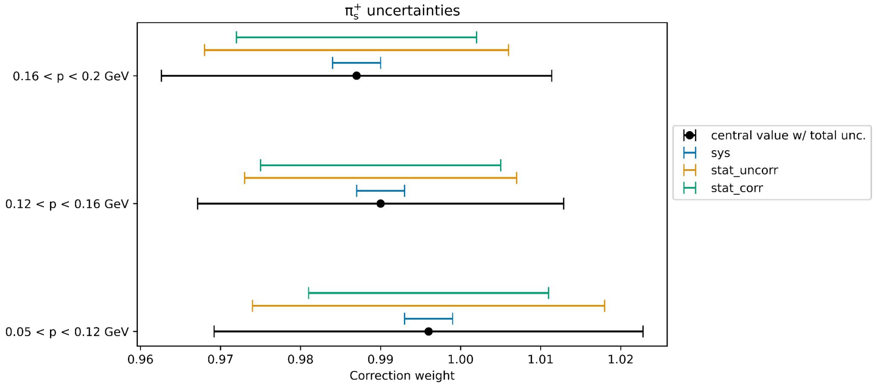


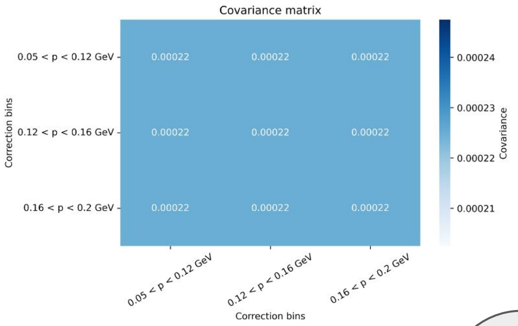


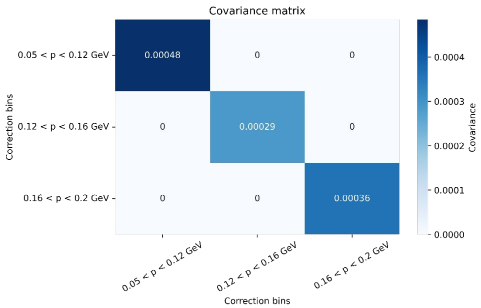


from sysvar import **plot_correction_errors**
plot_correction_errors(eigendecomposer_object)

High Level Visualization UI







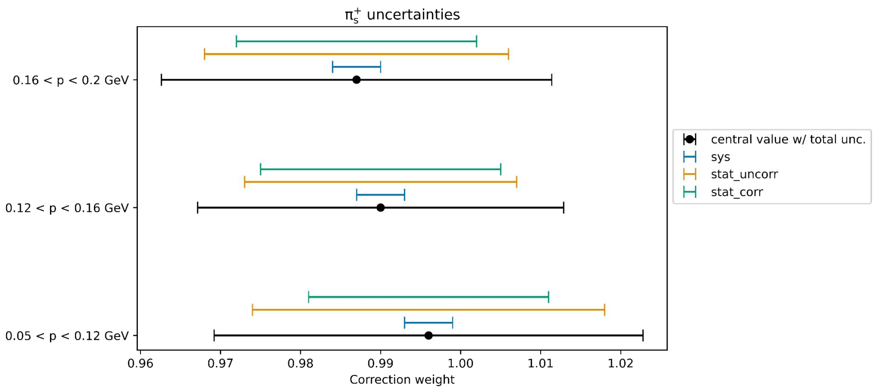
from sysvar import **plot_correction_errors**
plot_correction_errors(eigendecomposer_object)

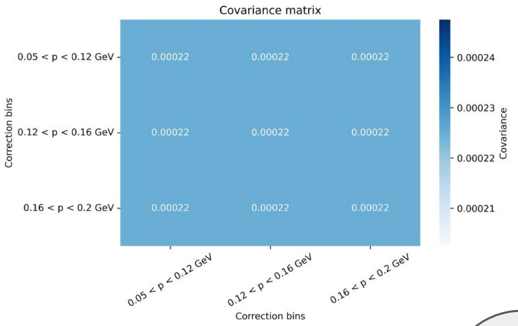
Currently supporting:

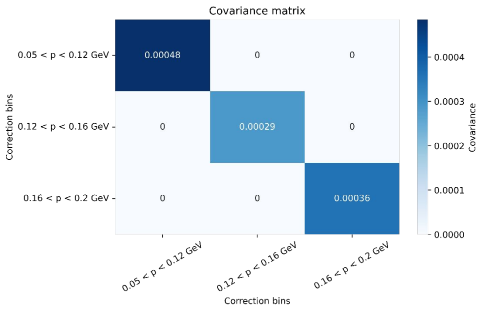
1. Tracking eff.
2. slow π eff.
3. π^0 eff.
4. e/μ eff./fakes
5. π/K eff./fakes
6. Charmed meson BF
7. τ BF
8. hadronic B decays BF

Saving varied templates using existing eigenvariations of the corrections is also possible

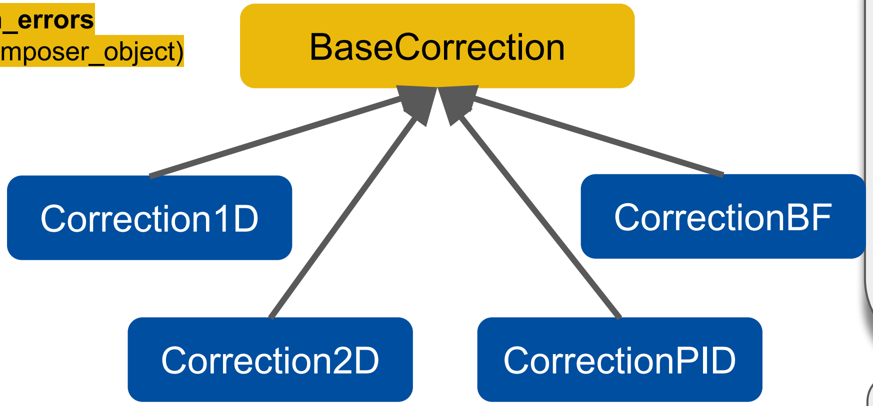
High Level Visualization UI







from sysvar import **plot_correction_errors**
plot_correction_errors(eigendecomposer_object)

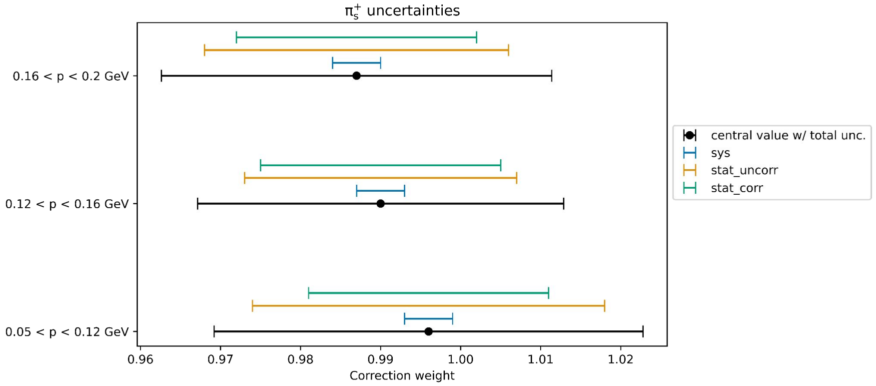


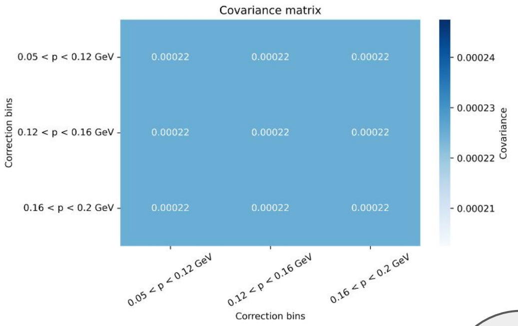
Currently supporting:

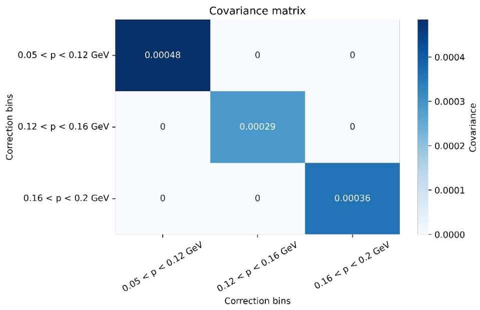
1. Tracking eff.
2. slow π eff.
3. π^0 eff.
4. e/μ eff./fakes
5. π/K eff./fakes
6. Charmed meson BF
7. τ BF
8. hadronic B decays BF

Saving varied templates using existing eigenvariations of the corrections is also possible

High Level Visualization UI



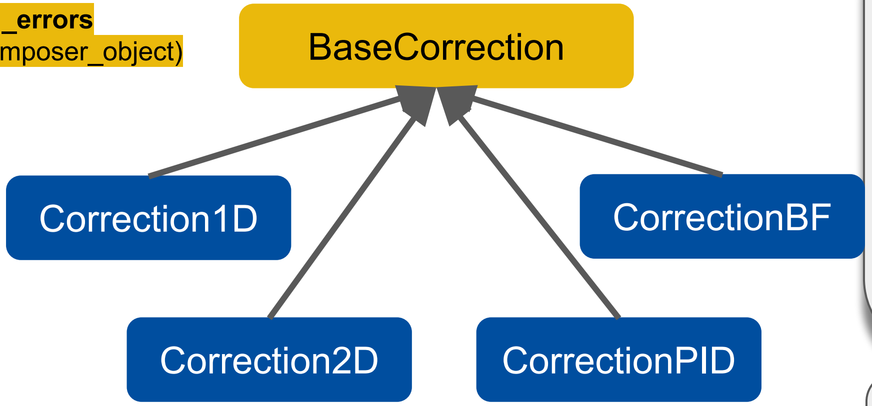




from sysvar import **plot_correction_errors**
plot_correction_errors(eigendecomposer_object)

Easily extended to other types of corrections or MC productions

Ultimate goal: Link SysVar to the central corrections repository



Currently supporting:

1. Tracking eff.
2. slow π eff.
3. π^0 eff.
4. e/μ eff./fakes
5. π/K eff./fakes
6. Charmed meson BF
7. τ BF
8. hadronic B decays BF

Saving varied templates using existing eigenvariations of the corrections is also possible

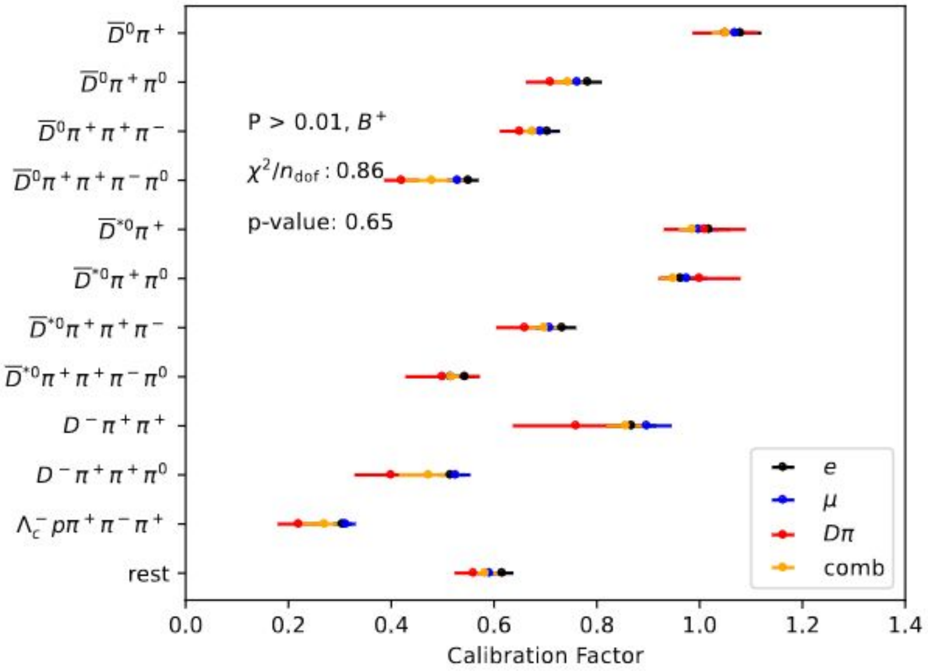
Why is this relevant ?

A usage example

Currently **combined** at a **second step with a chi2 fit** that assumes a **correlation model** for the included systematics between different reconstruction channels

SysVar provides all the necessary tools to prepare a global simultaneous fit that keeps track of all the correlations consistently!

FEI calibration factors



Simultaneous FEI calibration (Results)

- **Calibration used in the hFEI-htau R(D*) measurement**
- We fit all FEI modes for B^0 and B^+ simultaneously.
- We use two templates one for B→D*lν and one for backgrounds.
- We implement the fit using pyhf.
- All backgrounds are floating freely and independently in every mode
- Nuisance parameters are shared across all modes
- We get a p-value for goodness-of-fit of 11.06%

FEI mode	calibration factor
$\overline{D}^0\pi^+$	$1.082 \pm 0.09 \pm 0.061$
$\overline{D}^0\pi^+\pi^0$	$0.634 \pm 0.062 \pm 0.035$
$\overline{D}^0\pi^+\pi^+\pi^-$	$0.741 \pm 0.055 \pm 0.037$
$\overline{D}^0\pi^+\pi^+\pi^-\pi^0$	$0.649 \pm 0.056 \pm 0.033$
$\overline{D}^{*0}\pi^+$	$1.031 \pm 0.119 \pm 0.072$
$\overline{D}^{*0}\pi^+\pi^0$	$1.244 \pm 0.148 \pm 0.097$
$\overline{D}^{*0}\pi^+\pi^+\pi^-$	$0.758 \pm 0.092 \pm 0.048$
$\overline{D}^{*0}\pi^+\pi^+\pi^-\pi^0$	$0.599 \pm 0.098 \pm 0.044$
$D^-\pi^+\pi^+$	$0.528 \pm 0.144 \pm 0.056$
$D^-\pi^+\pi^+\pi^0$	$0.388 \pm 0.098 \pm 0.035$
$\Lambda_c^- p\pi^+\pi^-\pi^+$	$0.233 \pm 0.075 \pm 0.02$
Rest	$0.683 \pm 0.056 \pm 0.034$

FEI mode	calibration factor
$D^-\pi^+$	$1.033 \pm 0.128 \pm 0.071$
$D^-\pi^+\pi^0$	$1.023 \pm 0.101 \pm 0.059$
$D^-\pi^+\pi^+\pi^-$	$0.83 \pm 0.072 \pm 0.04$
$D^-\pi^+\pi^+\pi^-\pi^0$	$0.731 \pm 0.074 \pm 0.038$
$\overline{D}^0\pi^+\pi^-$	$0.868 \pm 0.121 \pm 0.061$
$D^{*-}\pi^+$	$1.253 \pm 0.148 \pm 0.09$
$\overline{D}^{*-}\pi^+\pi^0$	$0.493 \pm 0.055 \pm 0.024$
$\overline{D}^{*-}\pi^+\pi^+\pi^-$	$0.602 \pm 0.061 \pm 0.029$
$\overline{D}^{*-}\pi^+\pi^+\pi^-\pi^0$	$0.65 \pm 0.069 \pm 0.034$
$\Lambda_c^- p\pi^+\pi^-$	$0.693 \pm 0.168 \pm 0.07$
Rest	$1.04 \pm 0.074 \pm 0.047$

Simultaneous FEI calibration (Results)

- **Calibration used in the hFEI-htau R(D*) measurement**
- We fit all FEI modes for B^0 and B^+ simultaneously.
- We use two templates one for B→D*lν and one for backgrounds.
- We implement the fit using pyhf.
- All backgrounds are floating freely and independently in every mode
- Nuisance parameters are shared across all modes
- We get a p-value for goodness-of-fit of 11.06%

FEI mode	calibration factor
$\overline{D}^0\pi^+$	$1.082 \pm 0.09 \pm 0.061$
$\overline{D}^0\pi^+\pi^0$	$0.634 \pm 0.062 \pm 0.035$
$\overline{D}^0\pi^+\pi^+\pi^-$	$0.741 \pm 0.055 \pm 0.037$
$\overline{D}^0\pi^+\pi^+\pi^-\pi^0$	$0.649 \pm 0.056 \pm 0.033$
$\overline{D}^{*0}\pi^+$	$1.031 \pm 0.119 \pm 0.072$
$\overline{D}^{*0}\pi^+\pi^0$	$1.244 \pm 0.148 \pm 0.097$
$\overline{D}^{*0}\pi^+\pi^+\pi^-$	$0.758 \pm 0.092 \pm 0.048$
$\overline{D}^{*0}\pi^+\pi^+\pi^-\pi^0$	$0.599 \pm 0.098 \pm 0.044$
$D^-\pi^+\pi^+$	$0.528 \pm 0.144 \pm 0.056$
$D^-\pi^+\pi^+\pi^0$	$0.388 \pm 0.098 \pm 0.035$
$\Lambda_c^- p\pi^+\pi^-\pi^+$	$0.233 \pm 0.075 \pm 0.02$
Rest	$0.683 \pm 0.056 \pm 0.034$

FEI mode	calibration factor
$D^-\pi^+$	$1.033 \pm 0.128 \pm 0.071$
$D^-\pi^+\pi^0$	$1.023 \pm 0.101 \pm 0.059$
$D^-\pi^+\pi^+\pi^-$	$0.83 \pm 0.072 \pm 0.04$
$D^-\pi^+\pi^+\pi^-\pi^0$	$0.731 \pm 0.074 \pm 0.038$
$\overline{D}^0\pi^+\pi^-$	$0.868 \pm 0.121 \pm 0.061$
$D^{*-}\pi^+$	$1.253 \pm 0.148 \pm 0.09$
$\overline{D}^{*-}\pi^+\pi^0$	$0.493 \pm 0.055 \pm 0.024$
$\overline{D}^{*-}\pi^+\pi^+\pi^-$	$0.602 \pm 0.061 \pm 0.029$
$\overline{D}^{*-}\pi^+\pi^+\pi^-\pi^0$	$0.65 \pm 0.069 \pm 0.034$
$\Lambda_c^- p\pi^+\pi^-$	$0.693 \pm 0.168 \pm 0.07$
Rest	$1.04 \pm 0.074 \pm 0.047$

Systematics considered:

- tracking efficiency
- slow π efficiency
- π^0 efficiency
- D/D*/D** BF
- LID
- HID
- FF
- B→D*lν BF

Simultaneous FEI calibration (Results)

- **Calibration used in the hFEI-htau R(D*) measurement**
- We fit all FEI modes for B^0 and B^+ simultaneously.
- We use two templates one for B→D*lν and one for backgrounds.
- We implement the fit using pyhf.
- All backgrounds are floating freely and independently in every mode
- Nuisance parameters are shared across all modes
- We get a p-value for goodness-of-fit of 11.06%

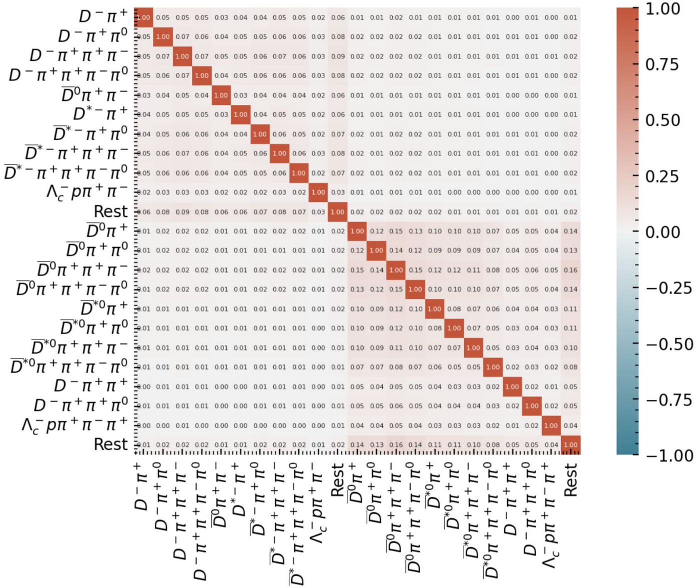
Systematics considered:

- tracking efficiency
- slow π efficiency
- π^0 efficiency
- D/D*/D** BF
- LID
- HID
- FF
- B→D*lν BF

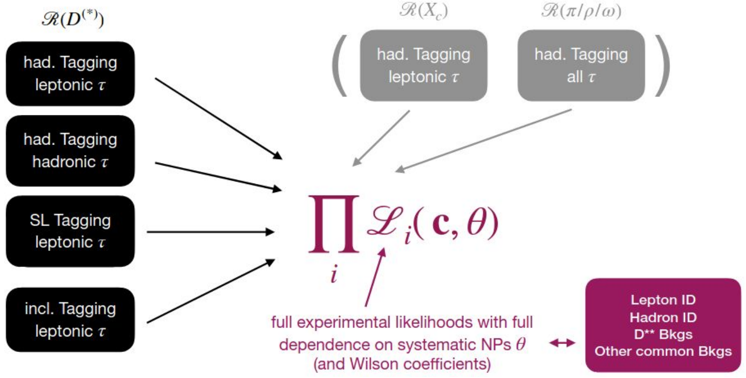
Correlation of calibration factors across FEI modes is given from the fit

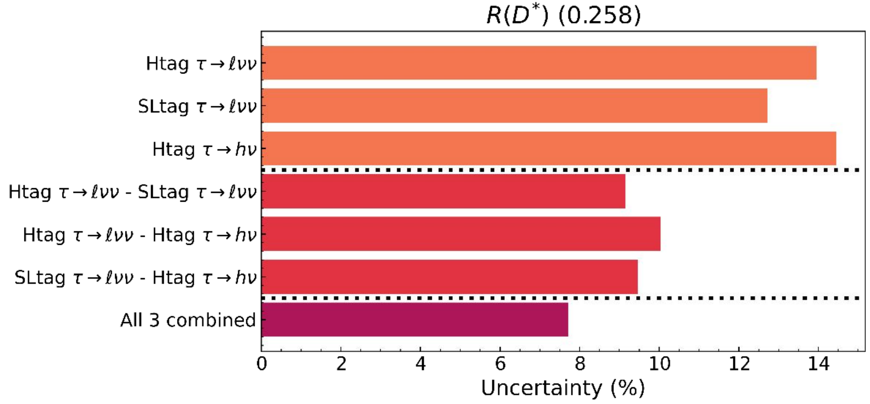
FEI mode	calibration factor
$\overline{D}^0\pi^+$	$1.082 \pm 0.09 \pm 0.061$
$\overline{D}^0\pi^+\pi^0$	$0.634 \pm 0.062 \pm 0.035$
$\overline{D}^0\pi^+\pi^+\pi^-$	$0.741 \pm 0.055 \pm 0.037$
$\overline{D}^0\pi^+\pi^+\pi^-\pi^0$	$0.649 \pm 0.056 \pm 0.033$
$\overline{D}^{*0}\pi^+$	$1.031 \pm 0.119 \pm 0.072$
$\overline{D}^{*0}\pi^+\pi^0$	$1.244 \pm 0.148 \pm 0.097$
$\overline{D}^{*0}\pi^+\pi^+\pi^-$	$0.758 \pm 0.092 \pm 0.048$
$\overline{D}^{*0}\pi^+\pi^+\pi^-\pi^0$	$0.599 \pm 0.098 \pm 0.044$
$D^-\pi^+\pi^+$	$0.528 \pm 0.144 \pm 0.056$
$D^-\pi^+\pi^+\pi^0$	$0.388 \pm 0.098 \pm 0.035$
$\Lambda_c^- p\pi^+\pi^-\pi^+$	$0.233 \pm 0.075 \pm 0.02$
Rest	$0.683 \pm 0.056 \pm 0.034$

FEI mode	calibration factor
$D^-\pi^+$	$1.033 \pm 0.128 \pm 0.071$
$D^-\pi^+\pi^0$	$1.023 \pm 0.101 \pm 0.059$
$D^-\pi^+\pi^+\pi^-$	$0.83 \pm 0.072 \pm 0.04$
$D^-\pi^+\pi^+\pi^-\pi^0$	$0.731 \pm 0.074 \pm 0.038$
$\overline{D}^0\pi^+\pi^-$	$0.868 \pm 0.121 \pm 0.061$
$D^{*-}\pi^+$	$1.253 \pm 0.148 \pm 0.09$
$\overline{D}^{*-}\pi^+\pi^0$	$0.493 \pm 0.055 \pm 0.024$
$\overline{D}^{*-}\pi^+\pi^+\pi^-$	$0.602 \pm 0.061 \pm 0.029$
$\overline{D}^{*-}\pi^+\pi^+\pi^-\pi^0$	$0.65 \pm 0.069 \pm 0.034$
$\Lambda_c^- p\pi^+\pi^-$	$0.693 \pm 0.168 \pm 0.07$
Rest	$1.04 \pm 0.074 \pm 0.047$

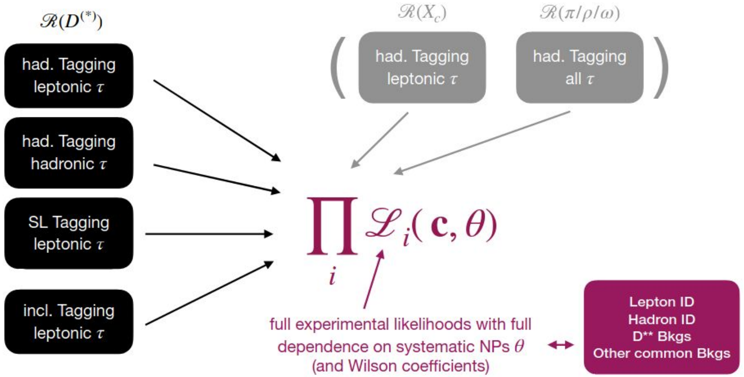


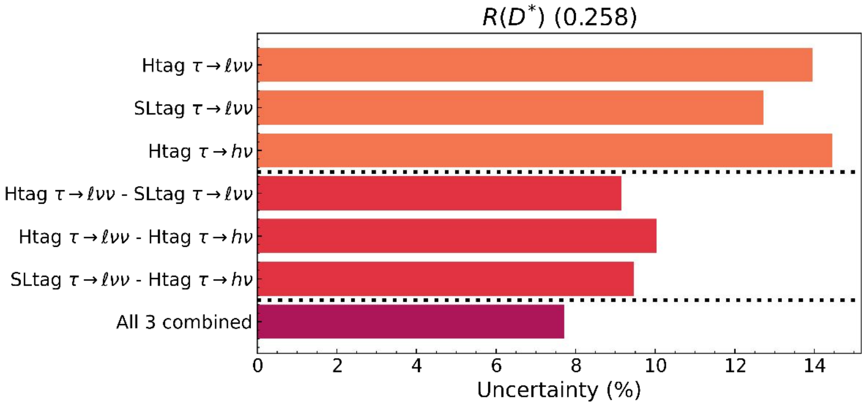
Combination of results

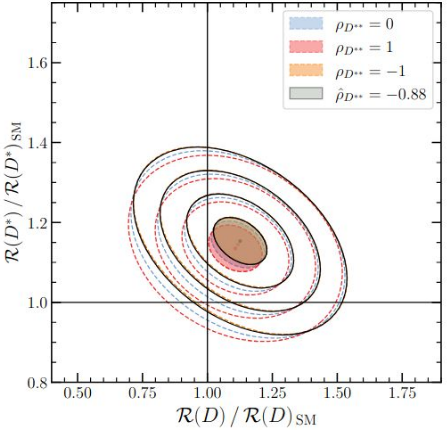


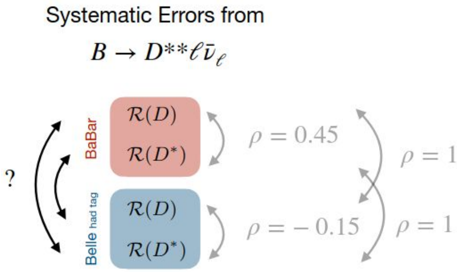


Combination of results



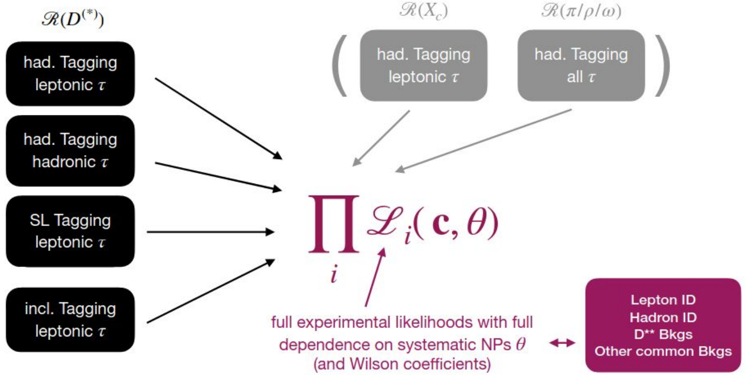


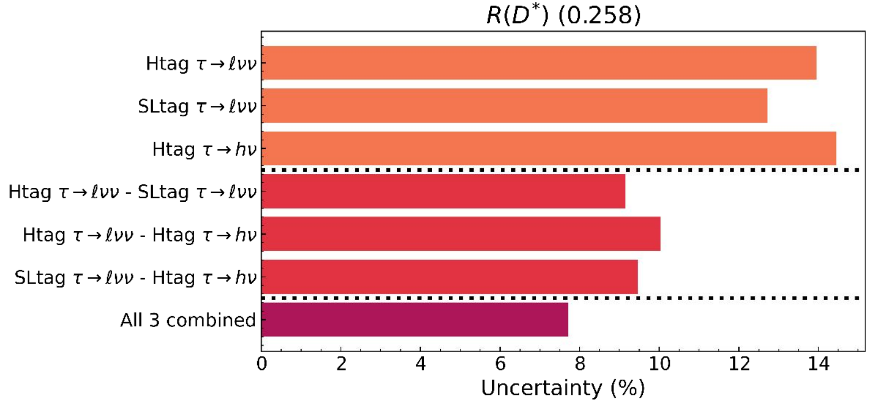


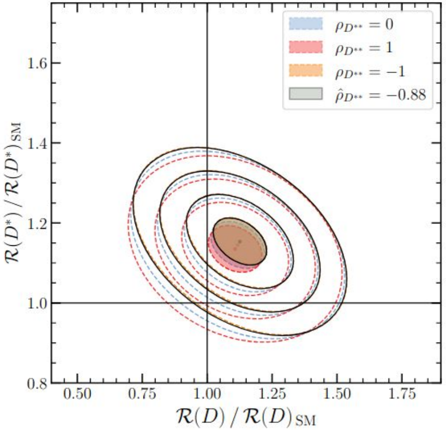


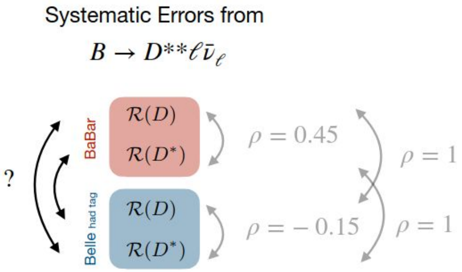
F. Bernlochner at Challenges in SL B decays, 2022

Combination of results

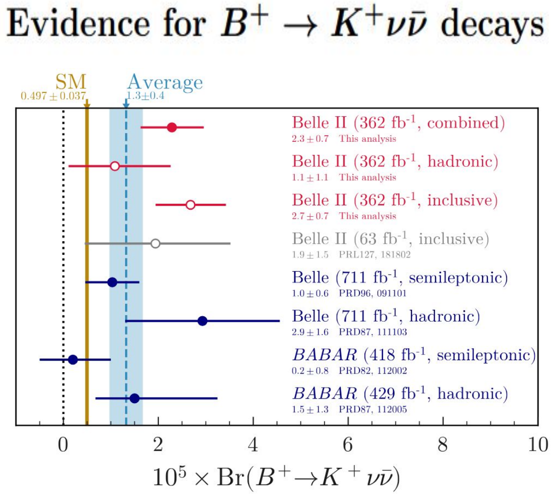




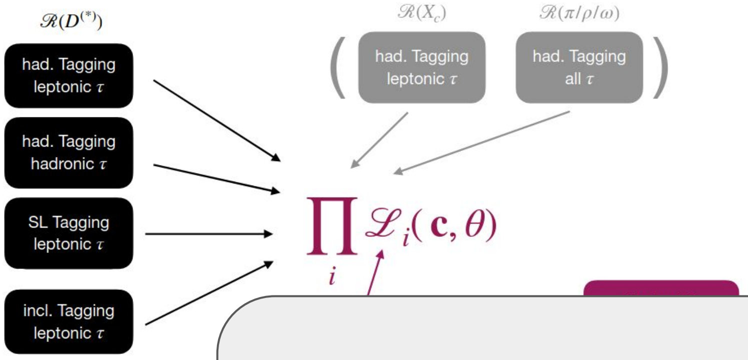


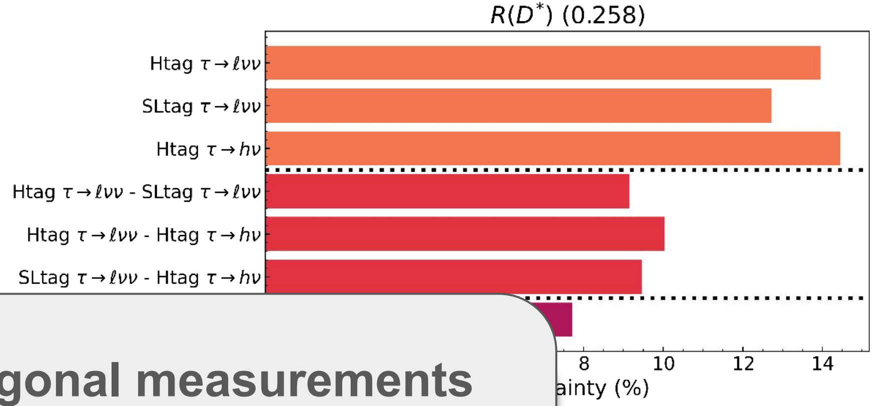


[F. Bernlochner at Challenges in SL B decays, 2022]()

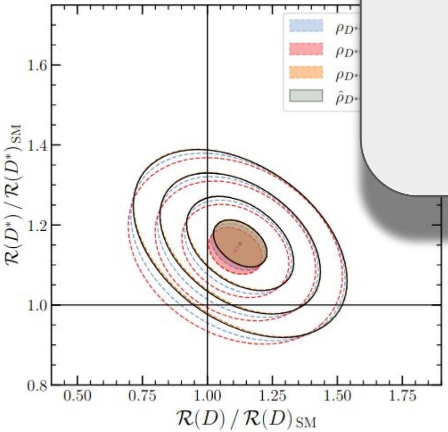


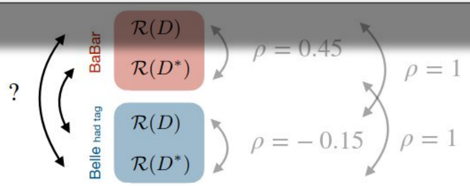
Combination of results

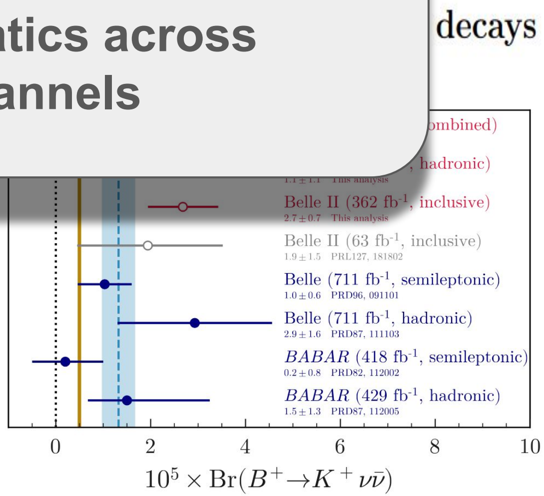




Combination of orthogonal measurements requires a framework that can treat correlations of systematics across reconstruction channels







F. Bernlochner at Challenges in SL B decays, 2022

Possible combination scheme

Belle II analysts

A few years time

Htag had τ
$R(D^{*})$

Htag lep τ
$R(D^{(*)})$

SLtag lep τ
$R(D^{(*)})$

Possible combination scheme

Belle II analysts

A few years time

Htag had τ
$R(D^*)$

Htag lep τ
$R(D^{(*)})$

SLtag lep τ
$R(D^{(*)})$

Minimal config files with essential analysis specific information

Belle II member

Current requirements:

1. The tuples are stored on the **same machine**
2. The analysts prepare a **SysVar config file**


```yaml
input_filepath: /home/test_input.root
output_filepath: /home/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
MC_prod: MC15ri
Nvar: 500

bins:
  channel1:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]
  channel2:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]

systematics:
  charged_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include: [channel1]
      exclude:

  neutral_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude: [channel1]
```

Possible combination scheme

Belle II analysts

A few years time

Htag had τ
$R(D^*)$

Htag lep τ
$R(D^{(*)})$

SLtag lep τ
$R(D^{(*)})$

Minimal config files with essential analysis specific information

Belle II member

`sysvar.combine(*cfgs)`

Eigendecomposition

Current requirements:

1. The tuples are stored on the **same machine**
2. The analysts prepare a **SysVar config file**


```
input_filepath: /home/test_input.root
output_filepath: /home/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
MC_prod: MC15ri
Nvar: 500

bins:
  channel1:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]
  channel2:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]

systematics:
  charged_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include: [channel1]
      exclude:

  neutral_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude: [channel1]
```

Possible combination scheme

Belle II analysts

A few years time

Htag had τ
$R(D^*)$

Htag lep τ
$R(D^{(*)})$

SLtag lep τ
$R(D^{(*)})$

Minimal config files with essential analysis specific information


```yaml
input_filepath: /home/test_input.root
output_filepath: /home/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
MC_prod: MC15ri
Nvar: 500

bins:
  channel1:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]
  channel2:
    fit_variable1: [0, 0.2, 0.4, 0.6, 0.8, 1]
    fit_variable2: [1, 2, 3, 4]

systematics:
  charged_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include: [channel1]
      exclude:

  neutral_slow_pi:
    weight: "weight"
    prefices: "slow_pi"
    reco_channels:
      include:
      exclude: [channel1]
```


Belle II member

`sysvar.combine(*cfgs)`

Eigendecomposition

pyhf
∂ifferentiable
$\mathscr{L}$ikelihoods

Full likelihood

$$\prod_i \mathscr{L}_i(\mathbf{c}, \theta)$$

statistically validate fit!

A few months time

Belle II combined $R(D^*)$

Current requirements:

1. The tuples are stored on the **same machine**
2. The analysts prepare a **SysVar config file**

SysVar’s status

Repository

The code for SysVar lives in a [gitlab repository]

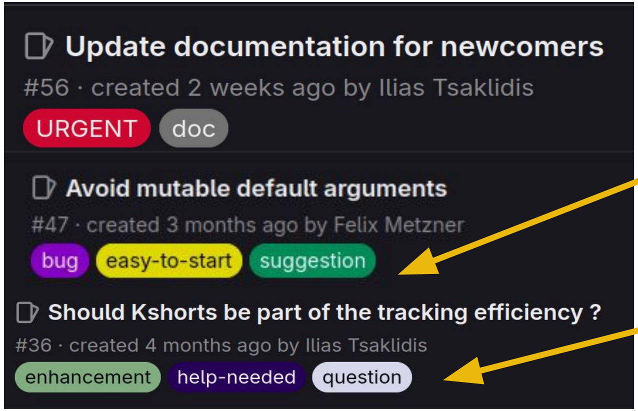
/examples/minimal_example.ipynb
was used to create a pseudodataset and
generate all the plots for today's talk

We're aiming at making SysVar pip-installable as soon as possible.

Contributors

We gratefully acknowledge the following individuals for their code contributions to this project.

Contributor	Email	Commits	Contributions
Ilias Tsaklidis	itsaklid@uni-bonn.de	267	Original idea, main developer
Agrim Aggarwal	s6agagga@uni-bonn.de	20	Co-developer, documentation, testing
Felix Metzner	felixmetzner@outlook.com	4	Feature additions, validation, feedback
Georgios Alexandris	galexand@uni-bonn.de	2	CI/CD setup and maintenance, feedback
Maximilian Hoverath	s6mahove@uni-bonn.de	2	BF correction updates from PDG



Want to get in touch ?

If there's something missing use the suggestion label and let's discuss!

For specific questions about the code/method/usage
Please open a gitlab issue adding the question label
This will help other users with similar questions in the future

Summary/Outlook

- SysVar is tool for handling systematics consistently
- A variety of systematic effects are incorporated
- Correlating multiple reconstruction channels (even from different analyses) is possible as a single unified workflow.
- The design of the software allows for smooth maintenance and extension of its features

Summary/Outlook

- SysVar is tool for handling systematics consistently
- A variety of systematic effects are incorporated
- Correlating multiple reconstruction channels (even from different analyses) is possible as a single unified workflow.
- The design of the software allows for smooth maintenance and extension of its features

Important future milestones:

1. Documentation
2. unit tests
3. Link SysVar to the envisioned central corrections' repository
4. Consolidation of combination API

Summary/Outlook

- SysVar is tool for handling systematics consistently
- A variety of systematic effects are incorporated
- Correlating multiple reconstruction channels (even from different analyses) is possible as a single unified workflow.
- The design of the software allows for smooth maintenance and extension of its features

Thank you for the attention!
Hope that in a few months many analyses can profit from SysVar improving Belle II results!

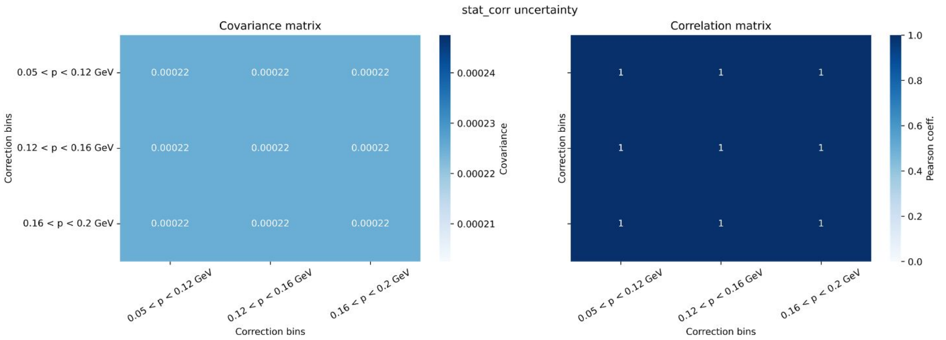
Important future milestones:

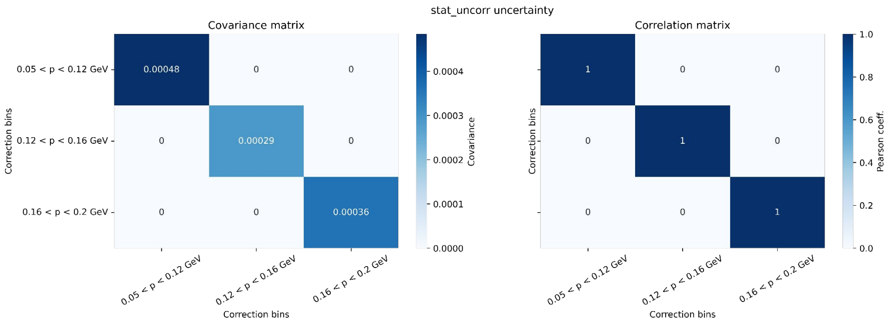
1. Documentation
2. unit tests
3. Link SysVar to the envisioned central corrections' repository
4. Consolidation of combination API

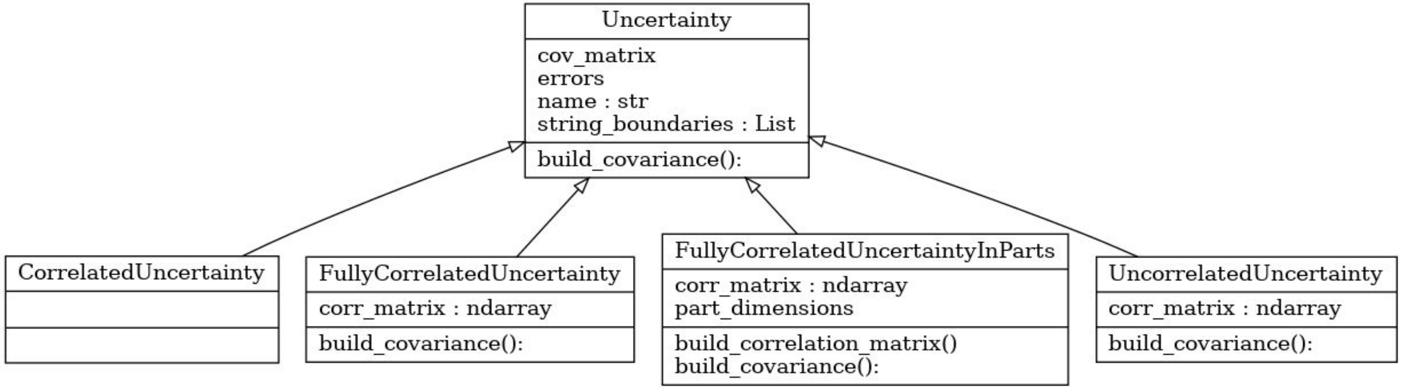
BACK-UP

Uncertainties

All uncertainties are implemented in separate Uncertainty objects

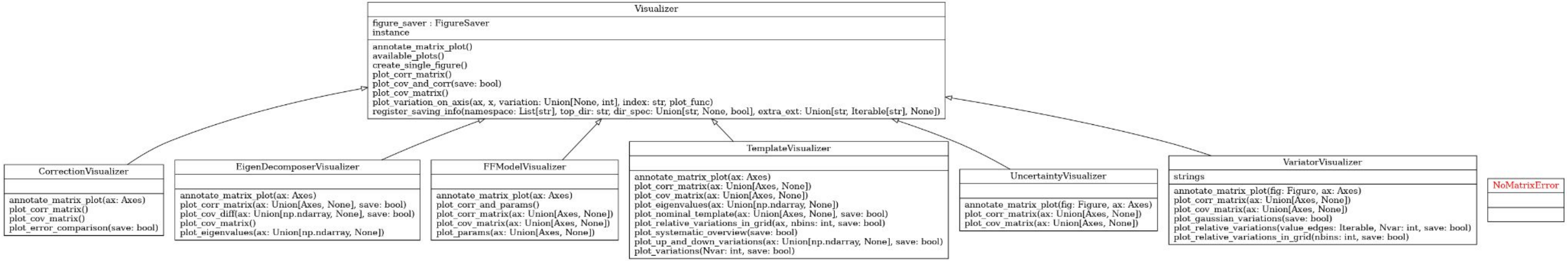






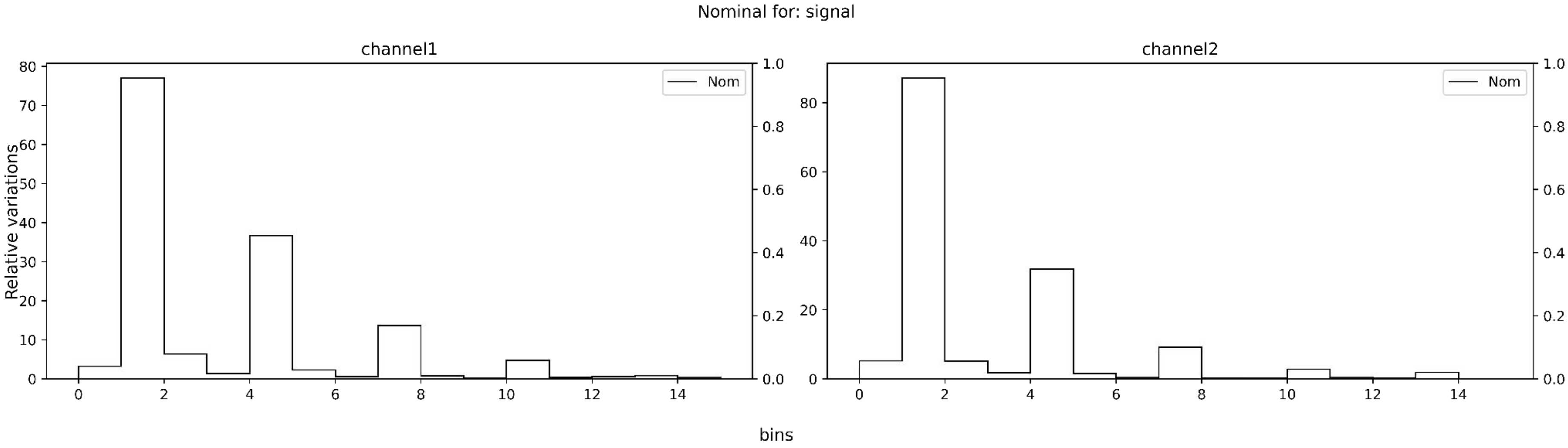
Visualizer

Every object in SysVar comes with its respective *Visualizer*.
The user never interacts with this object.
All of its methods are overridden and called by the main objects.
This serves as an extra layer of modularization

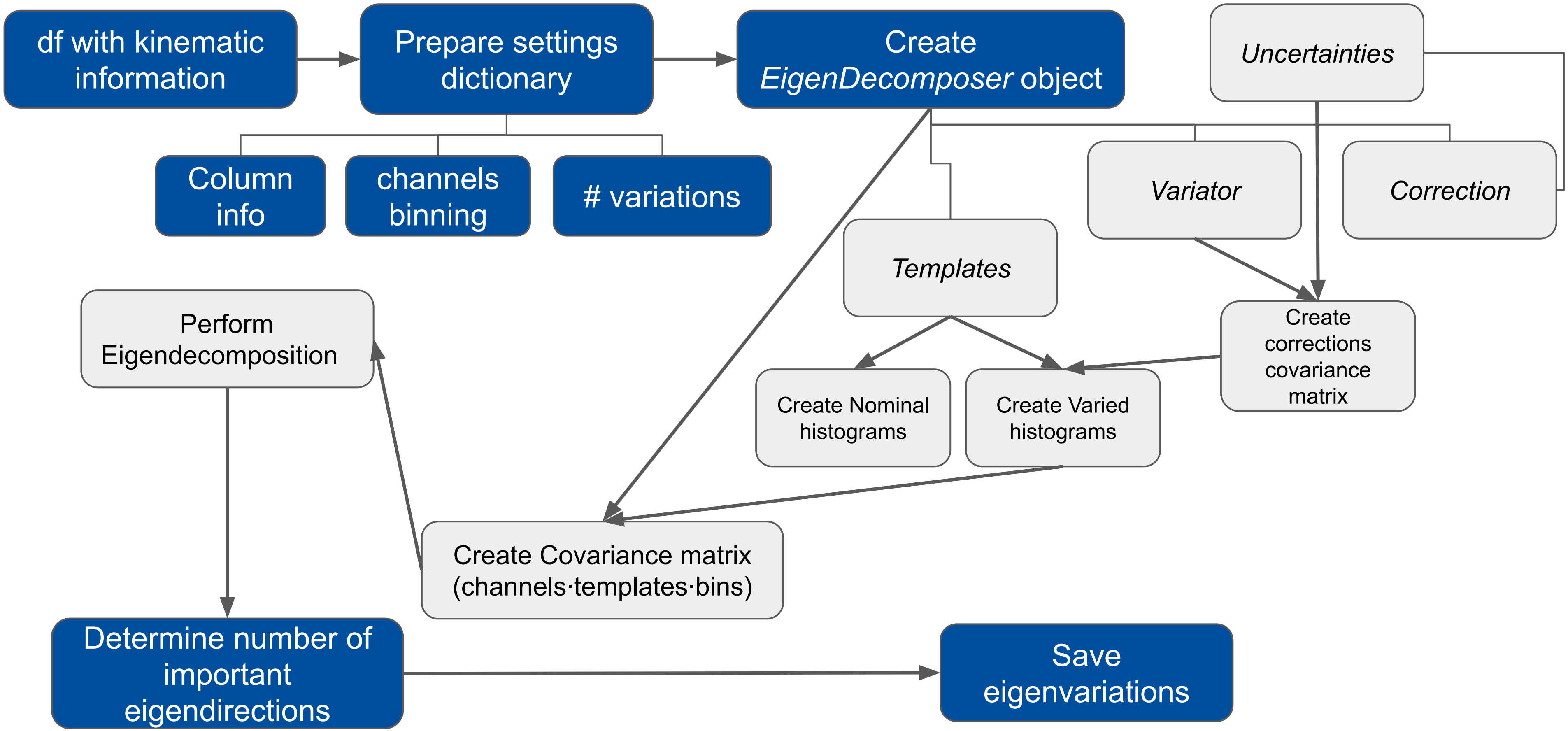


Similarly for saving the objects' output
i.e. figures, covariance matrices etc
A *Saver* object is deployed behind the scenes

Nominal template



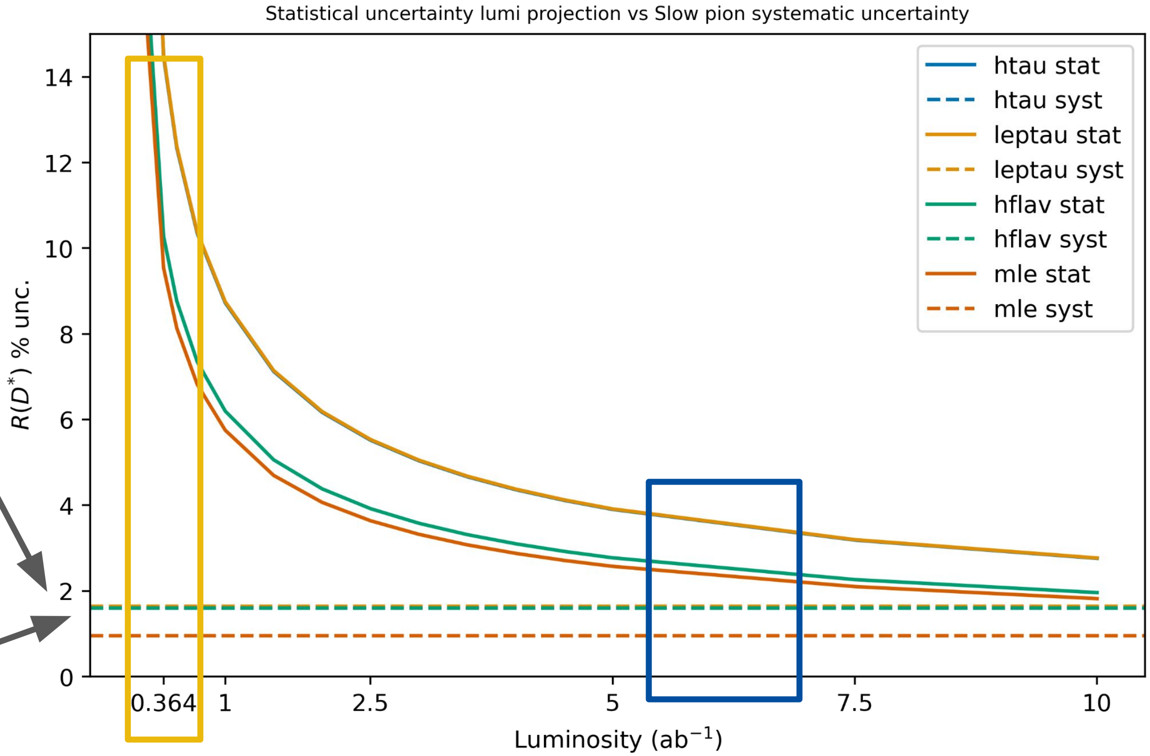
Quick flow chart of producing EigenVariations



Comparison of combination methods

<table>
<tr><th></th><th>R(D*)</th><th>$\sigma^{R(D^*)}_{stat}$ %</th><th>$\sigma^{R(D^*)}_{syst\,\text{slow pion}}$ %</th><th>R(D)</th><th>$\sigma^{R(D)}_{stat}$ %</th><th>$\sigma^{R(D)}_{syst\,\text{slow pion}}$ %</th></tr>
<tr><td>Htag, $\tau \to \ell\nu\nu$</td><td rowspan="6">0.258</td><td>14.50</td><td>1.64</td><td rowspan="6">0.299</td><td>27.61</td><td>2.10</td></tr>
<tr><td>Htag, $\tau \to h\nu$</td><td>14.46</td><td>1.60</td><td>-</td><td>-</td></tr>
<tr><td>HFLAV style comb $\rho = 0$</td><td rowspan="2">10.27</td><td>1.15</td><td>-</td><td>-</td></tr>
<tr><td>HFLAV style comb $\rho = 1$</td><td>1.60</td><td>-</td><td>-</td></tr>
<tr><td>MLE comb $\rho = 0$</td><td rowspan="2">9.53</td><td>0.88</td><td rowspan="2">21.98</td><td>1.03</td></tr>
<tr><td>MLE comb $\rho = $ true</td><td>0.95</td><td>1.21</td></tr>
</table>

Statistical vs Systematic uncertainty



$$\sigma_{\mathrm{R(D^*)}} = \sigma^{\mathrm{LS1}}_{\mathrm{R(D^*)}} \cdot \sqrt{\frac{\mathcal{L}_{\mathrm{LS1}}}{\mathcal{L}}}$$

not the total systematic error

not the leading systematic uncertainty

Now we're here

In a couple of years we'll be there

Need to add a few of those in quadrature

Can we handle all the data ?

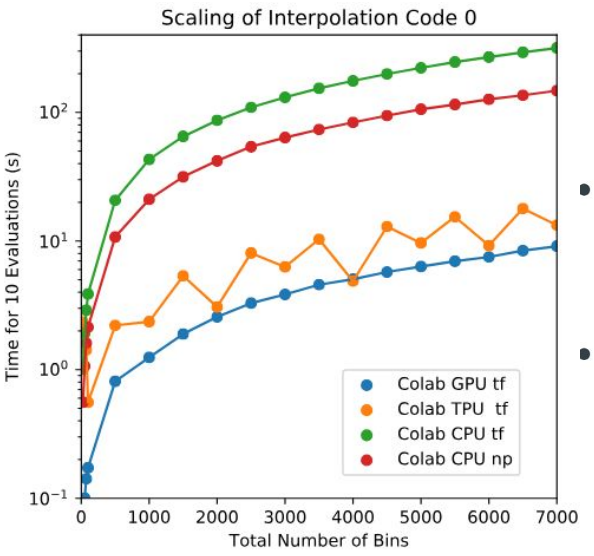
Eigendecomposition on the likelihood level should not be preferred

SysVar's Combination API: A Powerful Tool for Streamlined Combined EigenDecomposition

1. **Selective column collection**
   - Automatically retrieve only essential columns, as defined in the cfg file, minimizing memory consumption. Currently supporting a growing list of file formats.
2. **Input merging and unification**
   - Combine and standardize input data structures to ensure consistency across analyses.
3. **Automatic cfg generation**
   - Build a new cfg file for all analyses, simplifying configuration management.
4. **Flexible multi-channel processing**
   - Handle multiple reconstruction channels (even from different analyses) as a single unified workflow.

Can we fit an increasing number of nuisance parameters ?

pyhf is using **deep learning frameworks** as **computational backends** which allows for exploitation of auto differentiation (**autograd**) and **GPU acceleration**



- Show hardware acceleration giving **order of magnitude speedup** for some models!
- Improvements over traditional
  - 10 hrs to 30 min; 20 min to 10 sec

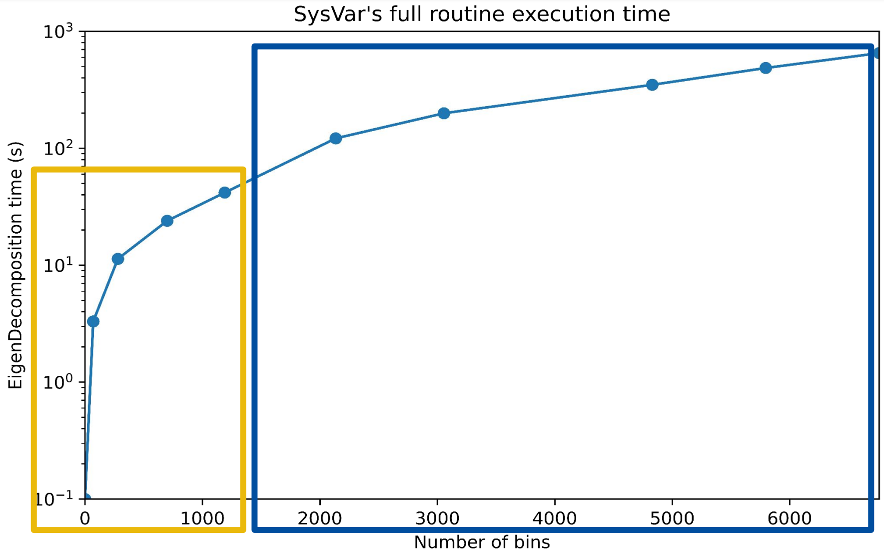
L. Heinrich at ACAT 2019
M. Feickert at Scipy 2020

SysVar timing benchmarks

How computationally intensive is the EigenDecomposition ?

Execution time includes:

1. Building templates
2. Calculating variations
3. Performing Eigendecomposition
4. Determining important eigendirections



Depends on the criterion used.
There's room for improvement in the implementation
but still the execution time doesn't make the task impractical

SysVar's future

Agrim Aggarwal : s6agagga@uni-bonn.de is already providing valuable support in the development

His PhD starts in September

We've already discussed with Frank and we have a document outlining Agrim's service task Sep-Feb

Info soon to be registered on b2mms

June - July 2024

Meet targets **1-2**.

September - November 2024 **Extending features**

Meet targets **3-8**.

Ilias can continue providing support and guidance on these tasks. We suggest that the development remains in the GitLab repository of SysVar until target 8 is achieved. Ilias and Agrim have a well-defined workflow with tracked GitLab issues that Agrim can continue working on. Keeping the development in the current repository will ensure that no momentum is lost, and we meet the prerequisite targets before merging into basf2 as quickly as possible.

December - January 2025 **basf2 integration**

Meet targets **9-10**

Guidance from a member of the software group will be necessary. Ilias will be close to finishing his PhD and will not be able to provide a lot of support on the technical side of integrating SysVar into basf2. The presence of a person that Agrim can ask technical questions about basf2 development will ensure smooth integration into basf2 in a timely manner.

February - March 2025 **Usage recommendations**

Meet targets **11-12** and **finalize the support note** Once the basf2 integration is complete Markus can supervise these studies which will ensure that SysVar's official recommendations on the treatment of systematic uncertainties have solid statistical foundations.

We need **a person from software** to **provide support** to Agrim in the second phase of his service task (**basf2 integration**)

Should we present this to a **wider audience** e.g. Physics meeting and **collect feedback** for **desired features** in other WGs for phase 1 ?