Clustering of B Decay Kinematic Distributions

Kilian Lieret[1,3], Jason Aebischer[2, 3], Thomas Kuhr[1,3]

[1]Ludwig-Maximilian University
[2]Technical University of Munich
[3]Excellence Cluster Origins

March 15, 2019

Motivation

Phenomenology of NP models depends on free parameters (e.g. points in the space of Wilson coefficients) influencing the shape of kinematic distributions

Problems:

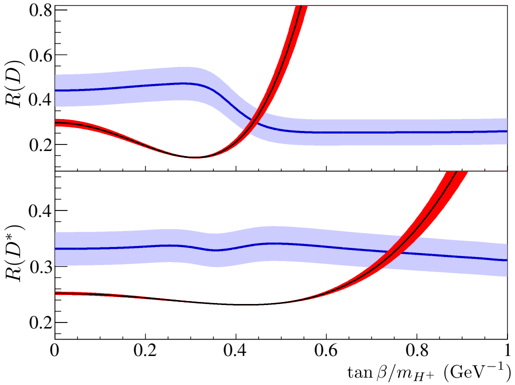
- Experimental analyses need to make assumptions on kinematic distributions to extract features of interest
  $\Longrightarrow$ Need to re-run analysis for different NP models and their parameters
  $\Longrightarrow$ Often only results under assumption of SM
- Difficult to present numeric results (e.g. exclusion limits)
  (there are no nice ways to visualize 3+ dimensions)

Motivation II

Example

BaBar's result for $R(D^{(*)})$ based on model assumption

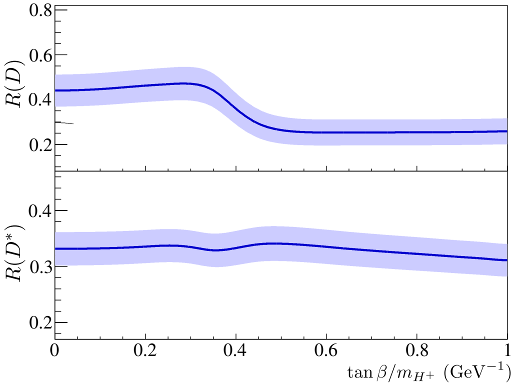
(2HDM with specific $\tan\beta$ parameter)



Motivation II

Example

BaBar's result for $R(D^{(*)})$ based on model assumption
(2HDM with specific $\tan\beta$ parameter)



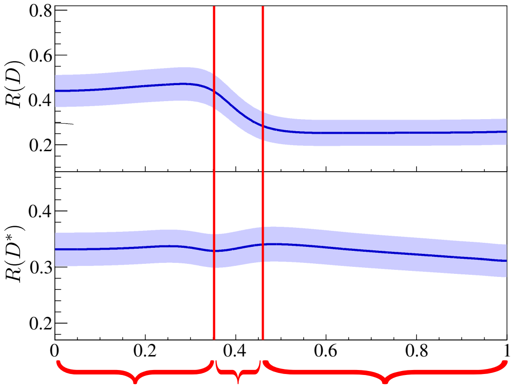
Results obtained from a 2D fit to m^2_{miss} and $|\vec{p}_\ell|$

Motivation II

Example

BaBar's result for $R(D^{(*)})$ based on model assumption

(2HDM with specific $\tan\beta$ parameter)



In the three parts, the m^2_{miss}–$|\vec{p}_\ell|$ behavior of the model has probably been different $\Longrightarrow$ Would have been nice to categorize the parameter space by this behavior right away

Motivation III

Clustering of distributions in the parameter space boils down multi-dimensional problems to few benchmark points!

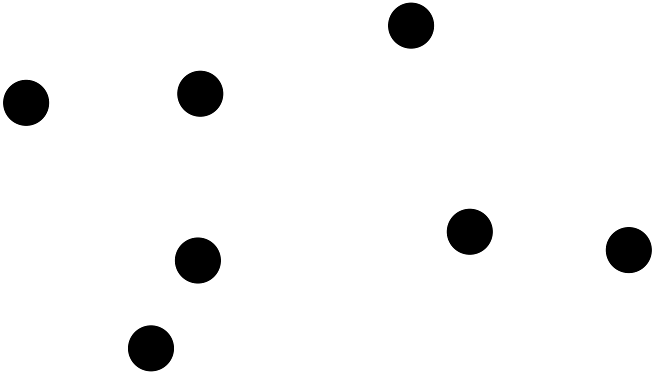
Algorithm

Quantify "similar" distributions:
$\longrightarrow$ similarity test, e.g. χ^2 test or Kolmogorov test

Build up groups (*clusters*) of similar distributions:
Need a clustering algorithm, e.g. hierarchical clustering

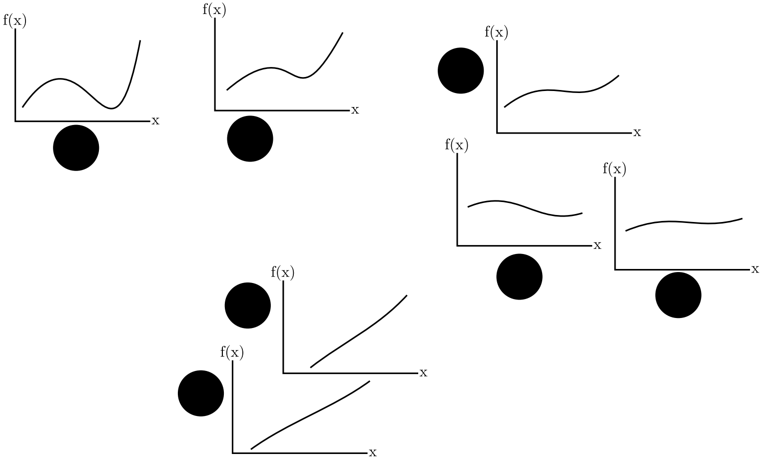
Need to know **how fine** (or coarse) our clustering should be:
$\longrightarrow$ Add experimental error expectation
$\longrightarrow$ Keep as many clusters as we can distinguish

Example: Hierarchical Clustering



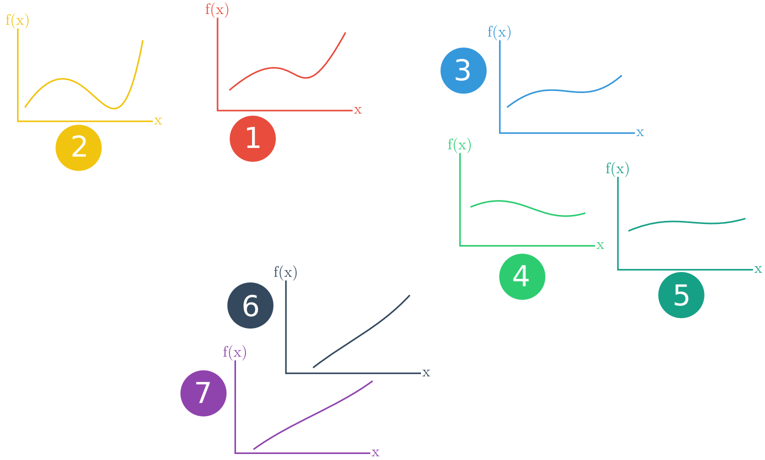
Step 0: 7 points in the parameter space

Example: Hierarchical Clustering



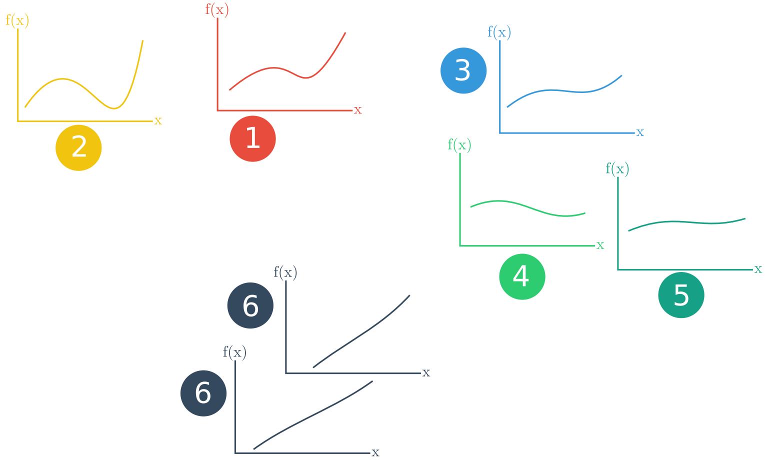
Step 0: 7 points in the parameter space = 7 distributions

Example: Hierarchical Clustering



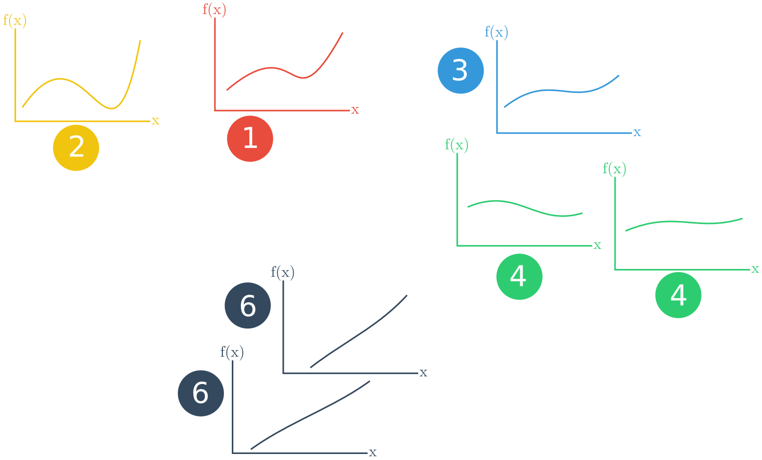
Step 1: Every point is its own cluster

Example: Hierarchical Clustering



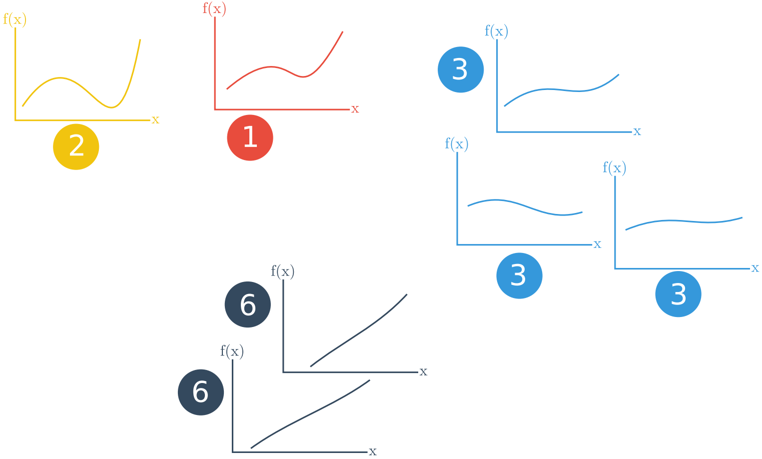
Step 1: Distributions from cluster 6 and 7 were the most similar $\Longrightarrow$ Merged!

Example: Hierarchical Clustering



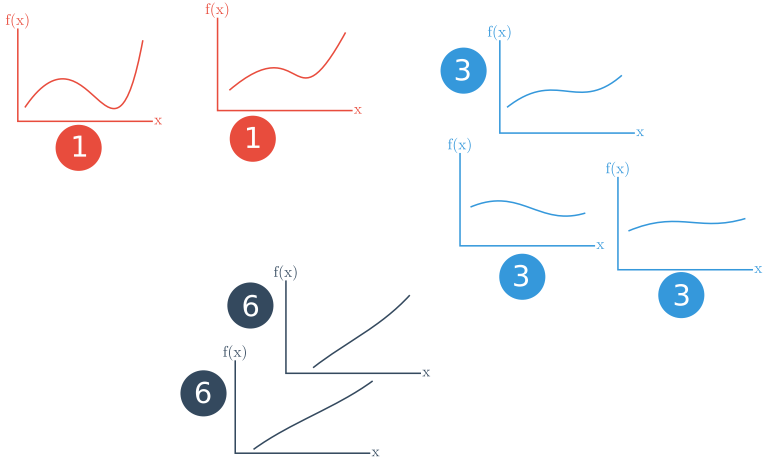
Step 1: Cluster 4 and 5 were the next most similar clusters $\Longrightarrow$ Merged!

Example: Hierarchical Clustering



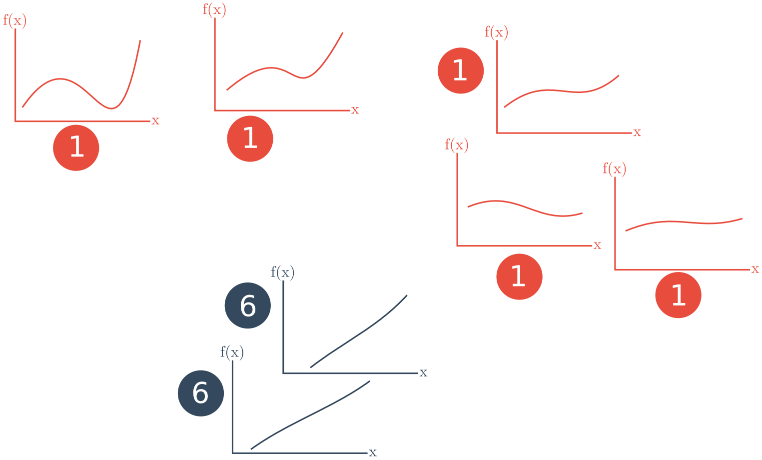
Step 1: 4 clusters remaining

Example: Hierarchical Clustering



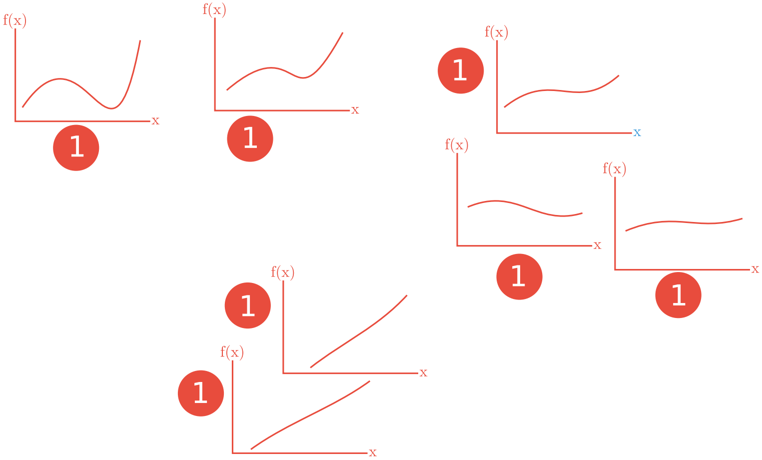
Step 1: 3 clusters remaining

Example: Hierarchical Clustering



Step 1: 2 clusters remaining

Example: Hierarchical Clustering



Step 1: 1 cluster remaining

Software

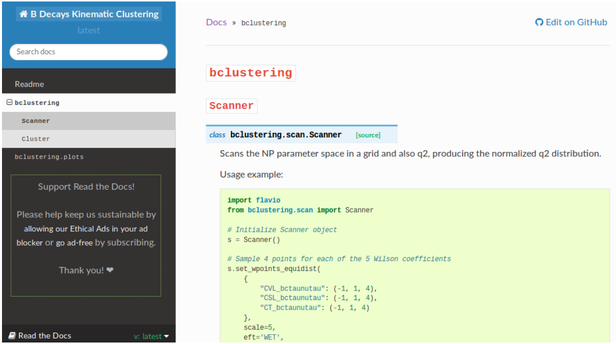
Our project is **openly** and **actively** developed on GitHub
https://github.com/RD-clustering/B_decays_clustering

Implemented in python™ using

- `numpy` (fast numeric operations on arrays)
- `pandas` (dataframes)
- `matplotlib` (beautiful plots)
- `scikit-learn` (clustering tools)
- `scipy` (integration and clustering tools)
- `jupyter` (interactive notebooks)
- `wcxf` (specify Wilson coefficients in a variety of bases)
- `Wilson` (running of Wilson coefficients)
- `Flavio` (various observables with NP predictions)

Software

- Up to date **documentation** on `Read the Docs`:



- **Interactive tutorials** using jupyter `jupyter notebooks`

Software

General steps:

1. **Scan**: Calculate binned distributions for your observable, e.g. $\mathrm{d}\Gamma / \mathrm{d}q^2$
   - An arbitrary python function can be specified
   - E.g. simply take an observable from `flavio`
   - Parallel processing supported
2. (optional) **Add errors**: Easy interface to add various kinds of errors (Poisson, flat relative errors, errors given by covariance matrix, maximally correlated errors etc.)
3. **Cluster**: Take the binned distributions and cluster them
   Clustering class is subclassed to support any clustering algorithm
4. **Benchmark point**: Select one representative for each cluster
5. **Plot**: Various plotting methods are provided

Software

Quick tutorial

Generate a sample of kinematic distributions (here $\mathrm{d}\Gamma / \mathrm{d}q^2$ for $B \longrightarrow D^* \tau \bar{\nu}_\tau$) using the `Scanner` class:

```
s = Scanner()
s.set_dfunction(
    bdlnu.dGq2,
    binning=np.linspace(bdlnu.q2min, bdlnu.q2max, 10),
    normalize=True
)
s.set_wpoints_equidist(
    {
        "CVL_bctaunutau": (-0.3, 0.3, 10),
        "CSL_bctaunutau": (-0.3, 0.3, 10),
        "CT_bctaunutau": (-0.4, 0.4, 10)
    },
    scale=5,
    eft='WET',
    basis='flavio'
)
s.run()
s.write("output/scan", "tutorial")
```

Software

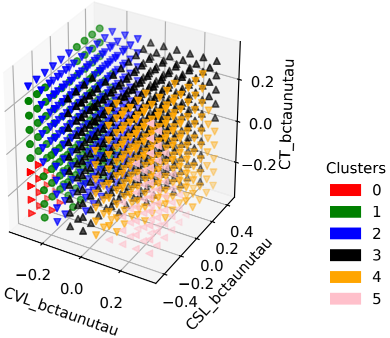
Quick tutorial

Now we cluster it:

```
c = HierarchyCluster("output/scan", "tutorial")
c.build_hierarchy()
c.cluster(max_d=0.1)
```

And can directly plot it:

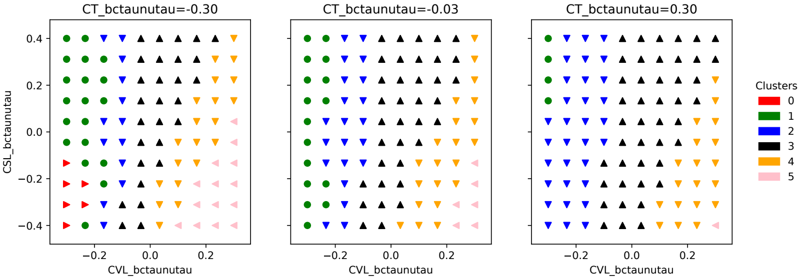
```
cp = ClusterPlot(df)
cp.scatter([
    'CVL_bctaunutau',
    'CSL_bctaunutau',
    'CT_bctaunutau'
])
```



Software

Quick tutorial

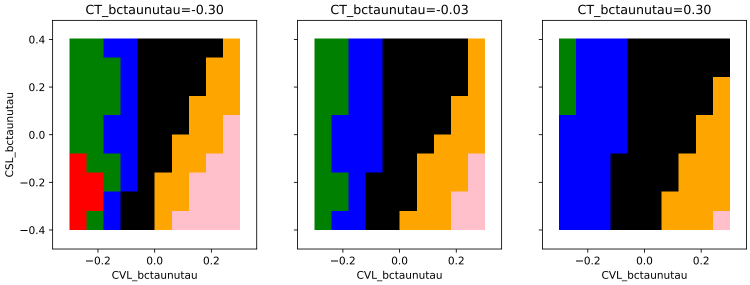
```
p.scatter(['CVL_bctaunutau', 'CSL_bctaunutau'])
```



Software

Quick tutorial

```
p.fill(['CVL_bctaunutau', 'CSL_bctaunutau'])
```



Summary

Clustering of kinematic distributions in the Wilson parmameter space boils down multi-dimensional problems to few benchmark points!

Openly and **actively** developed project on

GitHub

https://github.com/RD-clustering/B_decays_clustering

Feedback and suggestions, as well as helping hands are very welcome!