Corrections, s-Weights & systematics

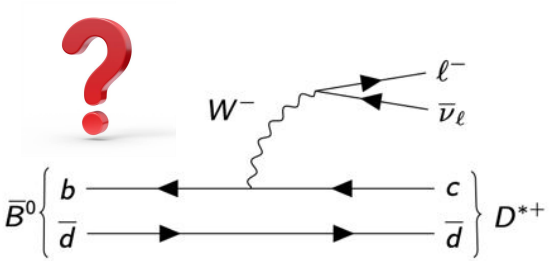
Common analysis techniques - 10/11/2023

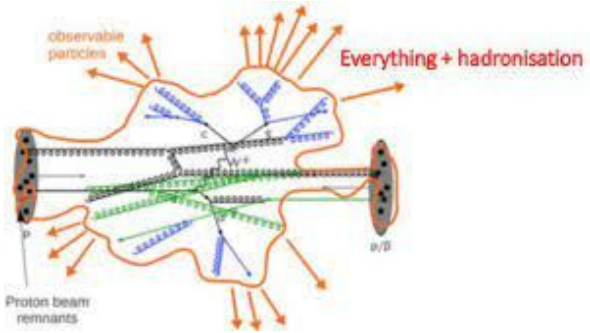
Comparing data to MC

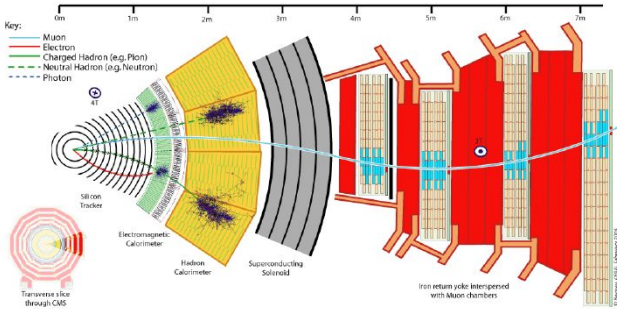
- We use Monte Carlo (MC) simulation to understand our physical process and separate signal from background
- But simulation is not perfect! Some reasons why?

Comparing data to MC

- We use Monte Carlo (MC) simulation to understand our physical process and separate signal from background
- But simulation is not perfect! Some reasons why (not exhaustive):
  - Cross sections and branching ratios of different processes have yet to be measured
  - Hard process simulation is not good enough (e.g. it is only calculated to some finite precision)
  - Detector simulation is not yet optimal (material budget, dead channels, ...)
  - Physics “behaves“ differently than initially expected in a new energy range







What can we do about it?

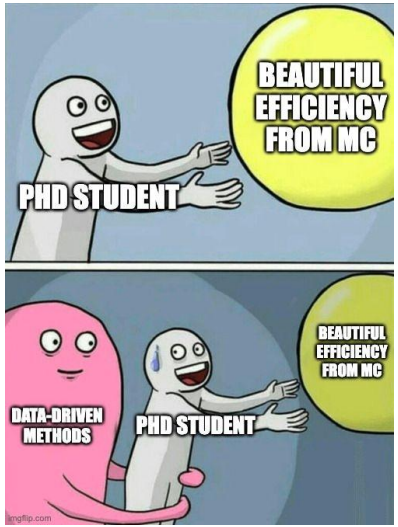
What can we do about it?

- Ideally: Improve simulation!
  - Better description of our detector
  - More precise theoretical predictions of the hard scattering process
  - Input from new measurements
  - …

What can we do about it?

- Ideally: Improve simulation!
  - Better description of our detector
  - More precise theoretical predictions of the hard scattering process
  - Input from new measurements
  - …

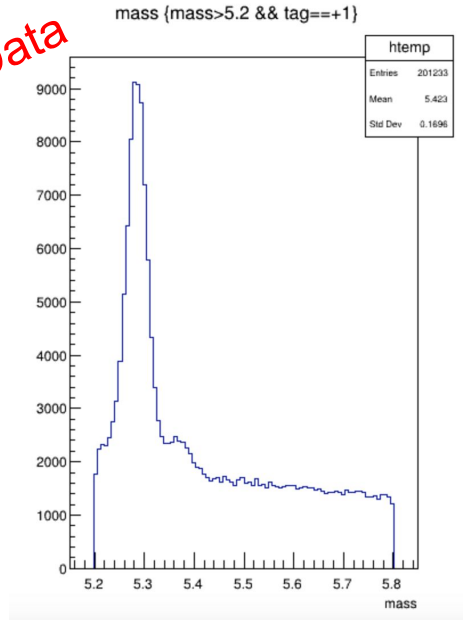
- But we don’t live in an ideal world. In the meantime:
  - Use our data in smart ways to correct the simulation!



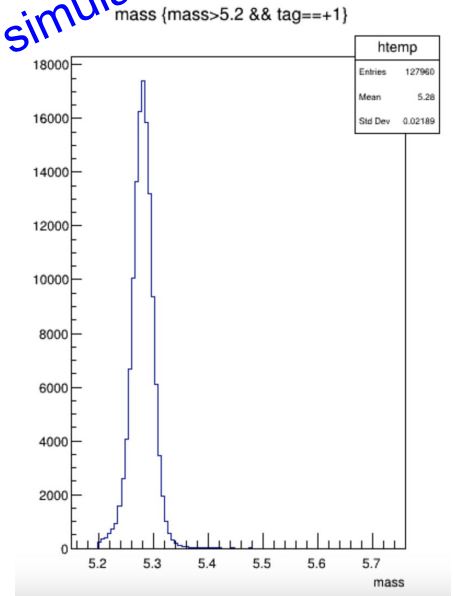
Correcting simulation using data

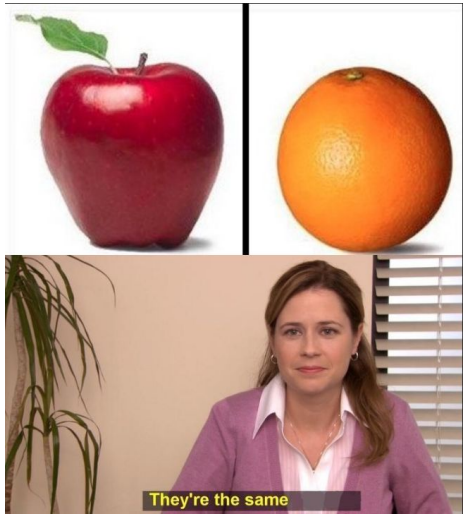
- We will usually want to have our signal simulation as representative of the data as possible
- But what is the main issue if we try to compare them?

Data



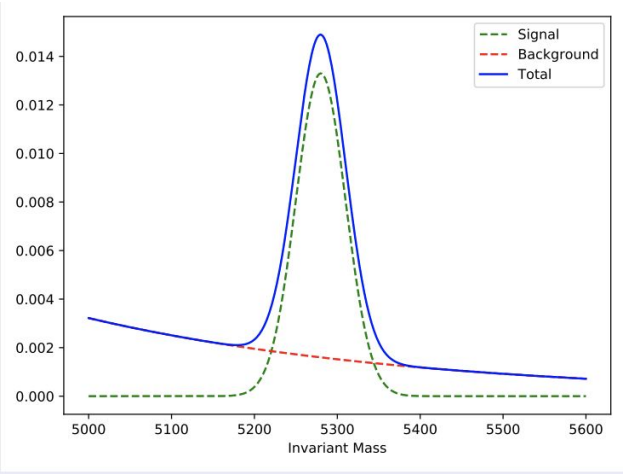
Signal simulation

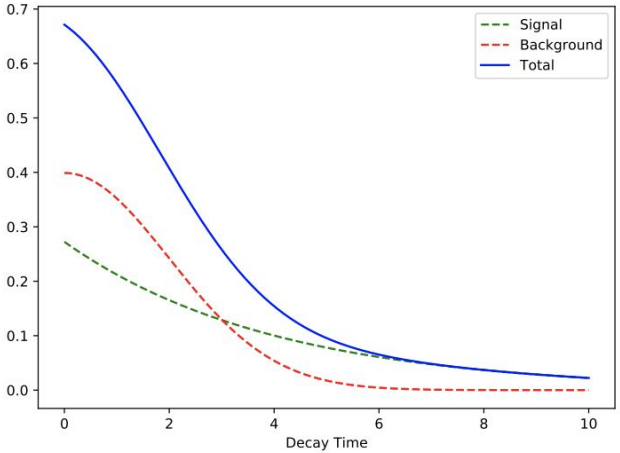




Correcting simulation using data

- We will usually want to have our signal simulation as representative of the data as possible
- But is the main issue if we try to compare them?
- Data is (almost always) an admixture of our signal process + a bunch of other stuff (backgrounds)
- **We want to “subtract” the background component(s) from our data**





S-weights: why?

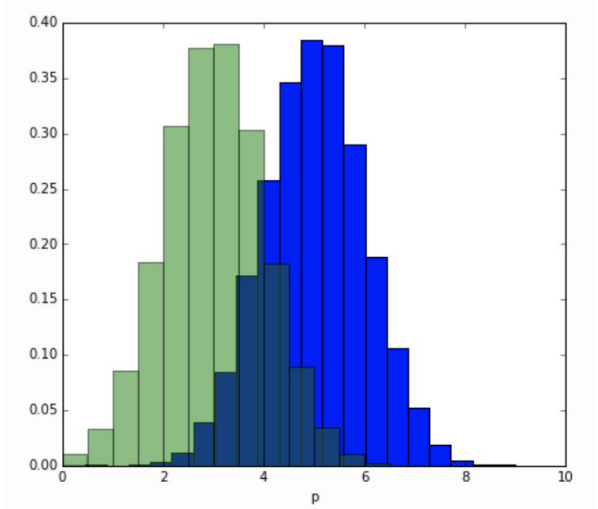
- In the previous lectures, we fitted the invariant mass distributions of the B systems to extract the CP violation
- But what if we want to extract information on other dimensions? For instance
  - Lifetime, angular distributions
  - Signal dependency on kinematics of the decay, typically needed for efficiency corrections
- One way to do this would be to have multi-dimensional fits
  - Drawback: we need to have a suitable model of all our background components in all dimensions of interest
- Alternative if modeling of background in n-dimensions is hard/undesirable: s-weights!

Material from: https://arxiv.org/pdf/physics/0402083.pdf http://arogozhnikov.github.io/2015/10/07/splot.html and https://hsf-training.github.io/analysis-essentials/advanced-python/60sPlot.html

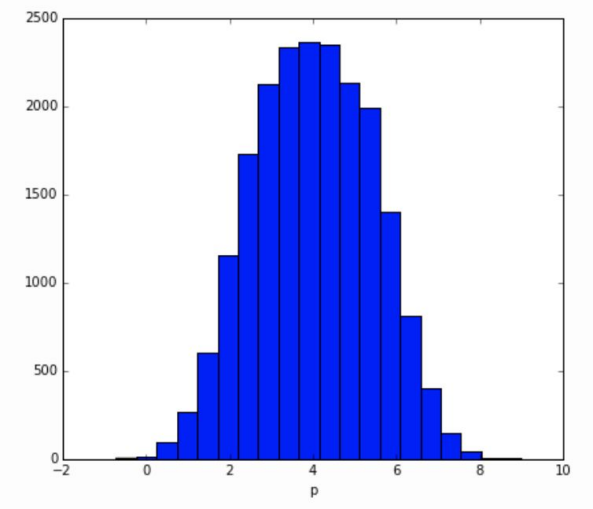
S-weights: how?

- We have a signal and background component in our discriminant (m) and control (kinematic variables such as p, decay time etc, denoted t) variables

What we would like to know



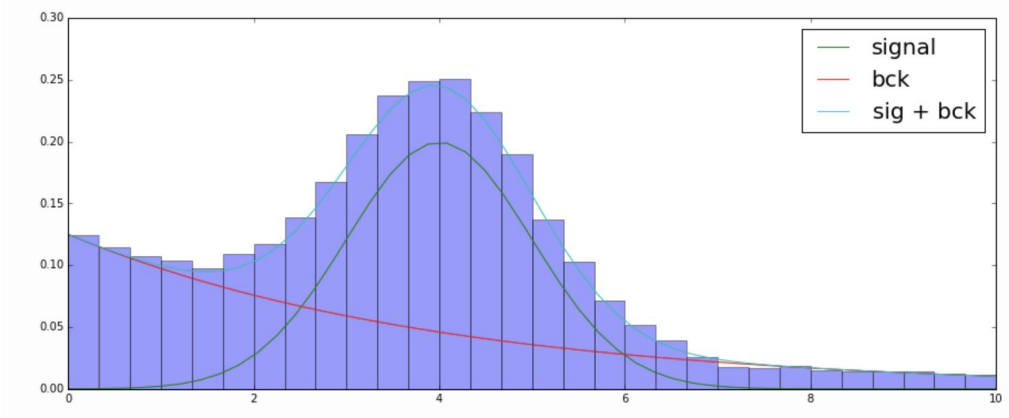
What we have



- **Main requirement:** our signal and background components need to be factorizable in both discriminant and control variables

S-weights: how?

- Let’s look at our discriminant variable (example mass):



- Fitting gives us a set of probabilities for each event to be signal or background
- sWeights: conversion to a probabilities of signal / background in each bin of the control variable (in this example p)

S-weights: some maths

- $p_s^{m(p)}$ and $p_b^{m(p)}$: unknown probabilities of signal/background in the discriminant (control) variable in given bin
- **Main requirement:** our signal and background components need to be factorizable in both discriminant and control variables -> p_s(b) don't depend on discriminant variable!
- Number of signal events in bin i of control variable : X = p_s^p Ns

$$p_s^p N_s = \Sigma_x w_s (p_s^p p_s^m + p_b^p p_b^m)$$

Since the previous equation should hold for all possible p_s and p_b, we get two equalities:

$$p_s N_s = \sum_x sw_s(x)\ p_s p_s(x)$$
$$0 = \sum_x sw_s(x)\ p_b p_b(x)$$

After reduction:

$$N_s = \sum_x sw_s(x)\ p_s(x)$$
$$0 = \sum_x sw_s(x)\ p_b(x)$$

Under assumption of linearity:

assuming that sPlot weight can be computed as a linear combination of conditional probabilities:

$$sw_s(x) = a_1 p_b(x) + a_2 p_s(x)$$

We can easily reconstruct those numbers, first let's rewrite our system:

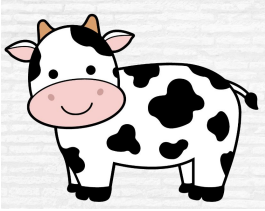
$$\sum_x (a_1 p_b(x) + a_2 p_s(x))\ p_s(x) = 0$$
$$\sum_x (a_1 p_b(x) + a_2 p_s(x))\ p_b(x) = N_{sig}$$

$$a_1 V_{bb} + a_2 V_{bs} = 0$$
$$a_1 V_{sb} + a_2 V_{ss} = N_{sig}$$

Where $V_{ss} = \sum_x p_s(x)\ p_s(x)$, $V_{bs} = V_{sb} = \sum_x p_s(x)\ p_b(x)$, $V_{bb} = \sum_x p_b(x)\ p_b(x)$

Having solved this linear equation, we get needed coefficients

Custom Orthogonal Weight functions (COWs)



Main requirement for sWeights: our signal and background components need to be factorizable in both discriminant and control variables

What if that's not the case? For example if we have a non-factorizable efficiency function e(m,t)

Generalizing sWeights: $w_k(m) = \frac{\sum_{k=0}^{n} a_k g_k(m)}{g(m)} \Rightarrow w_k(m) = \frac{\sum_{l=0}^{n} A_{kl} g_l(m)}{I(m)}$

- Where I(m) an arbitrary non-zero function (variance function) and $A_{kl}^{-1} = W_{kl} = \int \frac{g_k(m) g_l(m)}{I(m)} dm$

Then the weights to project out hk(t) are:

$$w_s = \sum_{k=0}^{j-1} \frac{w_k(m)}{\epsilon(m,t)}$$ Signal weight function

and

$$w_b = \sum_{k=j}^{n} \frac{w_k(m)}{\epsilon(m,t)}$$ Background weight function

No fitting needed, just:

1. A signal density with large, preferably maximal, overlap with the true signal density
2. A background density modelled by a truncated sum of polynomials
3. A variance function obtained directly from the data

More info in: paper, slides

Reweighting Simulation

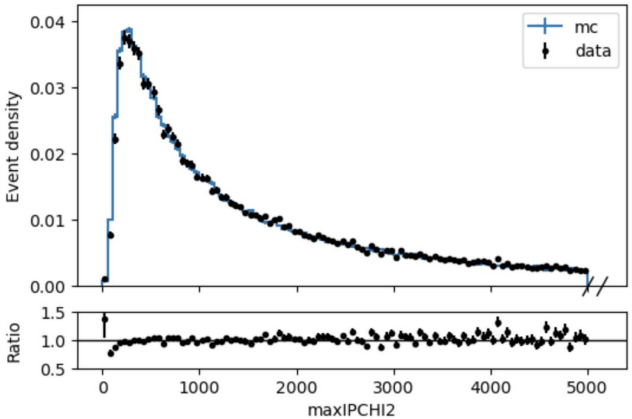
- We saw how to separate our signal from our background in our dataset…

- And now we can compare our signal simulation to our “background-free” sWeighted data!

Reweighting Simulation

- We saw how to separate our signal from our background in our dataset…

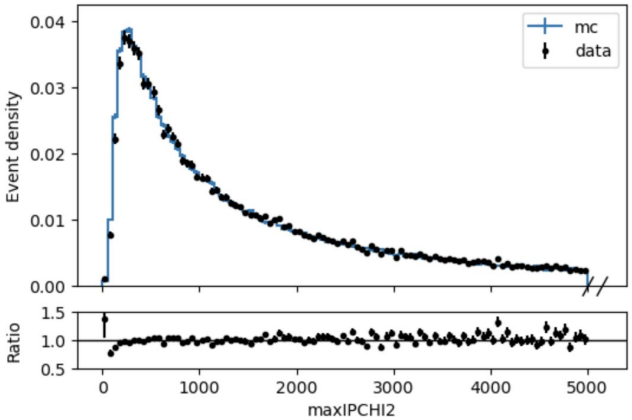
- And now we can compare our signal simulation to our “background-free” sWeighted data!

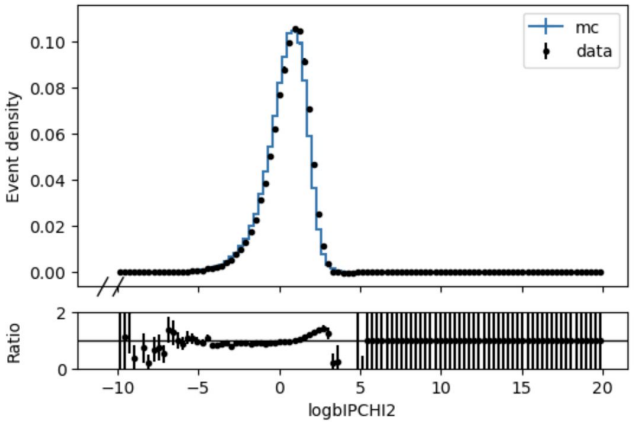


Reweighting Simulation

- We saw how to separate our signal from our background in our dataset…

- And now we can compare our signal simulation to our “background-free” sWeighted data!

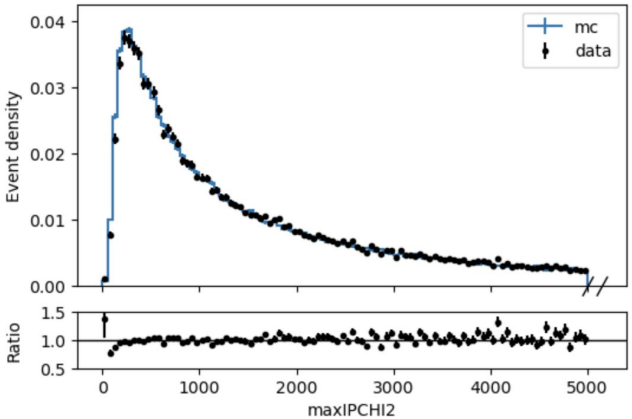


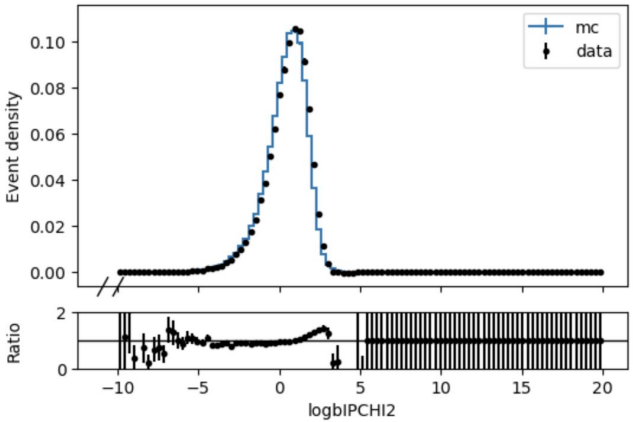


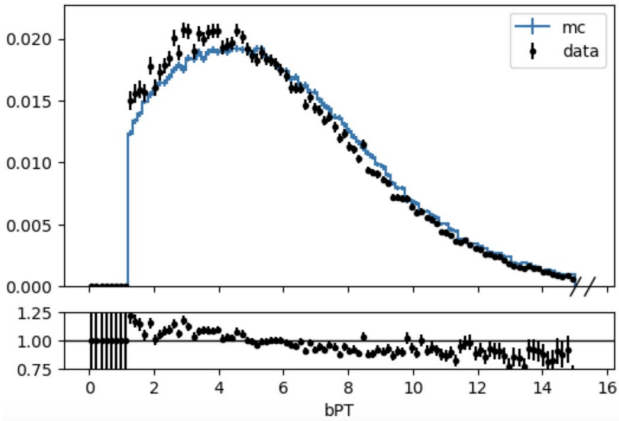
Reweighting Simulation

- We saw how to separate our signal from our background in our dataset…

- And now we can compare our signal simulation to our “background-free” sWeighted data!







Reweighting Simulation

- We can re-weight our simulation to look more like our data
- **ATTENTION:** Reweighting should be handled with care, you want to make sure you don't introduce biases to your measurement - typically extract weights from similar but orthogonal sample to your signal region

Most straightforward reweighting method: bin-based reweighting

- In each bin of your i variable, you multiply the original distribution by: $m_{bin} = w_{target} / w_{original}$
- w_{target} and $w_{original}$ are the total weights in each bin for the target (data) and original (simulation) distribution
- **Simple and fast!**
- But quickly breaks down when you need to reweight more than 1-2 variables
- Also, reweighting one variable can make disagreement in another variable worse
- Choice of which variables to use is not always obvious

Reweighting Simulation

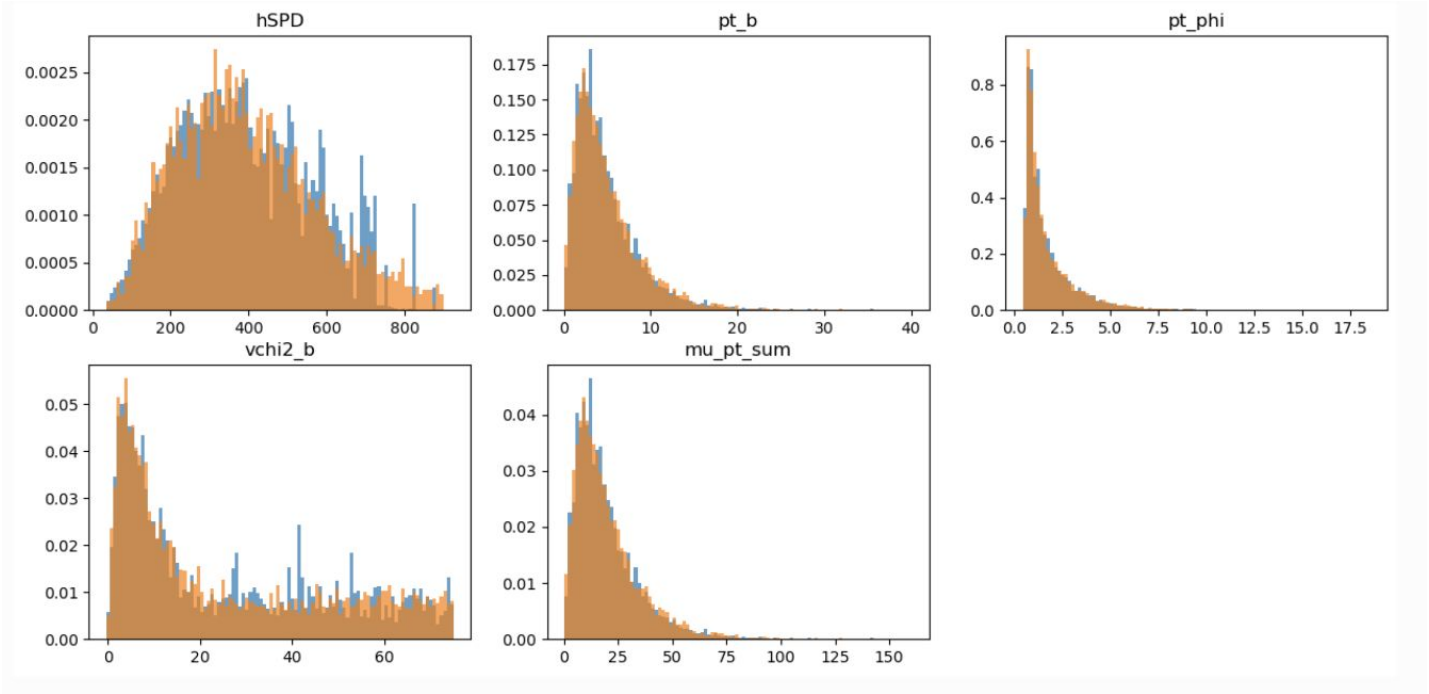
- We can re-weight our simulation to look more like our data
- **ATTENTION:** Reweighting should be handled with care, you want to make sure you don't introduce biases to your measurement - typically extract weights from similar but orthogonal sample to your signal region

Alternative: use MVA-based reweighting

- You train a classifier to discriminate between data and MC -> you get probability weights that a given event belongs to data or MC -> the ratio of these probabilities is an approximation of the data/MC density ratio per bin (i.e. our weight)
- Works well when you need to simultaneously correct in more than 1-2 variables
- Many libraries out there (BDT, GBRweighter, ANN)
- Usually doesn't work very well when density ratio is high
- Negative weights not always treated properly

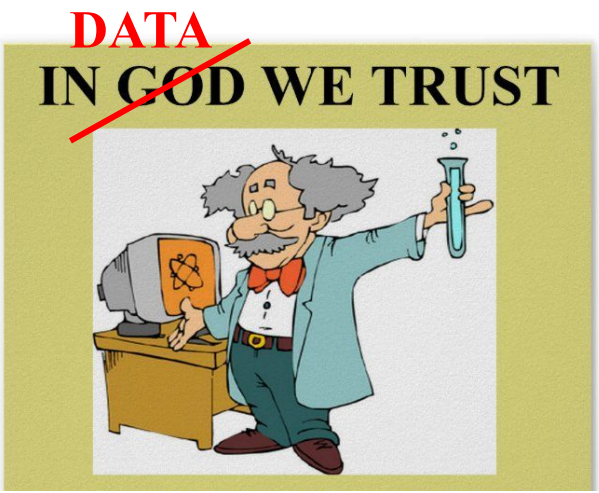
Reweighting Simulation

If everything works well…



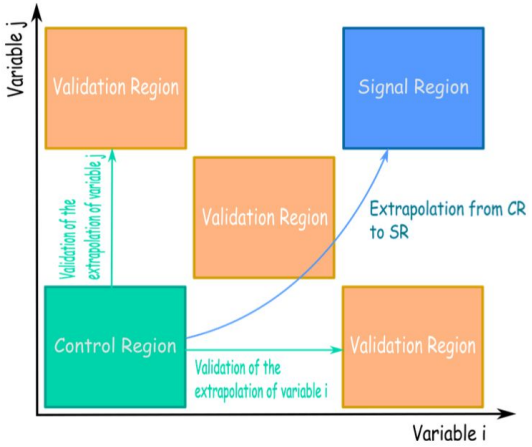
Pro tip: you can use the typical MVA tools to check the quality of your training (Kolmogorov - Smirnov distance, folding etc)

A word on data-driven corrections



- Sometimes we can bypass simulation all together
- **And if we can, we should**
  - No need to worry about data/simulation differences
  - Can take care of effects that the simulation doesn't even take into account
  - Not plagued by low generation statistics

- **Unfortunately, we don't know the truth about data, we can only approximate it**
  - Design regions of our data that give us a good approximation of the true process we are studying
  - For instance, if we want to study a specific background, we can apply cuts on our data that select it with high purity - this is a **Control Region/Sample**
  - The main challenge is usually how to extrapolate to our signal region - not a single recipe, mixture of data-driven and simulation approaches used (and often combined)
  - **Validation is key!**



An example

- We would like to measure the CP asymmetry between $B_s^0 \to K^+\pi^-$ and $B_s^0 \to K^-\pi^+$
- In the past two lectures, we saw how to setup the selection and fit, so getting the asymmetry should be simply getting the signal yields right?

An example

- We would like to measure the CP asymmetry between $B_s^0 \to K^+\pi^-$ and $B_s^0 \to K^-\pi^+$
- In the past two lectures, we saw how to setup the selection and fit, so getting the asymmetry should be simply getting the signal yields right?

Well… there might be other effects

- Production asymmetry
- Detection asymmetry
- **PID asymmetry**

An example

- We would like to measure the CP asymmetry between $B_s^0 \to K^+\pi^-$ and $B_s^0 \to K^-\pi^+$
- In the past two lectures, we saw how to setup the selection and fit, so getting the asymmetry should be simply getting the signal yields right?

Well… there might be other effects

- Production asymmetry
- Detection asymmetry
- **PID asymmetry**

Need to account for these effects in our model when extracting the asymmetry!

PID efficiency calculation

- Most of our data come out of the pre-analysis processing step with PID cuts already applied (in LHCb, this is the so-called stripping)
- Getting back the original number of candidates before the cut is not obvious
- Use dedicated data samples (calibration) without PID cuts pre-applied -> apply them on the fly and check the number of candidates before and after the PID selection!
- These samples are not necessarily the same decay channel or have different kinematics than our decay of interest… So we need to **extrapolate**
- Typically done via efficiency maps in key kinematic variables (P, ETA, track multiplicity, but can be any other variable that is crucial to the analysis)

Corrections and why we want them

What do we want to correct?

- False assumptions that are present in MC - simulation is not perfect!
  - Particularly important for estimates of efficiency
- False assumptions that are present in our interpretation of data is also not perfect!
  - For instance production/detection asymmetries, PID asymmetries

How do we correct?

- Compare data and MC and reweight where needed
  - Need a data background-subtraction and reweighting methods
- Define data-driven control modes/regions to extract corrections
  - Need to find regions that are orthogonal to our signal region + a way to extrapolate from control region to signal region

Systematic uncertainties

What is a systematic uncertainty (or error)?

What is a systematic uncertainty (or error)?

- **It IS:** the uncertainty on estimating systematic effects such as background, scanning efficiency, energy resolution, variation of counter efficiency with beam position and energy, dead time, etc.

- **It is NOT:** a reproducible inaccuracy introduced by faulty equipment, calibration, or technique. This is a MISTAKE (and should be corrected)

Example

- If you measure a potential of 12.3 V as 12.4 V, with a voltmeter accurate to 0.1V, that is fine. Even if you measure 12.5 V. If you measure it as 124 V, that is a mistake

How to find systematic uncertainties?

- Every time you make a choice in the life of your analysis you should ask yourself:
- Is there an equally (or close-to-equally) valid alternative choice I could be making ?
    - Examples: Choice of fit model, binning scheme in calculation of efficiencies…
- What is the effect of assuming the alternative?
- Stress on “equally”: exploring inappropriate alternatives does not make sense!

How to deal with them?

Various ways, depending on the type of systematic

How to deal with them?

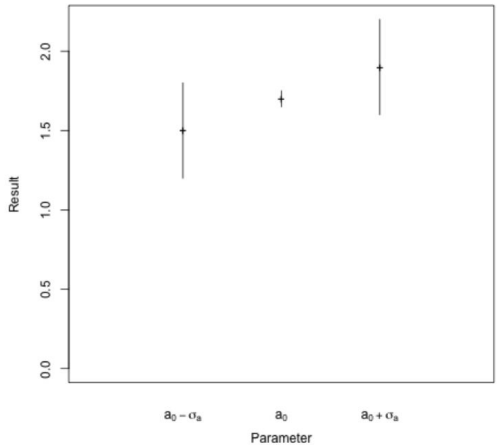
Various ways, depending on the type of systematic

- If they are **continuous** and **explicitly relate** to your measurement, you can simply propagate them using standard algebra
    - For instance: luminosity or efficiency uncertainty
    - Don’t forget the correlations!

How to deal with them?

Various ways, depending on the type of systematic

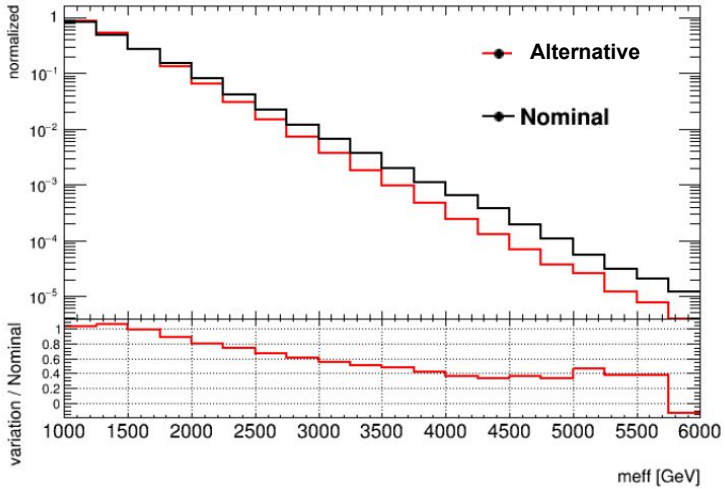
- If they are **continuous** but **do not explicitly relate** to your measurement, you cannot use algebra - what is typically done is to vary by +-sigma and see what happens to your result
    - Example: MC tuning parameters
    - Or take many Gaussian samples of parameter value and look at distribution of result. Nice, if you have the computing capacity



How to deal with them?

Various ways, depending on the type of systematic

- If they are **discrete**, usually one (or more) alternative points can be used to check the difference in the final result
  - Example: choice of fitting model, MC generator
  - Important to understand if alternatives are equal or a preference exists



For the case of 2 equal-preference models:

$$\frac{R_1+R_2}{2} \pm \left| \frac{R_1-R_2}{\sqrt{2}} \right|$$

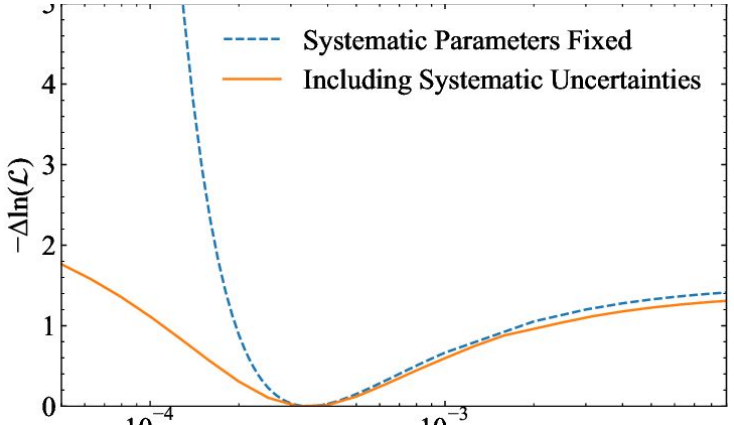
Including systematic uncertainties in our model

- Nuisance parameters: parameters that we include in our fit model, but which we don't really care about - perfect for systematic uncertainties
- Assuming nuisance parameters θ_j are independent, the total PDF is:

$$C_{Syst}(\theta^0, \theta) = \prod_{j \in \mathrm{SU}} G(\theta_j^0, \theta_j)$$

Where θ_0 are the nominal values around which we vary θ

- Usually one can assume a Gaussian constraint
- Check out the lecture of Vitalii yesterday on how to fit with nuisance parameters!



Checking the analysis

- Systematic uncertainties are not mistakes. We saw how to find and evaluate systematics… but what about mistakes?
- Statistical tools might give a hint for a mistake but not always
- That’s where the analysis checks come in!
- Design null-tests of your analysis - tests that you expect to give the same result as your main analysis strategy
    - Repeat analysis by splitting dataset by years, magnet polarity etc
    - Different selection cuts
    - Histogram binnings and fit settings
    - And whatever else you (or your reviewers) can think of!

Checking the analysis

If the check passes the test :)

- Tick the box and move on
- No need to assign a systematic uncertainty
  - It's illogical
  - It inflates errors
  - It penalizes diligence

If the check fails the test :(

- Worry! This could be a hint of bigger problem
- Check that the problem is not the test itself
  - If it is, fix it and repeat
- Check the analysis and try to find the problem
  - If you find it, fix it!
  - If not, worry more, try to check with other experiments
- Only as a last resort, you should assign a systematic uncertainty

The difference between evaluating a systematic and a performing check

What's the difference between?

Evaluating implicit systematic errors: vary lots of parameters, see what happens to the result, and include in systematic error

Checks: vary lots of parameters, see what happens to the result, and don't include in systematic error

(1) Are you expecting to see an effect? If so, it's an evaluation, if not, it's a check

(2) Do you clearly know how much to vary them by? If so, it's an evaluation. If not, it's a check.

Cover cases such as trigger energy cut where the energy calibration is uncertain - may be simpler to simulate the effect by varying the cut.

Backup

Minimization of variation

Previous part allows one to get the correct result. But there is still no reason for linearity given, we just *assumed this*..

Apart from having a correct mean, we should also minimize variation of any reconstructed variable. Let's try to optimize it

$$\mathbb{V}\, X = \sum_x sw_s(x)^2\, \mathbb{V}\, 1_{x \in bin} = \sum_x sw_s(x)^2\, (p_s p_s(x) + p_b p_b(x))(1 - p_s p_s(x) - p_b p_b(x))$$

A bit complex, isn't it? Instead of optimizing such a complex expression (which is individual for each bin), let's minimize it's **uniform upper estimate**

$$\mathbb{V}\, X = \sum_x sw_s(x)^2\, \mathbb{V}\, 1_{x \in bin} \leq \sum_x sw_s(x)^2$$

so if we are going to minimize this upper estimate, we should solve the following optimization problem with constraints:

$\sum_x sw_s(x)^2 \to \min$
$\sum_x sw_s(x)\ p_b(x) = 0$
$\sum_x sw_s(x)\ p_s(x) = N_{sig}$

Let's write lagrangian of optimization problem:

$$\mathcal{L} = \sum_x sw_s(x)^2 + \lambda_1 \left[\sum_x sw_s(x)\ p_b(x)\right] + \lambda_2 \left[\sum_x sw_s(x)\ p_s(x) - N_{sig}\right]$$

After taking derivative with respect to $sw_s(x)$ we get the equality:

$$0 = \frac{\partial \mathcal{L}}{\partial\ sw_s(x)} = 2sw_s(x) + \lambda_1 p_b(x) + \lambda_2 p_s(x)$$

which holds for every x. Thus, after renaming for convenience $a_1 = -\lambda_1/2$, $a_2 = -\lambda_2/2$, we confirmed linear dependency.