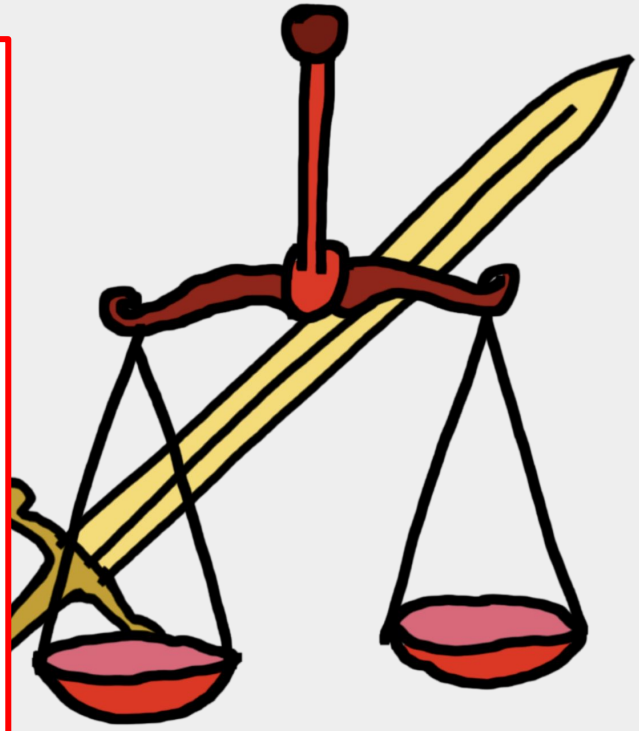
Getting More Power from Anomaly Detectors for *Free* with Weights

Rikab Gambhir
gambhirb@ucmail.uc.edu
In Collaboration with: *Radha Mastandrea* and *Jesse Thaler*



Alternate Talk Title:

The Least Interesting Thing to do with a Classifier is Classify.



Rikab Gambhir
gambhirb@ucmail.uc.edu
In Collaboration with: *Radha Mastandrea* and *Jesse Thaler*

PRELIMINARY



What I want to convince you of:

Weight-and-Count is *better* than Cut-and-Count.

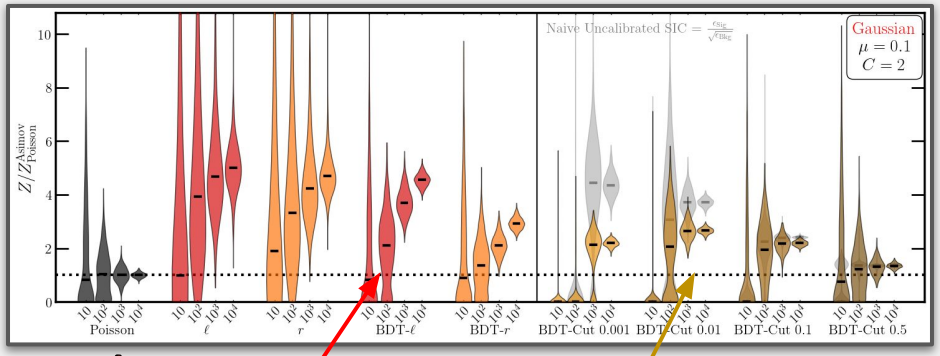
Weights weigh way more than cuts

It is a *free* improvement to expected significance in bump hunts over cutting

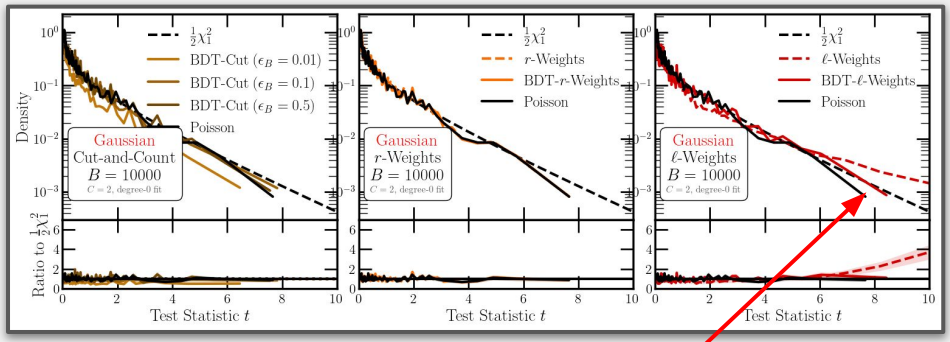
Free also includes calibration. We can show and prove this test is asymptotically well-calibrated and returns good *p*-values

(i.e. a reported Z-sigma significance is really Z sigmas).

Zero reason to not for you do this!



Weight-and-Count beats Cut-and-Count on practical examples

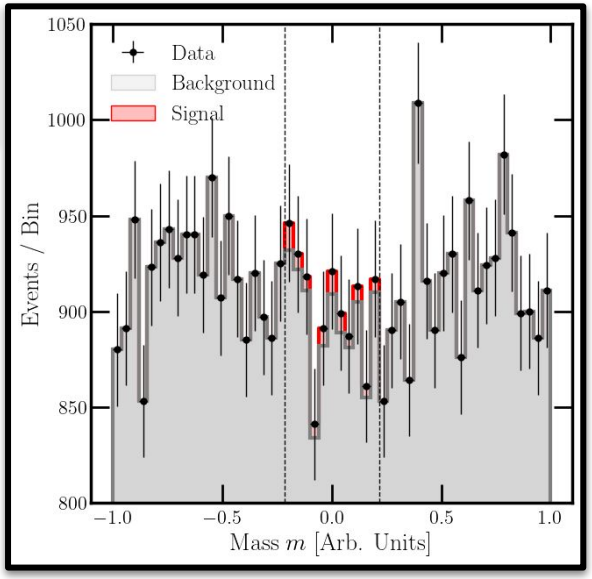


… while still being automatically well-calibrated (can easily report genuine *p*-values)

$Z = 1.0\sigma$

(By construction, for this toy)

$$Z \lesssim \frac{S}{\sqrt{B}}$$

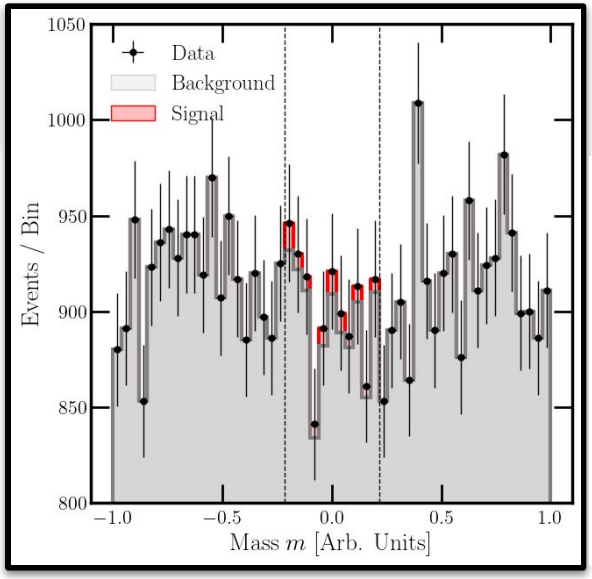


Let me walk you through a toy example!

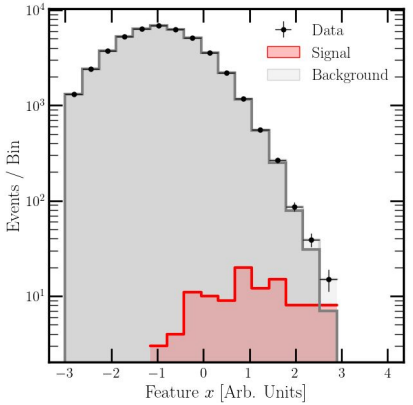
I want to elevate this **red signal**, located in a narrow signal region

$Z = 1.0\sigma$

$$Z \lesssim \frac{S}{\sqrt{B}}$$



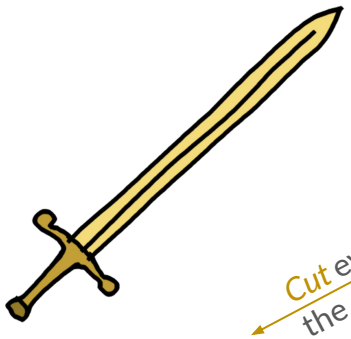
Gaussian Toy Features to cut on



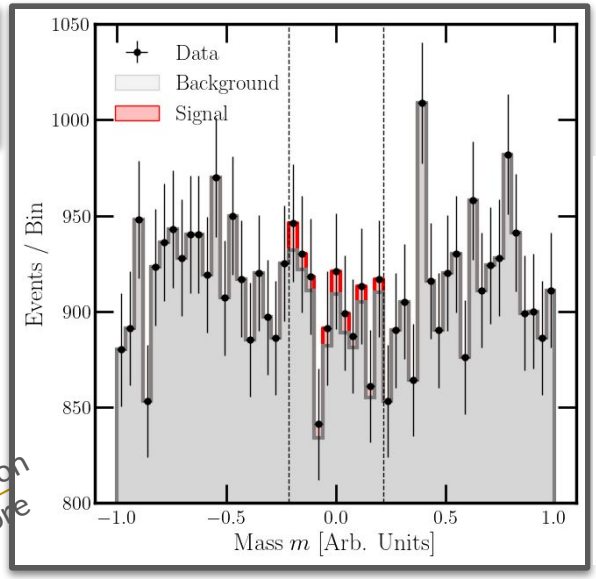
Like a normal Cut-and-Count, we train a **classifier** to tag Signal vs. Background (or Data vs. Background) using auxiliary features *x*.

$Z = 1.0\sigma$

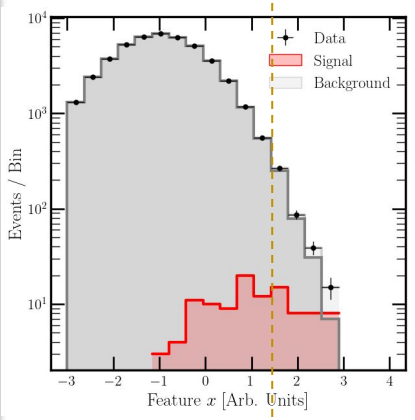
$$Z \lesssim \frac{S}{\sqrt{B}}$$



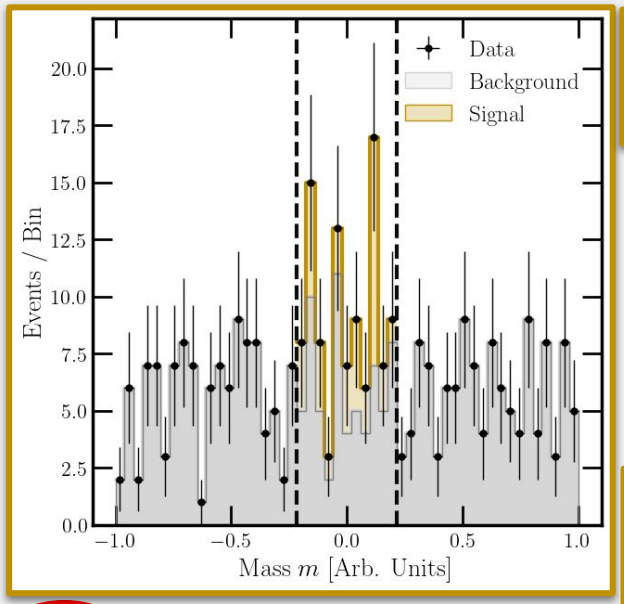
Cut events based on the classifier score



Gaussian Toy Features to cut on



Cut at $x = 1.5$, approximately the optimal analytic cut



$Z = 3.9\sigma$

(See Marie's Plenary earlier for a refresher!)

Standard Method: **Cut-and-Count**

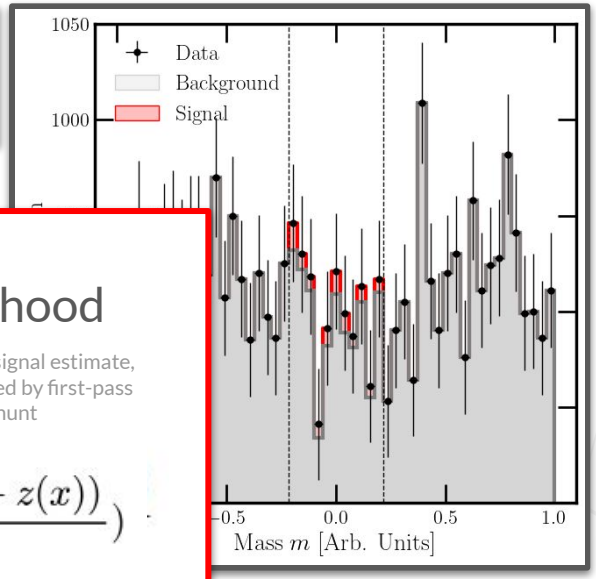
Throw away events that don't make the cut.

We lose stats, but hopefully more signal survives than background!

$$\frac{Z_{\text{New}}}{Z} \approx \frac{\text{TPR}}{\sqrt{\text{FPR}}}$$

Detail: This is an IAD bump hunt with a smooth background fit and weakly supervised BDT

$Z = 1.0\sigma$



Magic Weighting Function

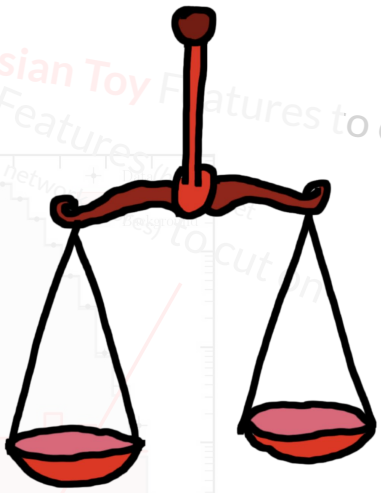
Signal-To-Background Likelihood

Data - Background Classifier (BDT) score

Naive signal estimate, obtained by first-pass bump-hunt

$$w(x) = \mathrm{ReLU}\left(\frac{z(x) - (1-\mu)(1-z(x))}{\mu(1-z(x))}\right)$$

[Will explain this formula shortly]



Weight events based on the classifier score

New Method: **Weight-and-Count**

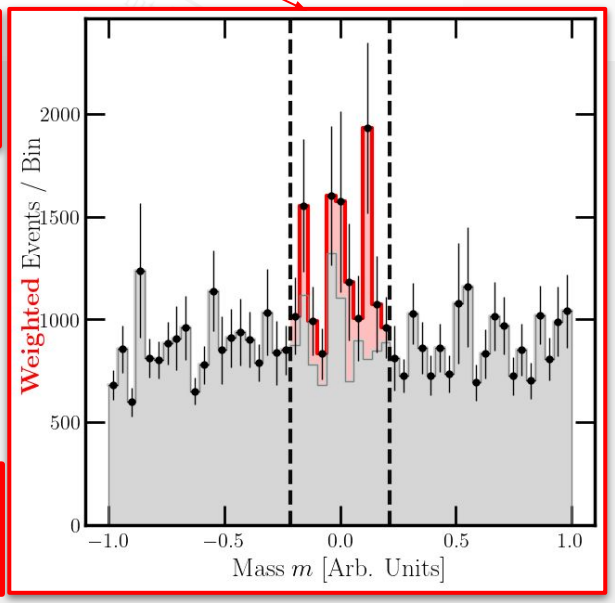
$Z = 6.6\sigma$

Assign every event a **weight** *w(x)* – high if signal-like, low if background-like.

Replace Poissonian Likelihood with **Weighted-Poissonian** Likelihoods.

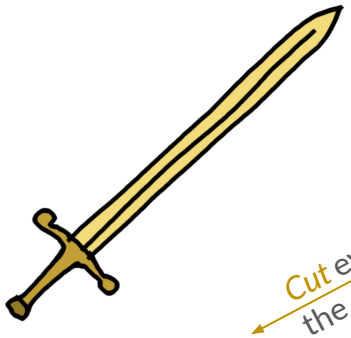
[definition to come]

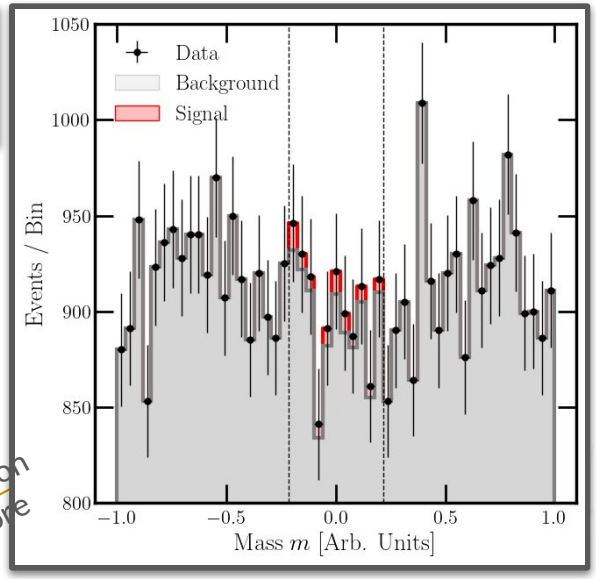
$$\frac{Z_{\mathrm{New}}}{Z} \approx \frac{\langle w \rangle_S}{\sqrt{\langle w^2 \rangle_B}}$$

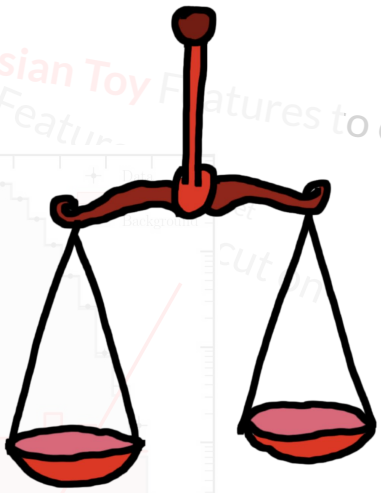


$Z = 1.0\sigma$

$$Z \lesssim \frac{S}{\sqrt{B}}$$

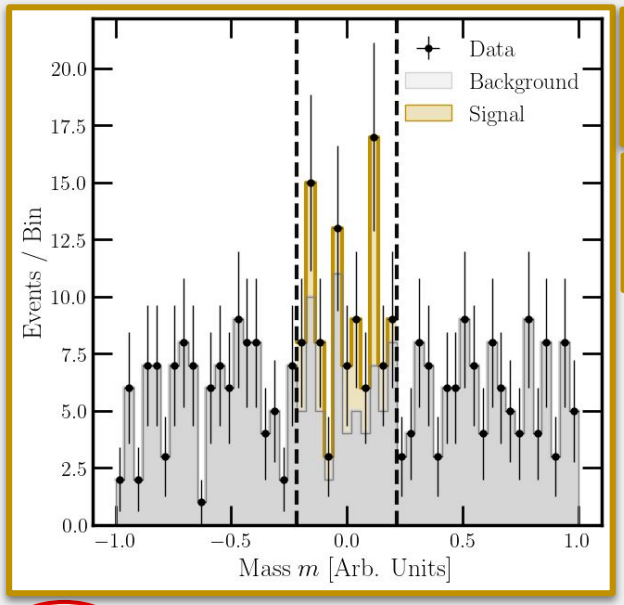






Cut events based on the classifier score

Weight events based on the classifier score

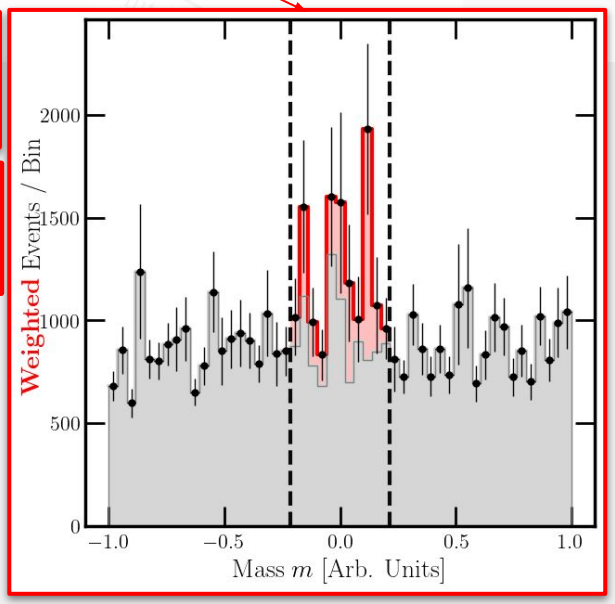


$Z = 3.9\sigma$

$$\frac{Z_{\text{New}}}{Z} \approx \frac{\text{TPR}}{\sqrt{\text{FPR}}}$$

$Z = 6.6\sigma$

$$\frac{Z_{\text{New}}}{Z} \approx \frac{\langle w \rangle_S}{\sqrt{\langle w^2 \rangle_B}}$$



Clear Winner

Using the *same* classifier!

Anatomy of a Cut-and-Count Search

Let's write a *hypothesis test* and construct a test statistic.

Neyman-Pearson tells us to use:

Nuisance Parameters characterizing unknown smooth background

$$t = -2\log\left(\frac{P(\text{ Data } | \text{ Signal})}{P(\text{ Data } | \text{ No Signal})}\right)$$

$$= -2\left(\sum_{i\in\text{Bins}} \log \text{Pois}(N_i, \lambda_B(\theta), \lambda_s) - \sum_{i\in\text{Bins}} \log \text{Pois}(N_i, \lambda_B(\theta), 0)\right)$$

(Ignoring details about maxima and definition of fits. Ask later if you want!)

Anatomy of a Cut-and-Count Search

Let's write a *hypothesis test* and construct a test statistic.

Neyman-Pearson tells us to use:

Nuisance Parameters characterizing unknown smooth background

$$t = -2\log\left(\frac{P(\text{ Data } | \text{ Signal})}{P(\text{ Data } | \text{ No Signal})}\right)$$

$$= -2\left(\sum_{i\in\text{Bins}} \log \text{Pois}(N_i, \lambda_B(\theta), \lambda_s) - \sum_{i\in\text{Bins}} \log \text{Pois}(N_i, \lambda_B(\theta), 0)\right)$$

(Ignoring details about maxima and definition of fits. Ask later if you want!)

Then, we can extract a *p*-value (significance level) and estimate sensitivity if we know:

$$p(t|\lambda_s = 0)$$

Necessary to define a *p*-values — "calibration"
Usually need either expensive pseudo experiments or **Wilks' Theorem** to know!

~~Cut~~Weight-and-Count Anomaly Detection [Details]

The effect of weighting is to replace *Poissons* with *Weighted (Compound) Poissons*

$$N_{\text{cut}} = \sum_i^N \Theta(f(x_i) - f_{\text{cut}}) \longrightarrow N_w = \sum_i^N w(x_i)$$

Observable: Bin Count *N*

Observable: Weighted Bin Count, depends on *N and* $\langle w \rangle$

$$\text{Pois}(N; \lambda) \longrightarrow \text{CPD}(N_w; \lambda, p(w))$$

Poissonian Bin Counts

Compound-Poissonian Bin Counts
Exact distribution of sum-of-weights

Approx

Treat as Nuisance

$$\text{SPD}(N_w; \lambda, \langle w^2 \rangle)$$

Tractable approximation to CPD

Net effect is to alter the first and second moments of the Poisson: Nice in the asymptotic limit!
Weighted Bump Hunts!

Weight each event *x* using some weight function *w(x)*.

The *best* weight function is the signal-to-background likelihood ratio, but any weight function is valid!

Cuts are basically just weights, anyways

Cutting is equivalent to just assigning weights of 0 or 1. The Scaled Poisson returns back to a Poisson in this case.

But it loses information about *how* signal-like or background like events are, so generic continuous weights are better.

Plus, we don't have to choose a cut point!

Our method completely reproduces ordinary cutting when weights are 0 and 1

Weights weigh way more than cuts

Modulo having to profile the extra weight variance parameter, which is an extremely tiny power hit

*Anything reasonable — no weight distributions with infinite variance or extreme skews!

[Neyman, Pearson; On the problem of the most efficient tests of statistical hypotheses]
[**RG**, Mastandrea, Nachman, Thaler; 2502.14036]

The *Optimal* Weights

What weight function should we choose? Anything is valid!*

Classifier cuts are also weights, but not the optimal (most powerful) ones

Classifier scores are *more than just scores* — by Neyman-Pearson, they're **Likelihood Ratios**!

Claim: Choosing the weight function to be the signal-to-background per-event likelihood ratio provides (near)-optimal sensitivity!

Optimal in the single-bin counting experiment limit with known likelihood ratio

This is essentially a *free* increase in sensitivity just by using the fact that classifiers learn the likelihood ratio, plus we don't have to pick a cut working point!.

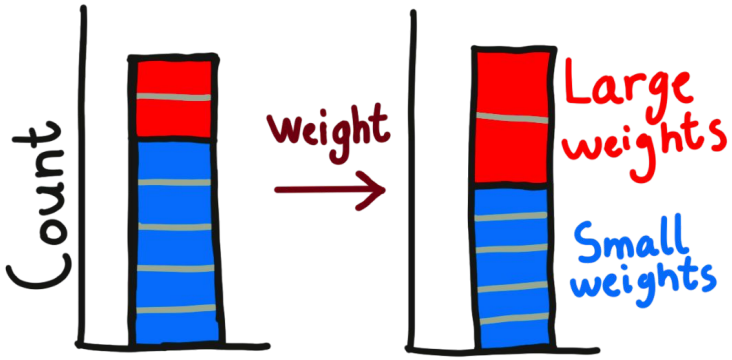
The worst thing to do with a classifier is classify!

[Neyman, Pearson; On the problem of the most efficient tests of statistical hypotheses]
[RG, Mastandrea, Nachman, Thaler; 2502.14036]

[Freytsis, Ovanesyan, Thaler; 0909.2862]

Sketch of Proof

For simplicity, assume we are doing a single-bin counting experiment in the asymptotic limit with known signal.* Let's allow every event to have a weight w**.



The total number of events is now:

$$S_{\text{eff}} = \int dx\, S(x)w(x), \quad B_{\text{eff}} = \int dx\, B(x)w(x)$$

The "significance" in big quotes! is:

$$Z = \frac{\int dx\, S(x)w(x)}{\sqrt{\int dx\, B(x)w^2(x)}} \quad \text{(Generalizes } Z = \frac{S}{\sqrt{B}}\text{)}$$

The Likelihood Ratio!

We can then calculate what choice of weights optimizes the significance:

$$w(x) = \frac{S(x)}{B(x)} = \frac{N_{\text{Sig}}}{N_{\text{Bkg}}} \frac{p_{\text{Sig}}(x)}{p_{\text{Bkg}}(x)}$$

Unimportant Constants

*In the full analysis, we do a complete Poisson likelihood treatment with nuisance parameters due to fits. When these are included, these weights may not be strictly optimal, but they are still better than nothing!
**Ordinary cutting can be seen as assigning a weight of 0.

The worst thing to do with a classifier is classify!

Last Ingredient: Estimating the Likelihood

We are *already* training BDT's anyways to perform the CWoLa-style cuts.
Neyman-Pearson tells us we learn the likelihood ratio out of this!

Classifier score

Naive signal estimate, obtained by first-pass Poisson bump-hunt

$$\ell(x) \equiv \frac{z(x) - (1-\mu)\,(1-z(x))}{\mu\,(1-z(x))} = \frac{p_{\mathrm{Sig}}(x)}{p_{\mathrm{Bkg}}(x)}$$

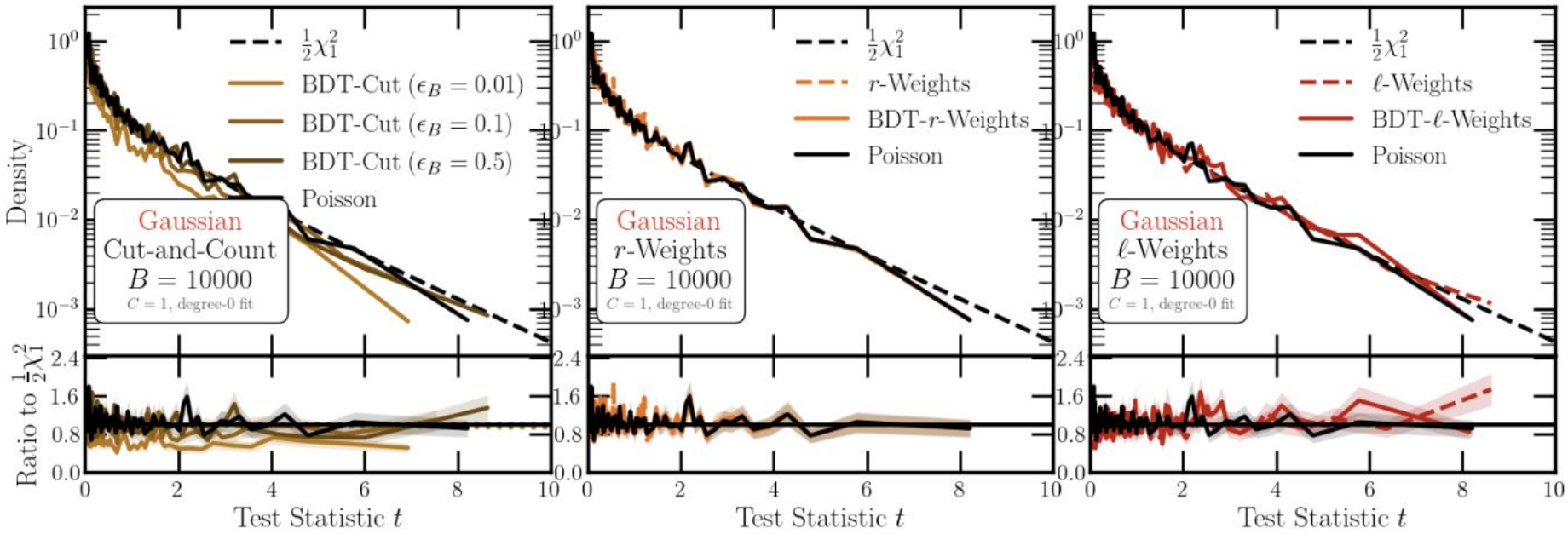
Technical detail: Both the classifier score z and the signal fraction μ are imperfect estimators, so the weights might go negative! But we can just cut those out — we were cutting things in the original analysis anyways! Reduces optimality, but not correctness!

Calibration and Wilks' Theorem

Fact: A test statistic is *only* meaningful if you know its distribution under the null

For this reason, it can be dangerous to talk only about SIC. It is not a calibrated test statistic and can overestimate true significance, especially in the presence of uncertainties.
We advocate for the AD community to report calibrated test statistics!

Our test statistic satisfies Wilks' theorem: so the calibration really is *free** even for imperfect weights



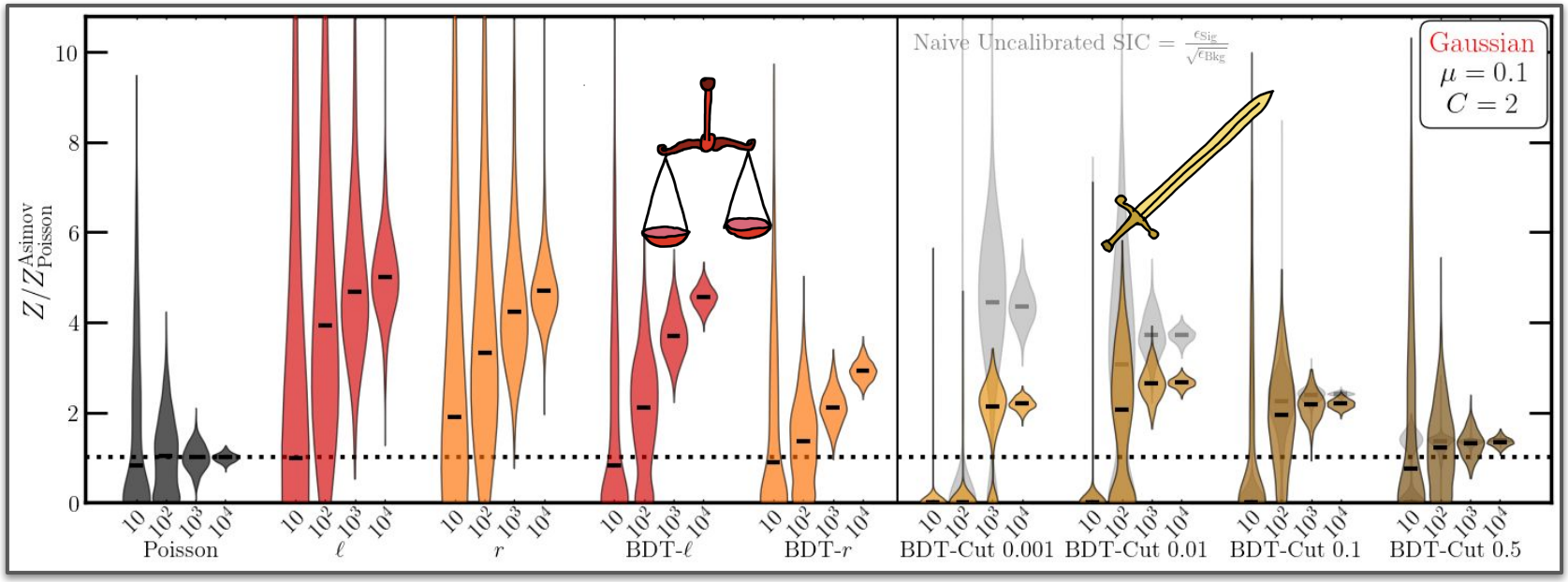
*We must assume that the original Poisson test also satisfies Wilks', and that our weights are not so heavy-tailed that SPD fails to approximate CPD

Power and Significance [Gaussian Toy]

Having checked calibration, we can now compute the *expected significance* of a signal.

This is our "so what?" — Weights give you higher expected significance!

Significance over Poisson



Poisson

Exact Likelihood

Different Weights Ask me later

BDT Likelihood

Different Weights Ask me later

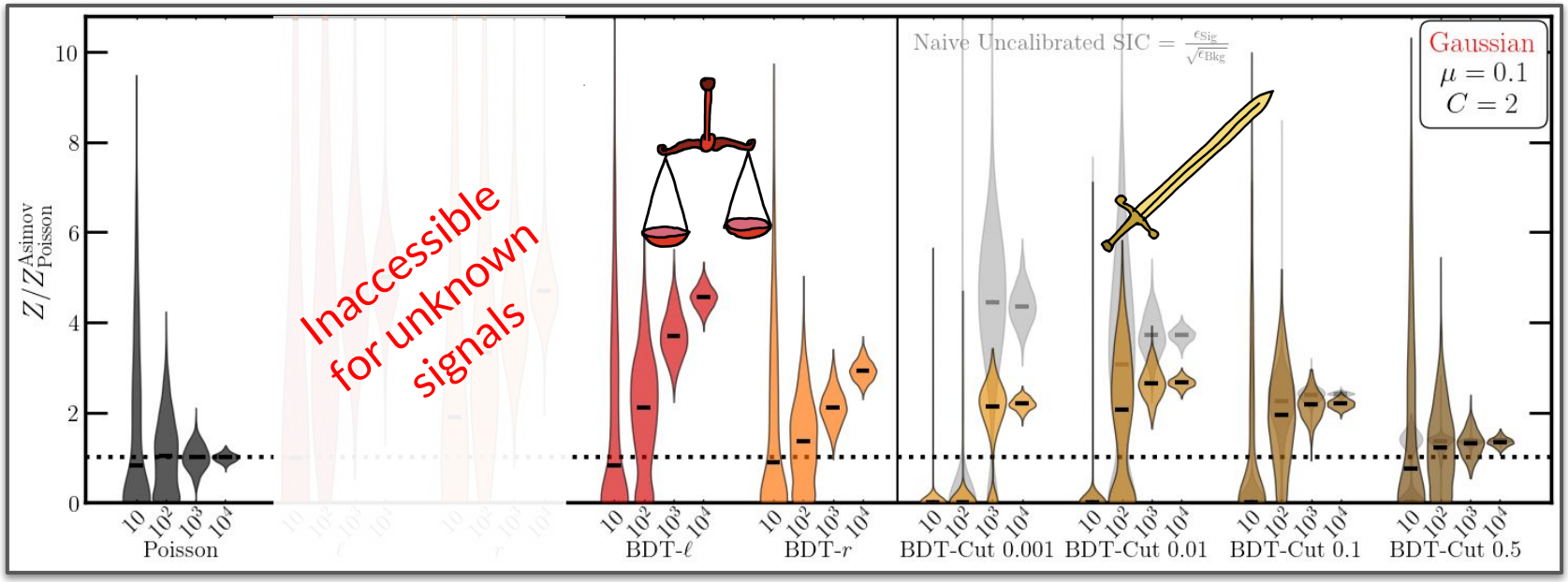
Cut and Count

Power and Significance [Gaussian Toy]

Having checked calibration, we can now compute the *expected significance* of a signal.

This is our "so what?" — Weights give you higher expected significance!

Significance over Poisson



Poisson

Exact Likelihood

Different Weights Ask me later

BDT Likelihood

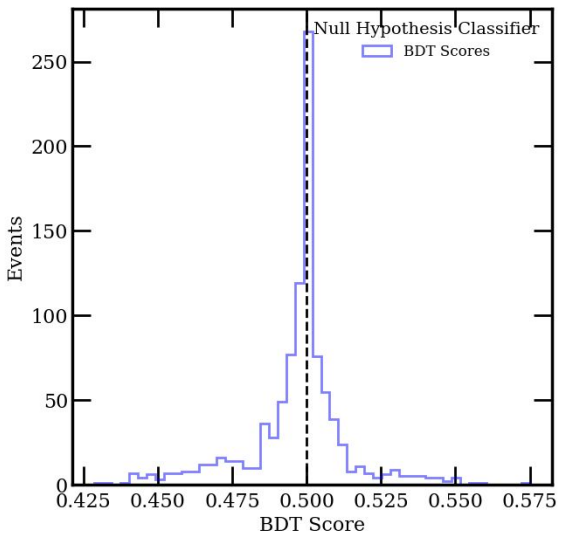
Different Weights Ask me later

Cut and Count

Objection: "BDT's are Fallible" – That's OK!

A real-life BDT does *not* learn the likelihood ratio. Even a BDT between two identical datasets will not learn a likelihood ratio of 1, it will be 1 ± noise.

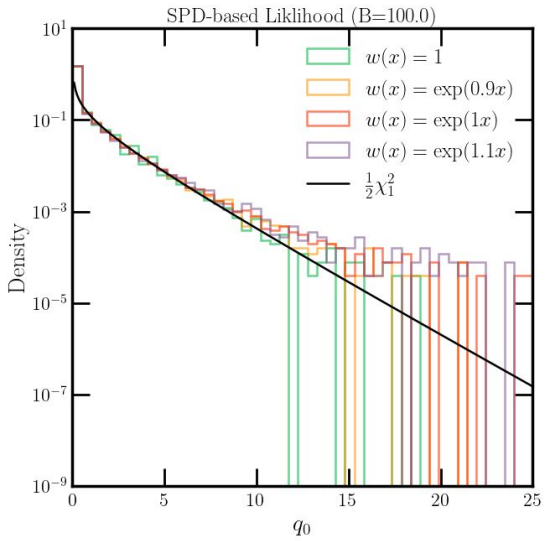
But this is OK! Any weights work, and produce the same test statistic distribution under the null. You may not have the *most powerful* weights anymore, but you will not get an incorrect result.



Example: Gaussian Signal on a Gaussian background

Constructed such that the true likelihood ratio is exp(*x*)

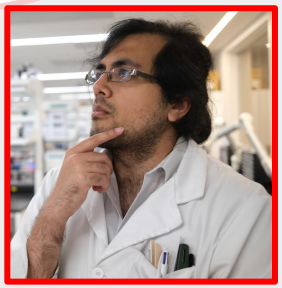
But mismodeling does not significantly change the distribution of the test statistic![*]



[*]At low significance values
[**]Extremely high variance can begin to ruin things

[Aside]:

"Why bother with all this weighting stuff? If I have the likelihood ratio per-event, isn't that *already* the most powerful hypothesis test?"



[Me learning about AD, confused why we do cuts and not just the full likelihood if we have classifiers]

The *full* information we have access to is the *set* of all per-event likelihoods

$$\mathcal{L}(\lambda; N, x_i) = \log \text{Pois}(N; \lambda) + \sum_i^N \log(\ell(x))$$

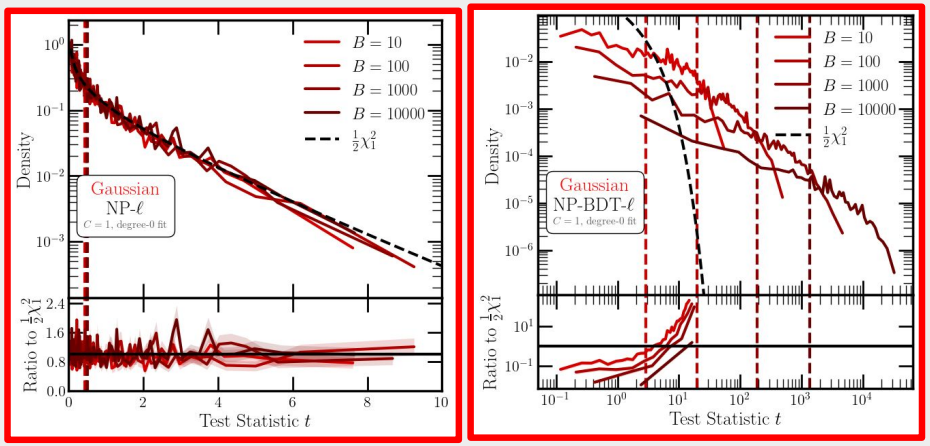
NP tells us this is powerful test statistic.

But it isn't ***stable*** — if ℓ is slightly off, the entire distribution of the test statistic is ruined. Extremely hard to calibrate!

But the sum of weights is more robust!

Toy model: Gaussian Likelihoods

True Likelihood **BDT Likelihood**



Test Statistic Distribution

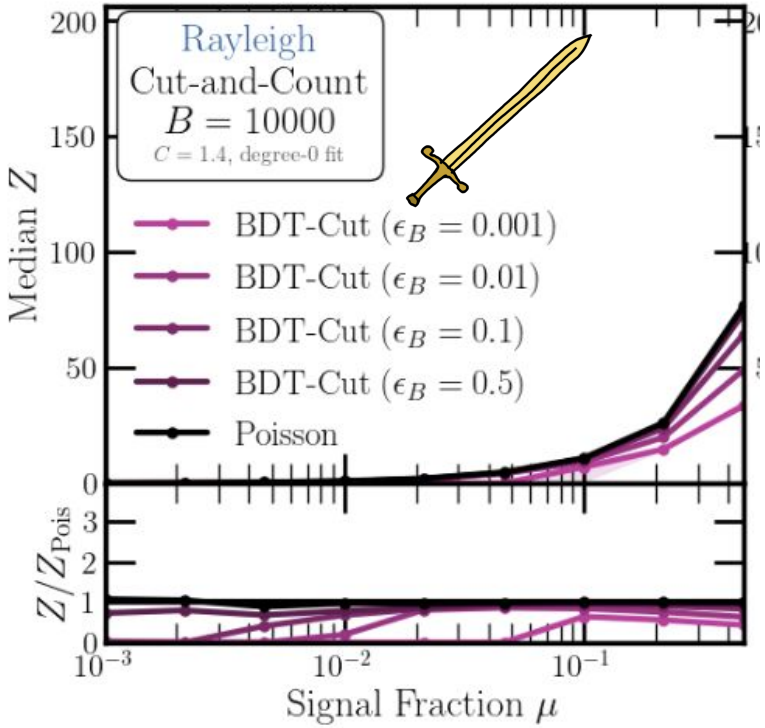
Extreme Example: Rayleigh Distributions

$$\mathcal{R}(x) = \frac{x}{\sigma^2} e^{-x^2/2\sigma^2}$$

If the signal and background both follow **Rayleigh Distributions** with different widths
(fun fact, this approximately happens discriminating quarks and gluons with angularities @LL)

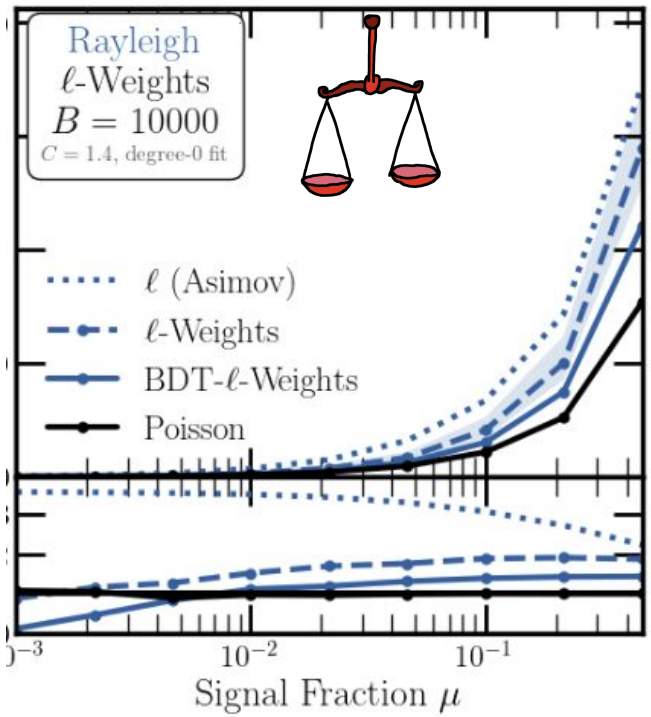
A cut is *never* worth it!

Can calculate: SIC ≤ 1, always!



But a weight is!

Can calculate: Improvement ≳ 1

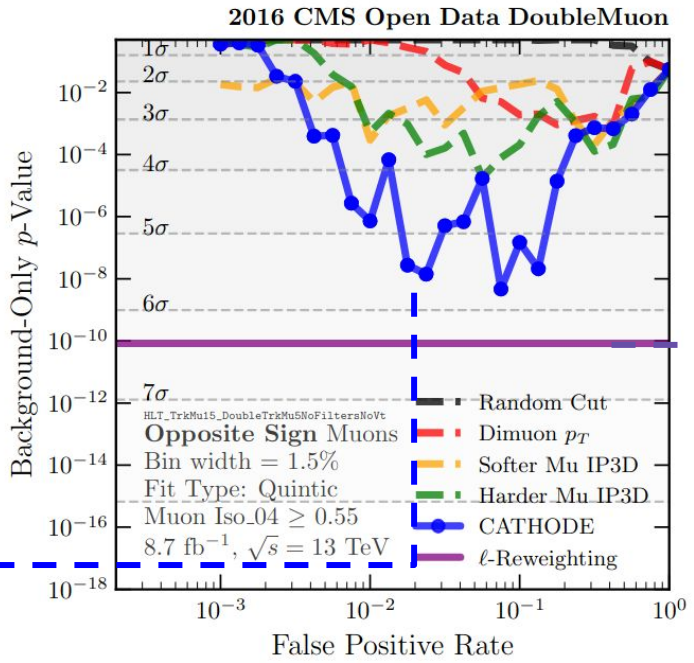


[**RG**, Mastandrea, Nachman, Thaler; 2502.14036]
[1][Hallin et. al.; 2109.00546]

(By Radha!)

Last Year @ ML4Jets:
Real Life Example

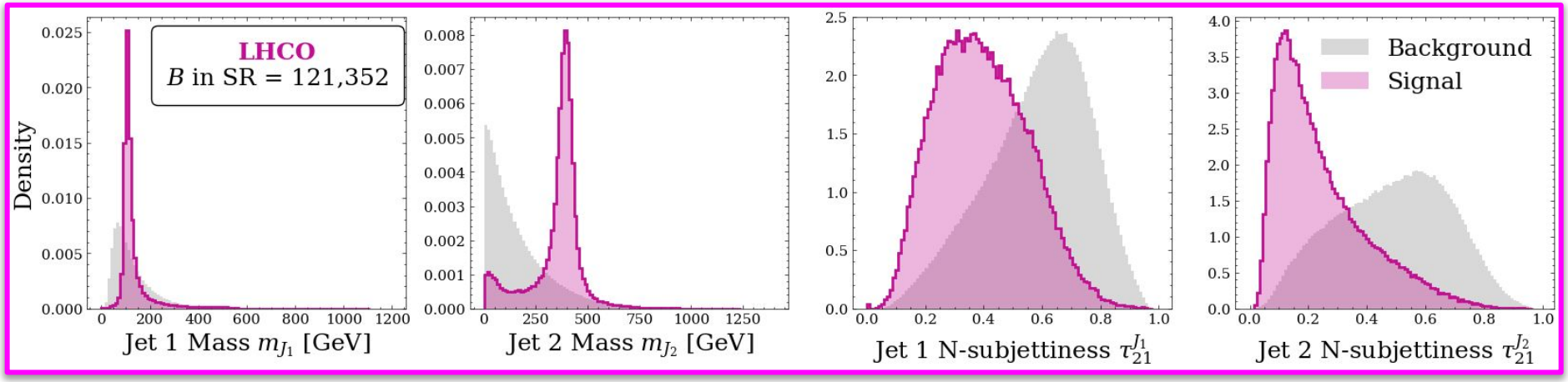
Isolating UnIsolated Upsilons



5.7σ Excess over a smooth background!

Last year: Early version of this weights method.
More developed and sophisticated now, including weight profiling

likelihood-based reweighting! → 6.4σ ←



(See Marie's Plenary earlier for a refresher!)

What we are still working on

We are in the process of performing a *fully calibrated bump hunt* on the ***Large Hadron Collider Olympics*** dataset.

Work in progress! Stay tuned!

Conclusion:

Weight-and-Count is *better* than Cut-and-Count.

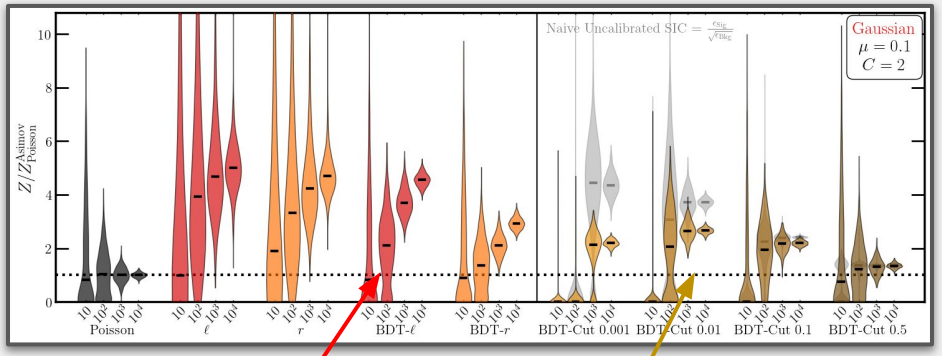
It is quick, it is easy, and it is free if you already had a classifier to cut on. More powerful than cutting while guaranteeing calibrated *p*-values

Zero reason not to do this!

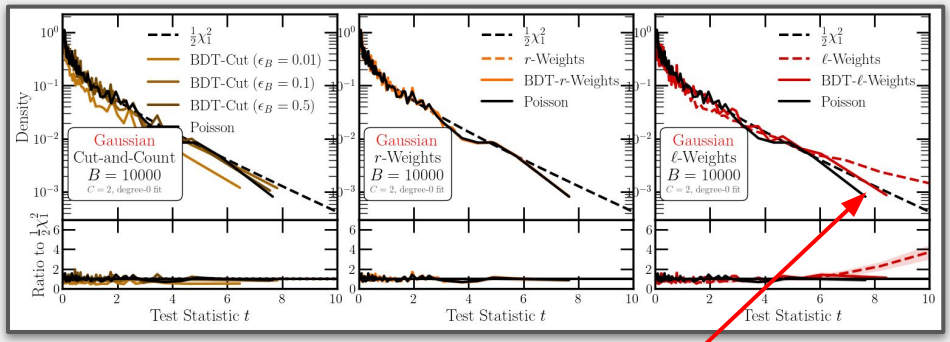
Weights weigh way more than cuts

Questions, comments, concerns?
Email me at *gambhirb@ucmail.uc.edu*

Based on [**RG**, Mastandrea, Thaler; To appear soon!]



Weight-and-Count beats Cut-and-Count on practical examples



… while still being automatically well-calibrated (can easily report genuine *p*-values)

We also encourage the community to report calibrated *p*-values / significances alongside SIC's!

Appendices

Full Likelihoods: Poisson

Per Bin:

$$-2\log\mathcal{L}(N_i|\lambda_i) = -2\left(-\lambda_i + N_i\log\lambda_i - \log\Gamma(N_i+1)\right)$$

Over Many Bins:

$$\log(\mathcal{L})(\{N\}|\lambda^{\theta}_{\mathrm{Bkg}},\lambda_{\mathrm{Sig}}) = \sum_{i\in\mathrm{SB}}\log(\mathcal{L})(N_i|\lambda_{\mathrm{Bkg},i}) + \sum_{j\in\mathrm{SR}}\log(\mathcal{L})(N_j|\lambda_{\mathrm{Bkg},j}+\lambda_{\mathrm{Sig}})$$

Full profile test statistic:

$$t^{\mathrm{Pois}}(\{N\}) = -2\Bigg(\underbrace{\max_{\lambda^{\theta}_{\mathrm{Bkg}}}\log(\mathcal{L})(\{N\}|\lambda^{\theta}_{\mathrm{Bkg}},0)}_{\text{“Null Term”}} - \underbrace{\max_{\lambda^{\theta}_{\mathrm{Bkg}},\lambda_{\mathrm{Sig}}}\log(\mathcal{L})(\{N\}|\lambda^{\theta}_{\mathrm{Bkg}},\lambda_{\mathrm{Sig}})}_{\text{“Alternate Term”}}\Bigg)$$

Full Likelihoods: Scaled Poisson

Per Bin:

$$\log(\mathcal{L})(W|\lambda,\kappa) = -\frac{\lambda\mathbb{E}[w]}{\kappa} + \frac{W}{\kappa}\log\left(\frac{\lambda\mathbb{E}[w]}{\kappa}\right) - \log\Gamma(\frac{W}{\kappa}+1) - \log(\kappa)$$

Over Many Bins:

$$\log(\mathcal{L})(\{N\}|\lambda^{\theta}_{\text{Bkg}},\lambda_{\text{Sig}}) = \sum_{i\in\text{SB}}\log(\mathcal{L})(N_i|\lambda_{\text{Bkg},i}) + \sum_{j\in\text{SR}}\log(\mathcal{L})(N_j|\lambda_{\text{Bkg},j}+\lambda_{\text{Sig}})$$

no separate κ for signal, this is a modeling choice

Full profile test statistic:

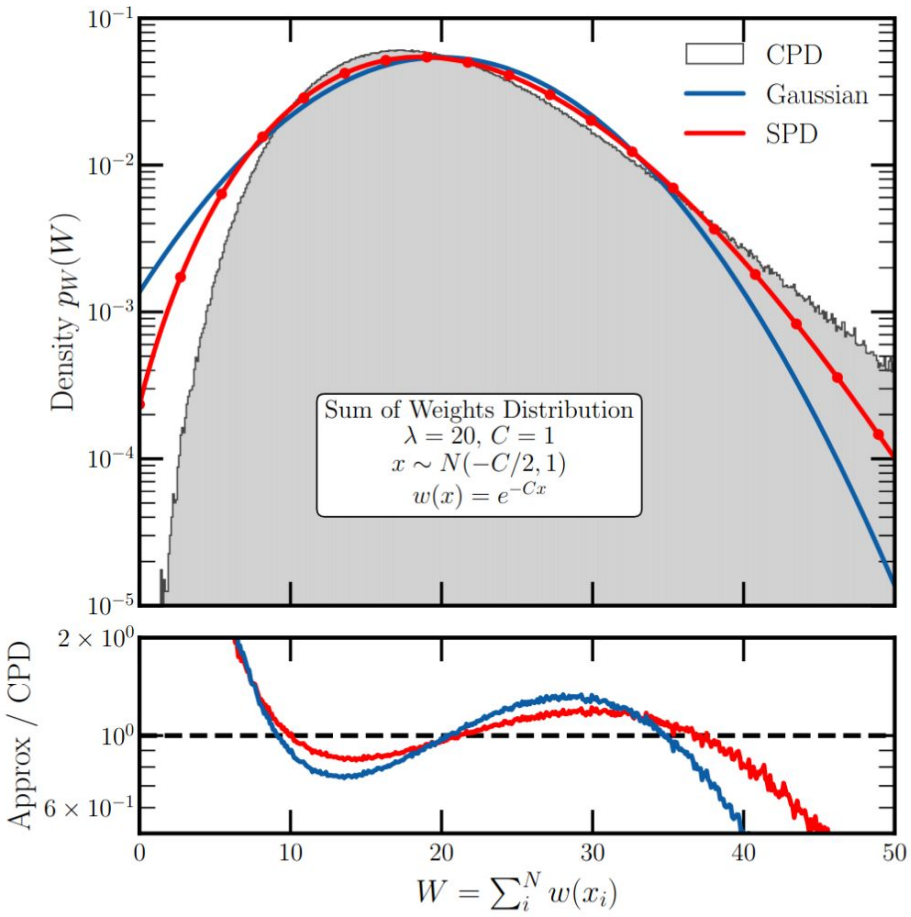
$$t^{\text{SPD}}(\{W\}) = -2\Bigg(\underbrace{\max_{\lambda^{\theta}_{\text{Bkg}},\kappa^{\theta}_{B}}\log(\mathcal{L})(\{N\}|\lambda^{\theta}_{\text{Bkg}},0,\kappa^{\theta}_{B})}_{\text{“Null Term”}} - \underbrace{\max_{\lambda^{\theta}_{\text{Bkg}},\lambda_{\text{Sig}},\kappa^{\theta}_{B}}\log(\mathcal{L})(\{N\}|\lambda^{\theta}_{\text{Bkg}},\lambda_{\text{Sig}},\kappa^{\theta}_{B})}_{\text{“Alternate Term”}}\Bigg)$$

CPD vs SPD Comparison

SPD gets the first two cumulants of CPD correct, always underestimates third cumulant *but, not as bad as a Gaussian approximation!*

This is why weights can't be too heavy-tailed.

Wilks' fails due to mismodelling, so SPD better be a good approximation.



vs. Neyman-Pearson

Using weights (BDT or God-Given) is about just as good as using the full NP test for moderate signal strengths

But it is far more robust!