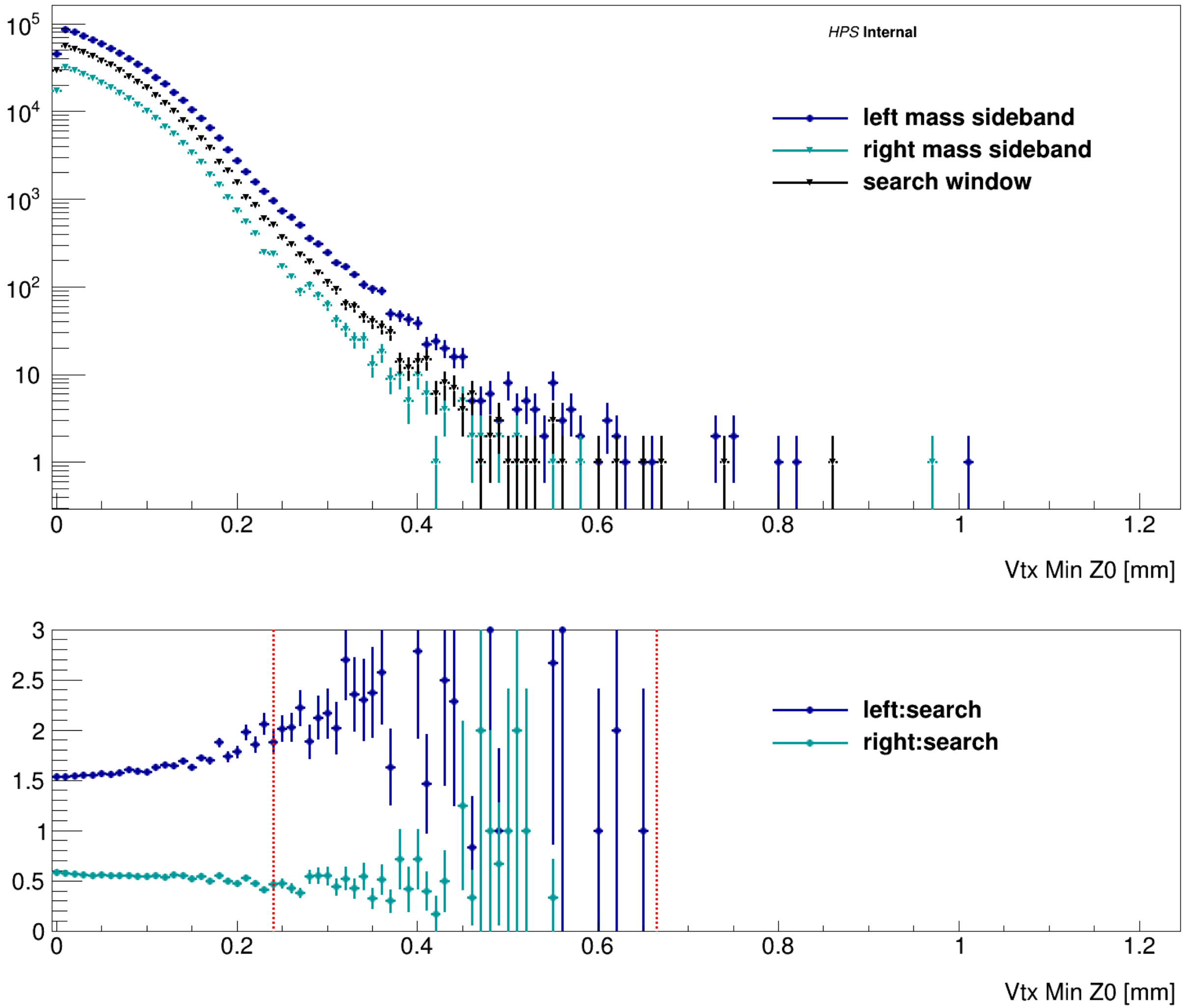
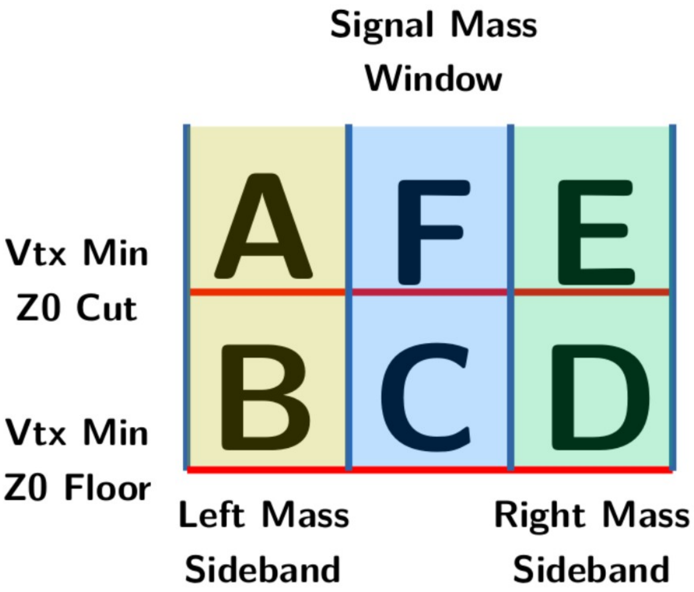
Estimated Background
100% Blinded Data

06/18/2024

Alic Spellman




Signal Mass Window

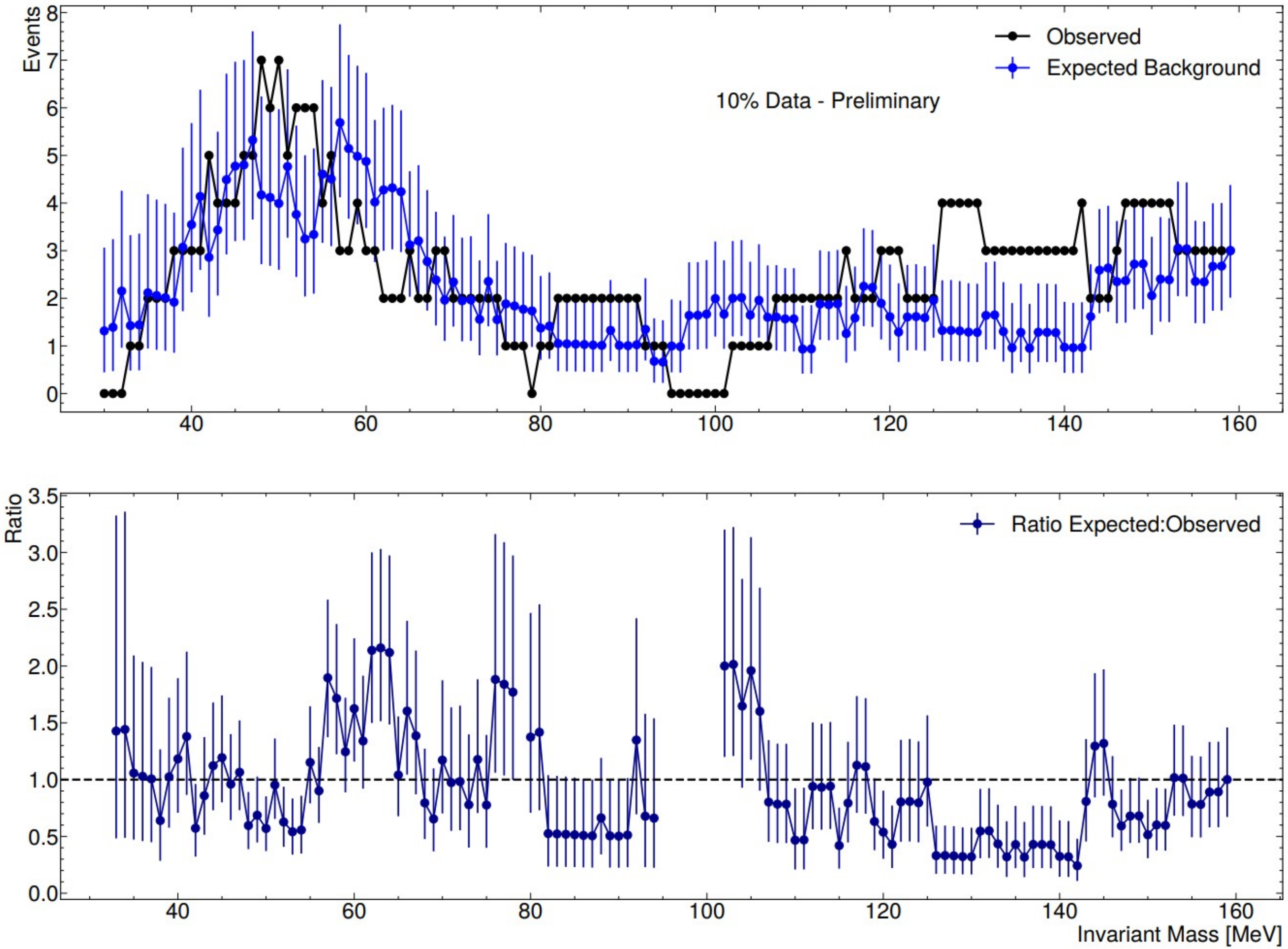
Vtx Min Z0 Cut

Vtx Min Z0 Floor

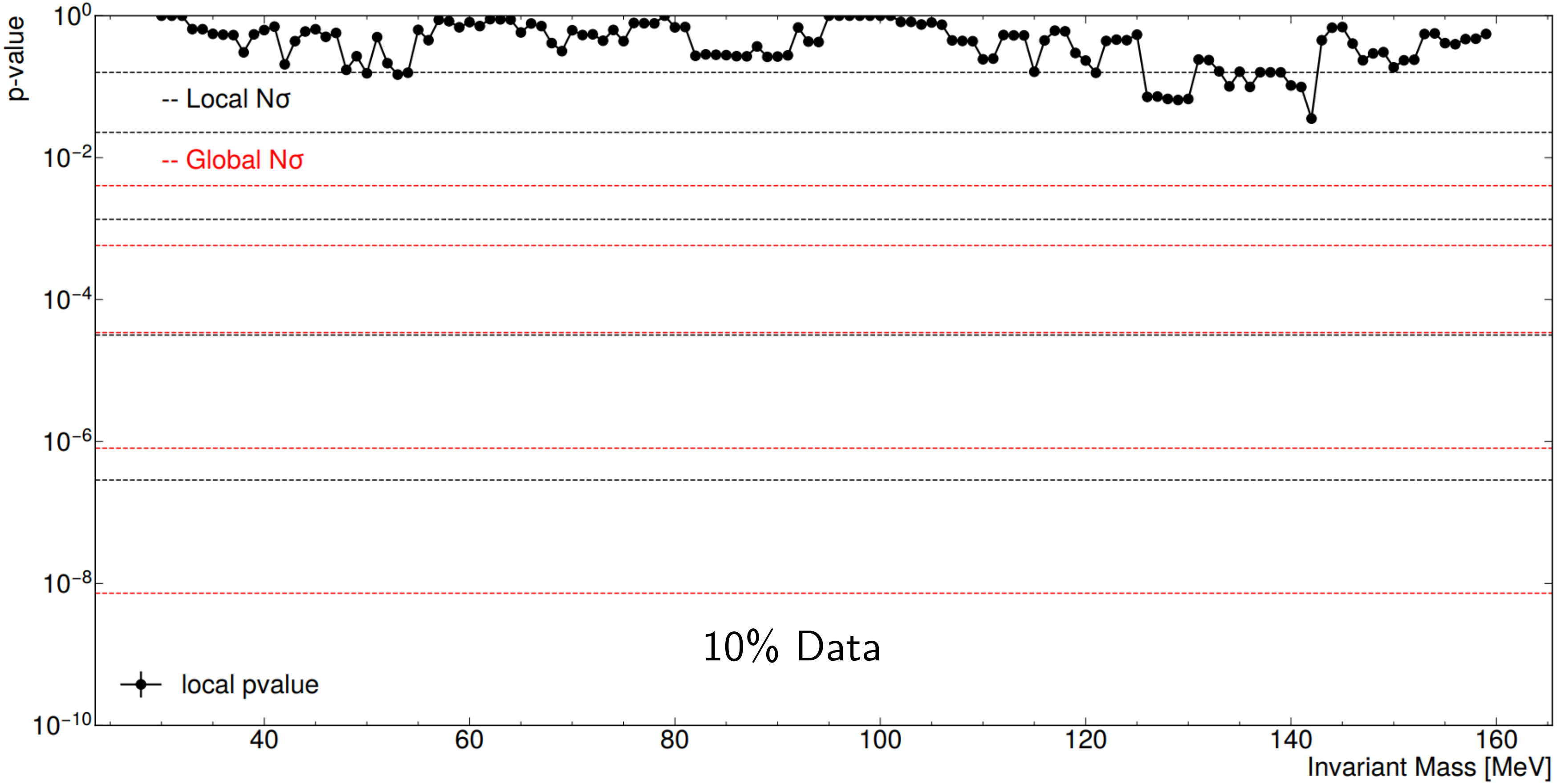
Left Mass Sideband

Right Mass Sideband

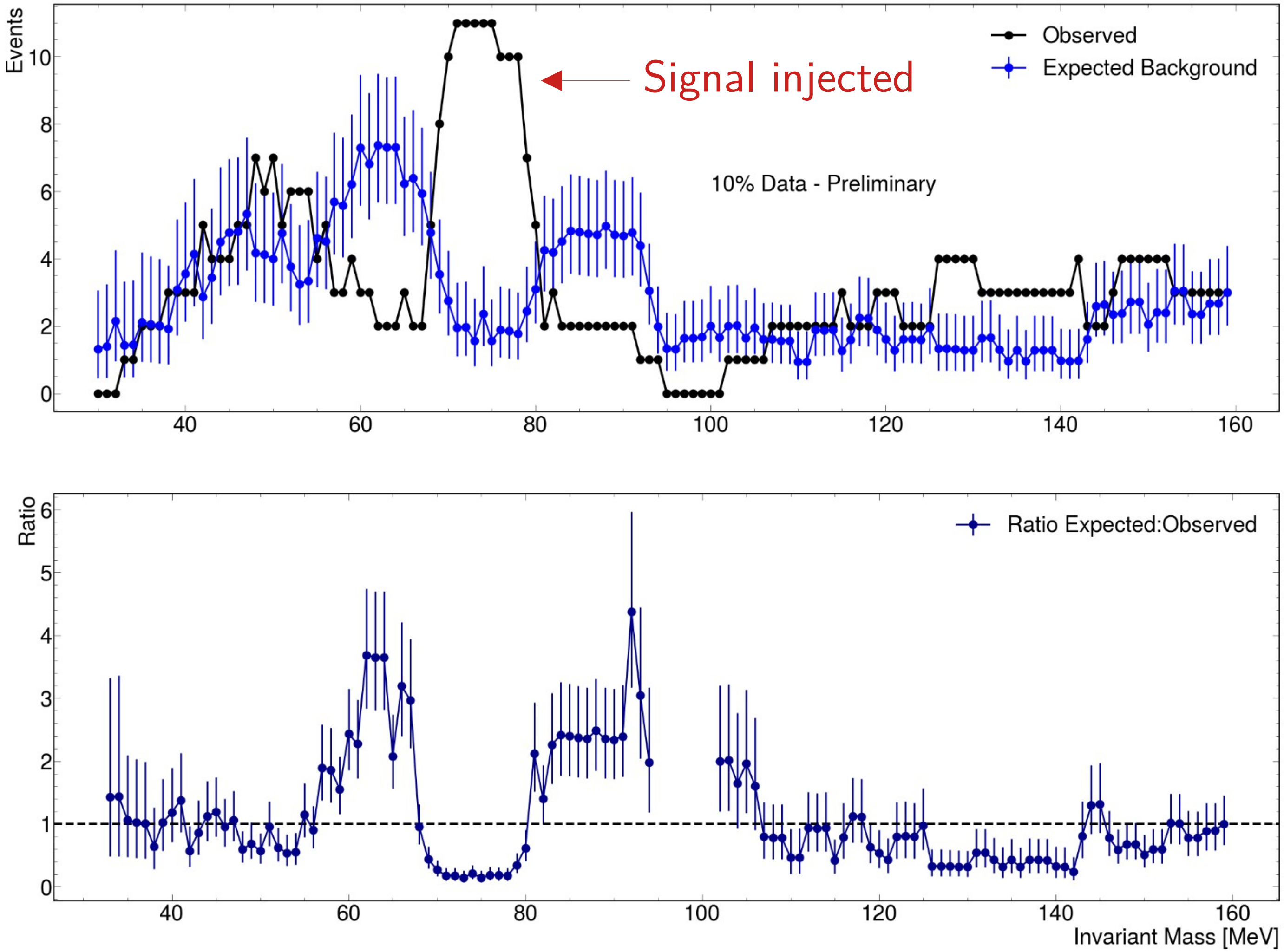
Expected Background - 10% Data



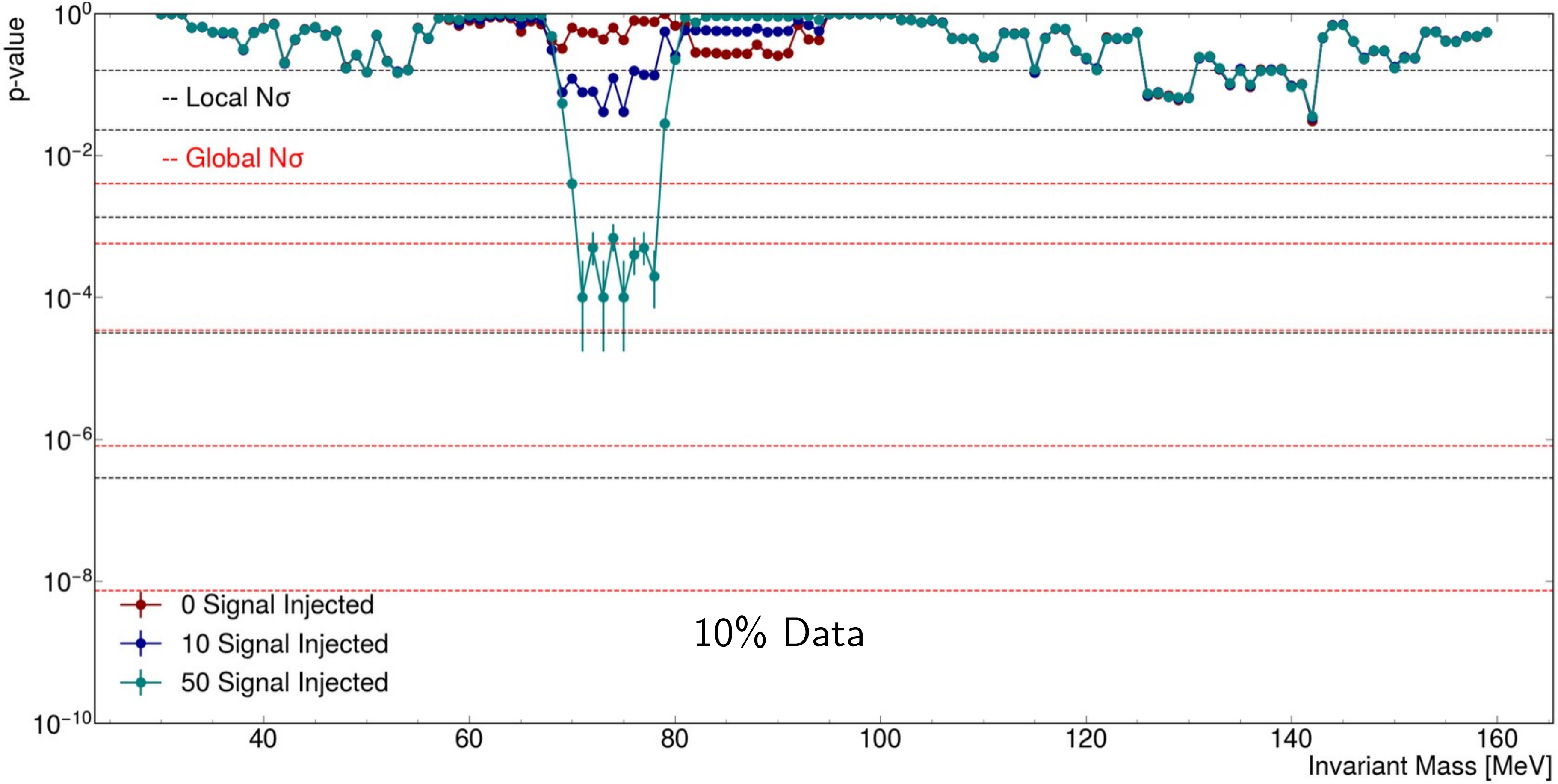
Pvalue – 10% Data



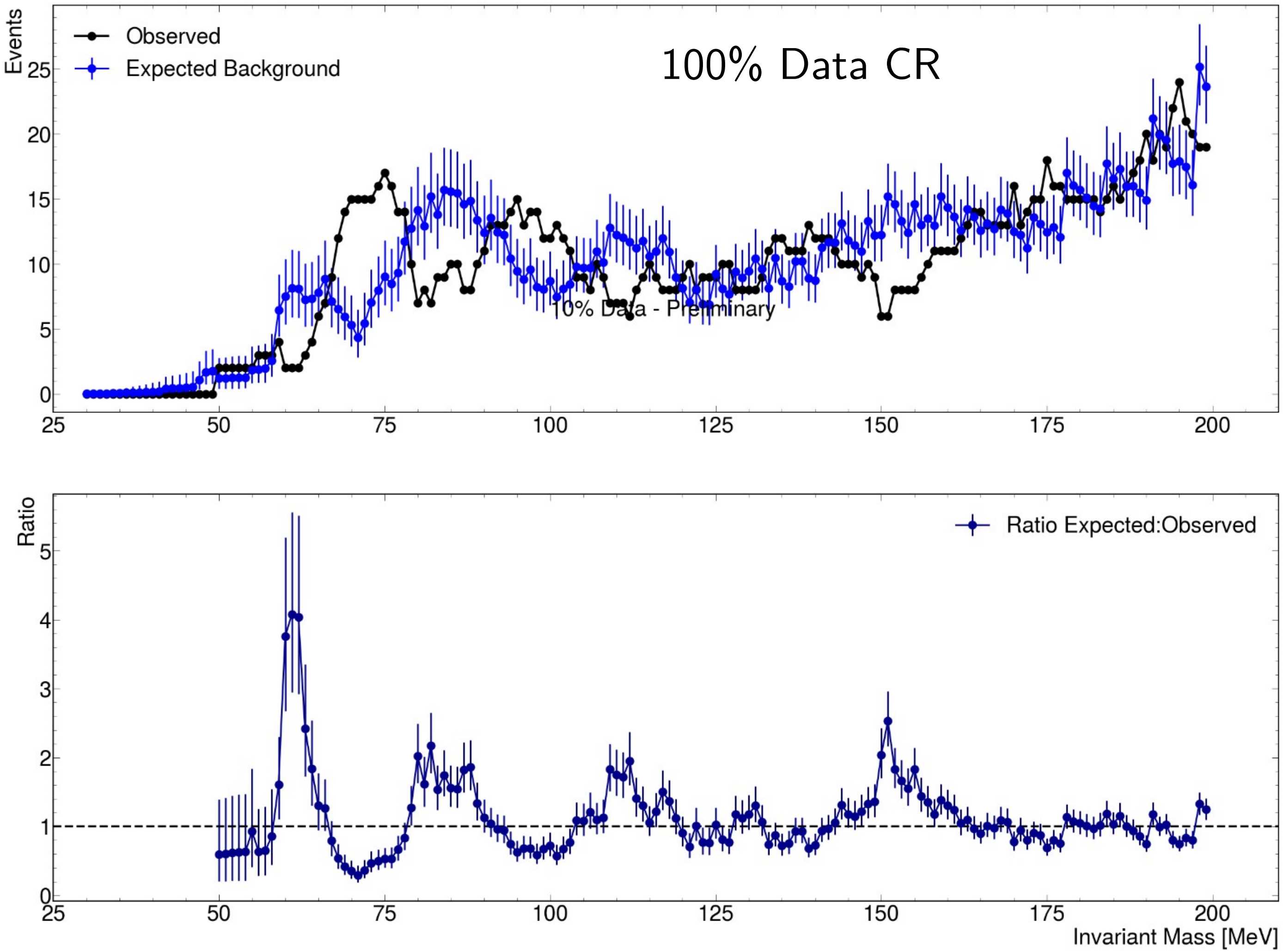
Signal Injected – 10% Data



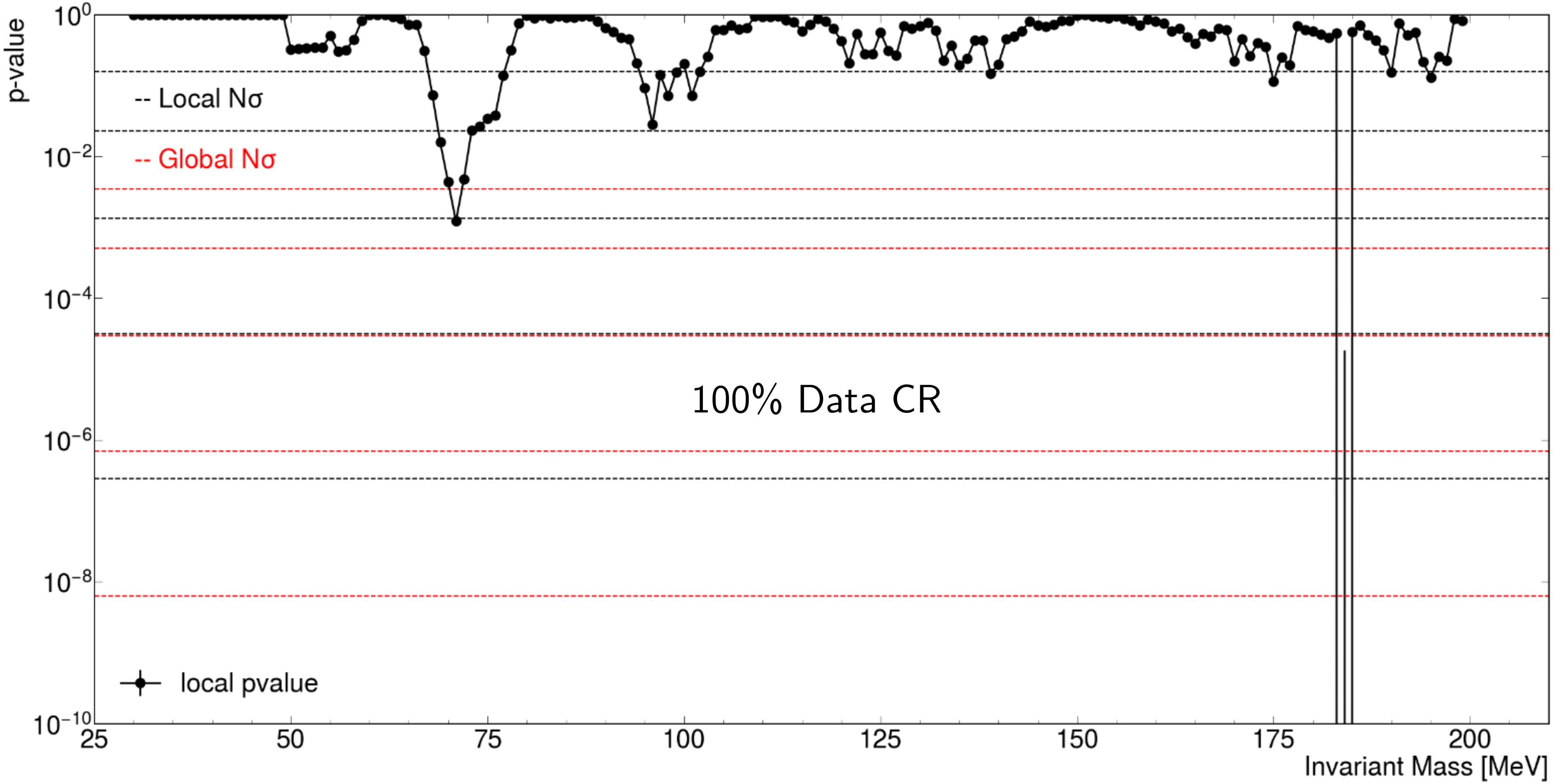
Signal Injected Pvalue – 10% Data



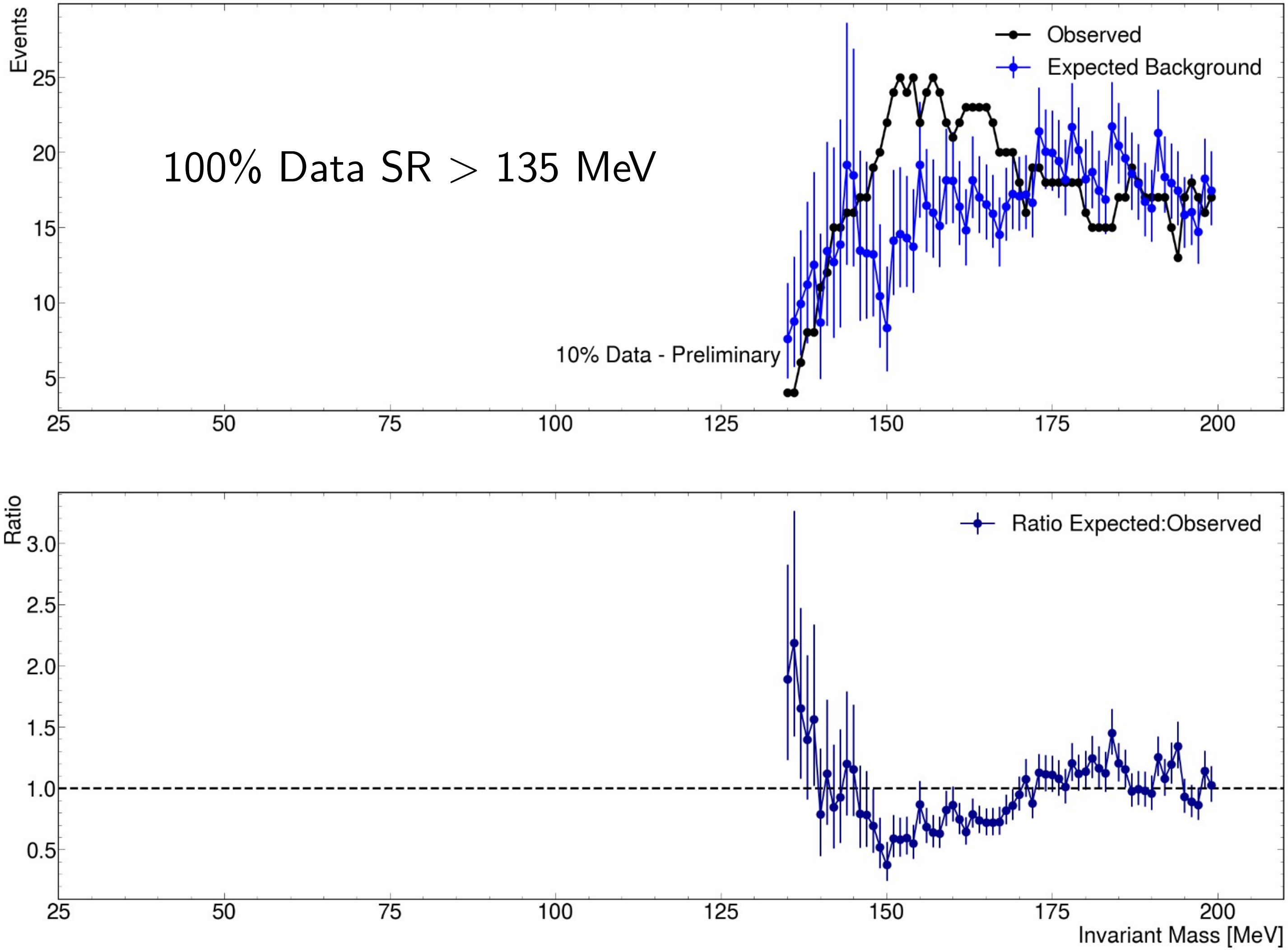
Expected Background – 100% Data CR



Pvalue – 100% Data CR

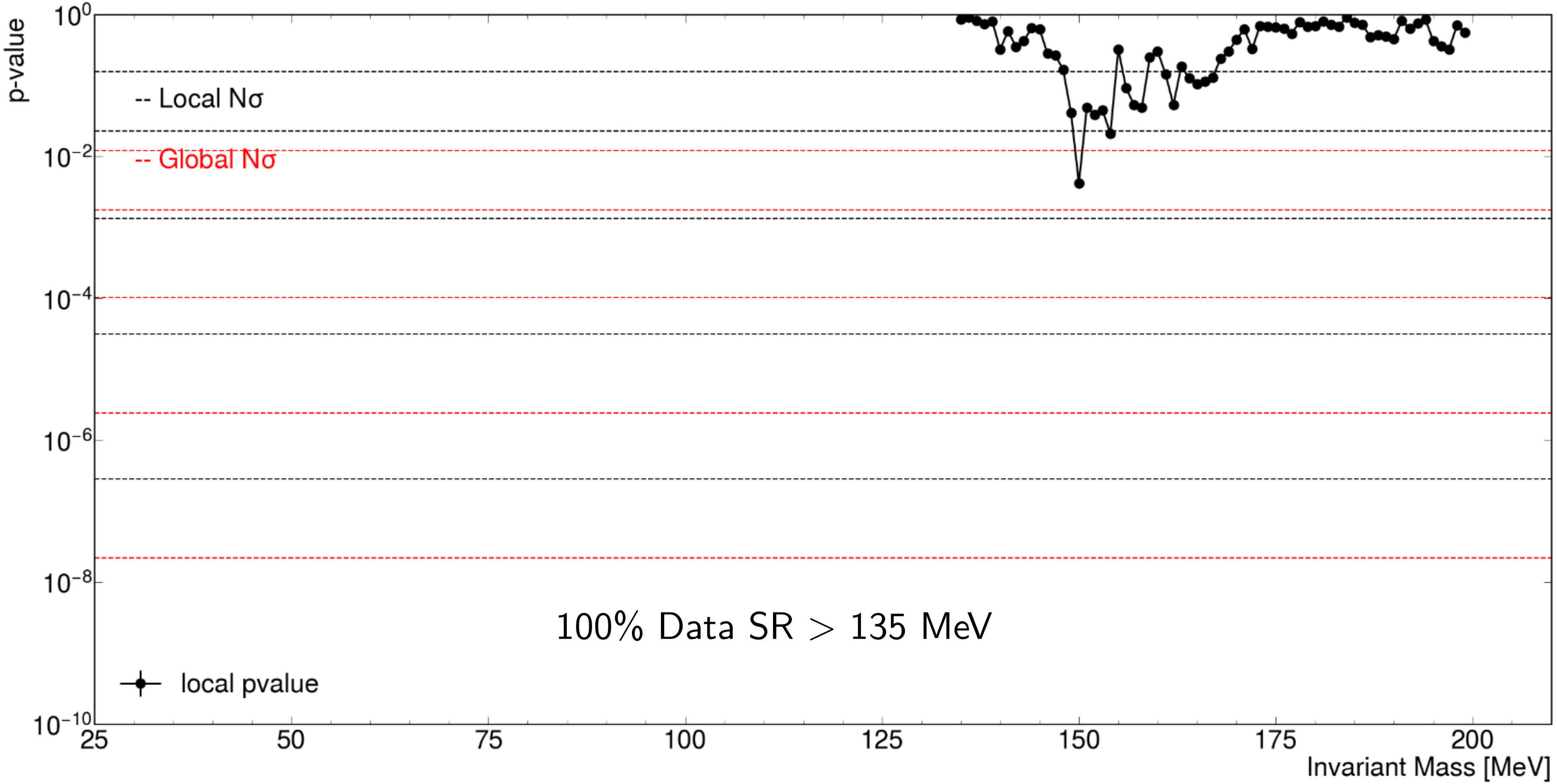


Expected Background – 100% Data SR Blind



100% Data SR $>$ 135 MeV

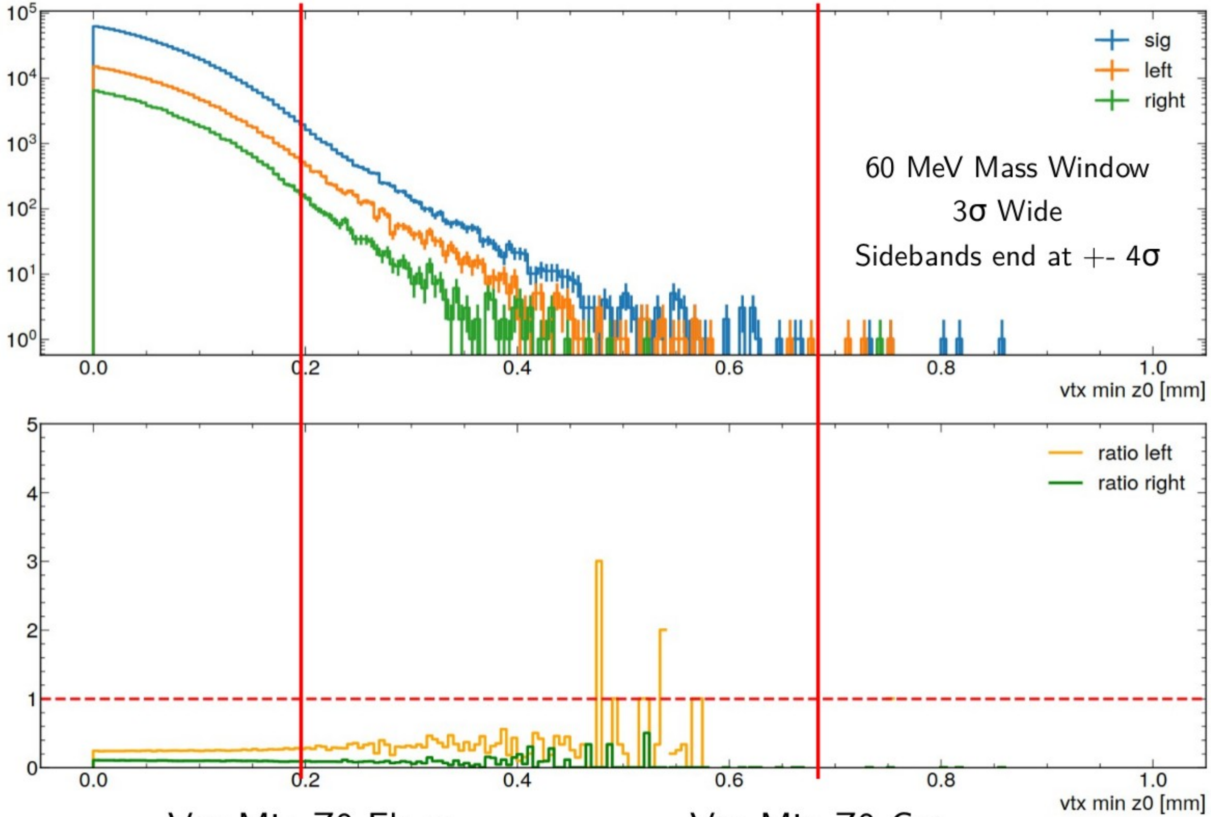
Expected Background – 100% Data SR Blind



100% Data SR > 135 MeV

Backup

Expected Background Estimate – ABCD Method



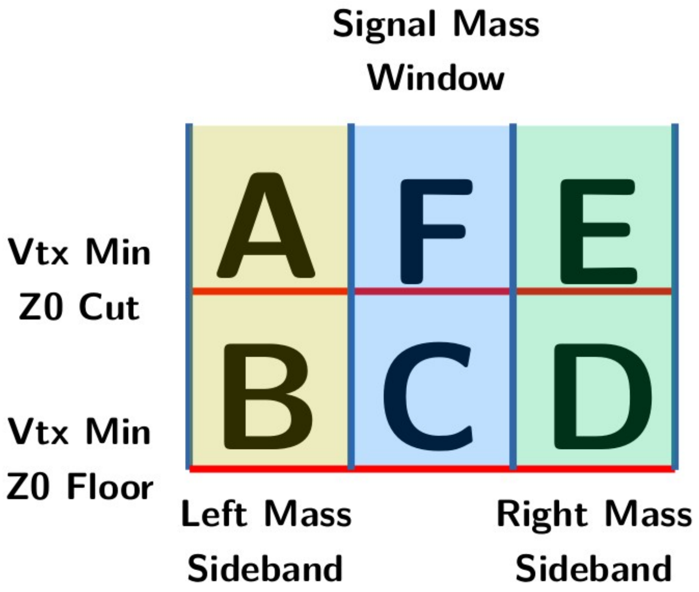
60 MeV Mass Window

3σ Wide

Sidebands end at +- 4σ

Vtx Min Z0 Floor

Vtx Min Z0 Cut



Signal Mass Window

Vtx Min Z0 Cut

Vtx Min Z0 Floor

A F E

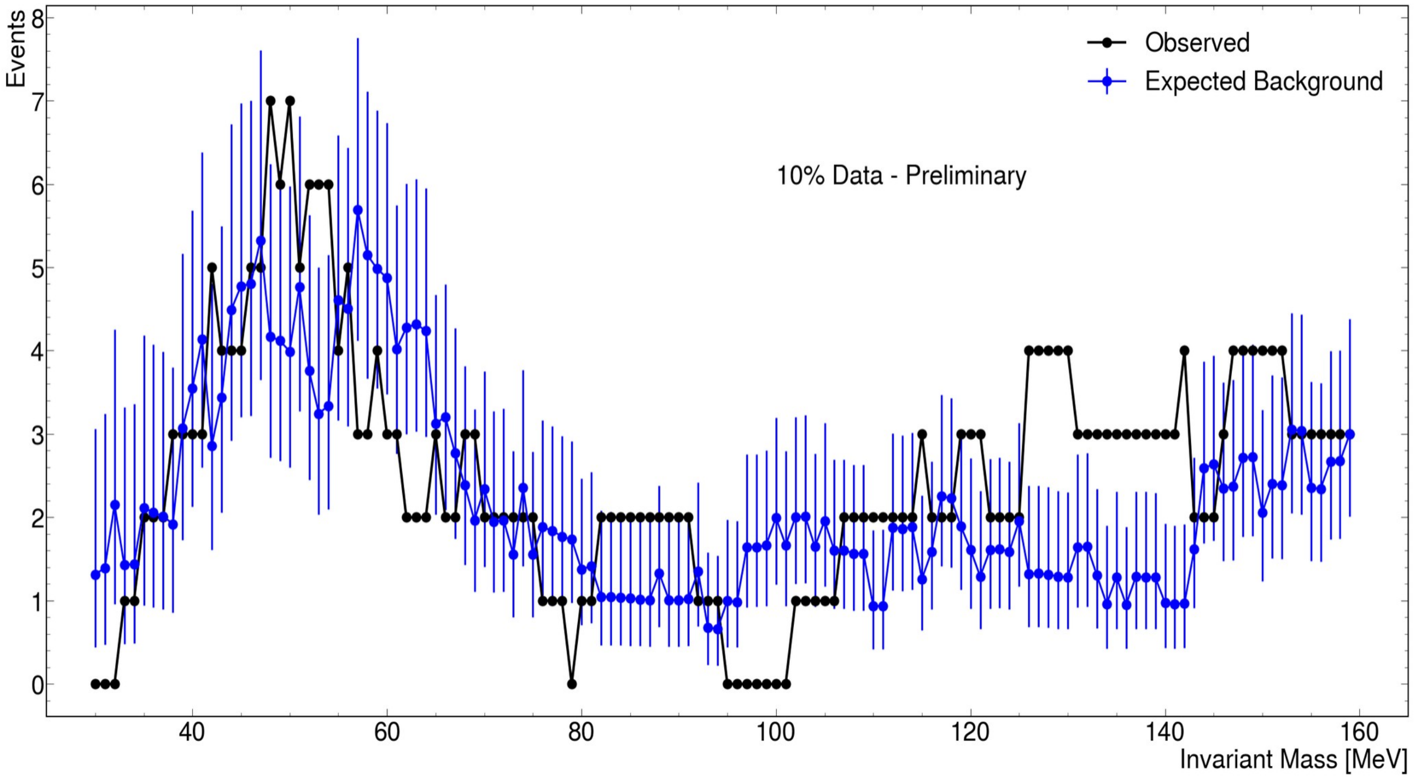
B C D

Left Mass Sideband

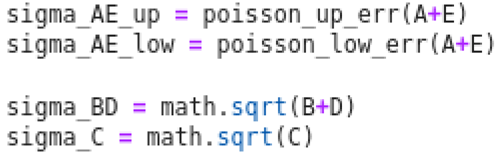
Right Mass Sideband

$$(A+E)/(B+D)*C = F$$

Expected Background Estimate – 10% Data

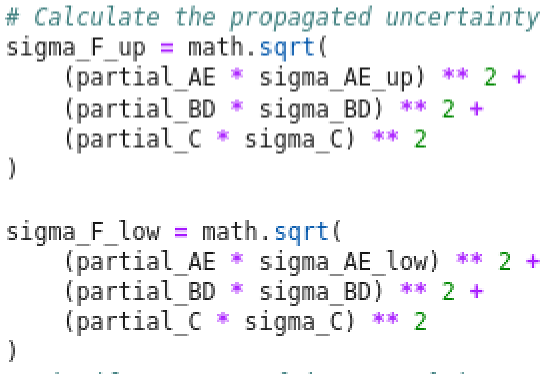


Error on (A+E) and (B+D)



```
sigma_AE_up = poisson_up_err(A+E)
sigma_AE_low = poisson_low_err(A+E)

sigma_BD = math.sqrt(B+D)
sigma_C = math.sqrt(C)
```



```
# Calculate the propagated uncertainty
sigma_F_up = math.sqrt(
    (partial_AE * sigma_AE_up) ** 2 +
    (partial_BD * sigma_BD) ** 2 +
    (partial_C * sigma_C) ** 2
)

sigma_F_low = math.sqrt(
    (partial_AE * sigma_AE_low) ** 2 +
    (partial_BD * sigma_BD) ** 2 +
    (partial_C * sigma_C) ** 2
)
```

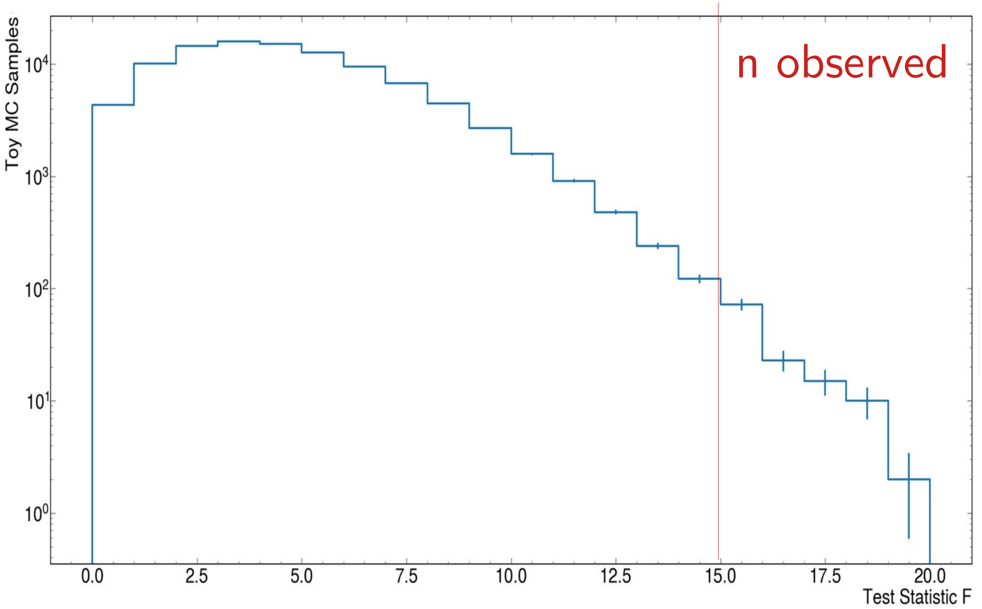
Calculating P-Value using MC Toys

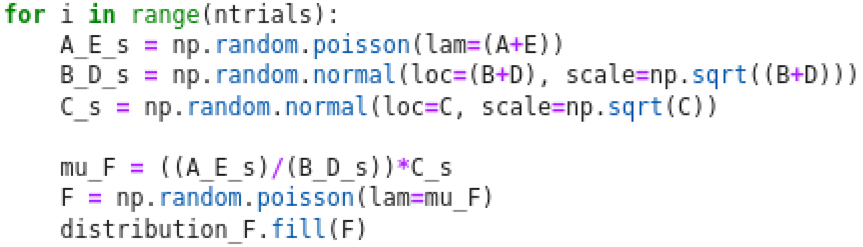
- Run N Toy MC experiments
- Build distribution of test statistic ‘F’ (expected background)

- Three distributions…
  - Poisson with mean=(A+E)
  - Gaus with mean=(B+D), std=sqrt(B+D)
  - Gaussian with mean=C, std=sqrt(C)

- For each experiment, sample the distributions, calculate expected background mean μ_F
- **Test Statistic: sample Poisson with mean = μ_F**

- Get pvalue by integrating (normalized) test statistic distribution from ‘nobs’ → ∞





```
for i in range(ntrials):
    A_E_s = np.random.poisson(lam=(A+E))
    B_D_s = np.random.normal(loc=(B+D), scale=np.sqrt((B+D)))
    C_s = np.random.normal(loc=C, scale=np.sqrt(C))

    mu_F = ((A_E_s)/(B_D_s))*C_s
    F = np.random.poisson(lam=mu_F)
    distribution_F.fill(F)
```

Calculating P-Value using MC Toys

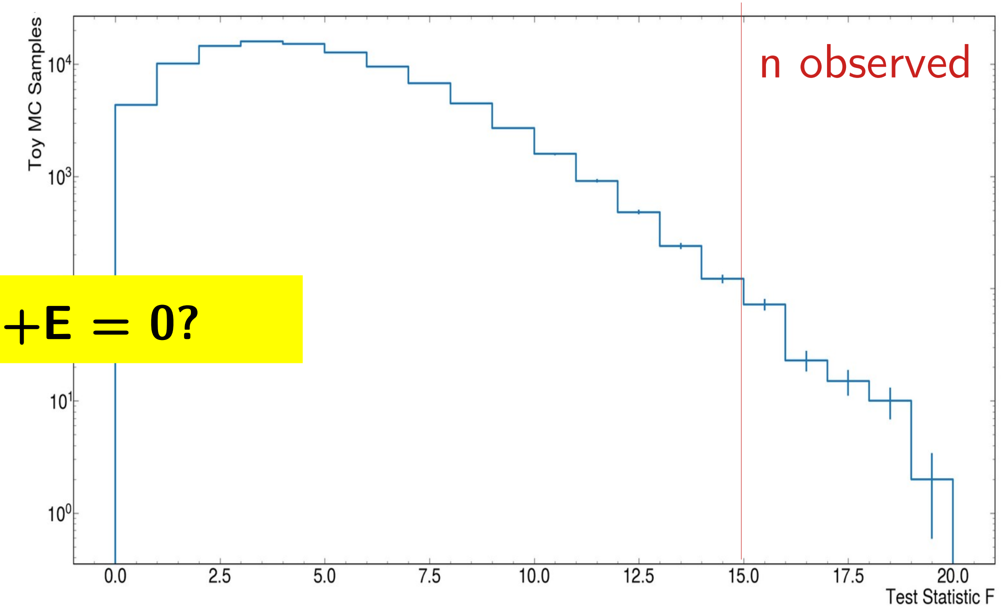
- Run N Toy MC experiments
- Build distribution of test statistic ‘F’ (expected background)

- Three distributions…
  - Poisson with mean=(A+E)
  - Gaus with mean=(B+D), std=sqrt(B+D)
  - Gaussian with mean=C, std=sqrt(C)

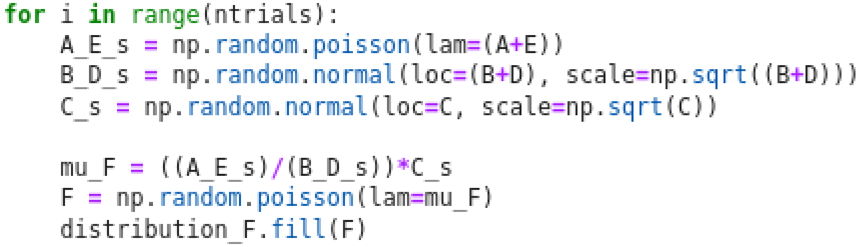
What if A+E = 0?

- For each experiment, sample the distributions, calculate expected background mean μ_F
- **Test Statistic: sample Poisson with mean = μ_F**

- Get pvalue by integrating (normalized) test statistic distribution from ‘nobs’ → ∞



n observed

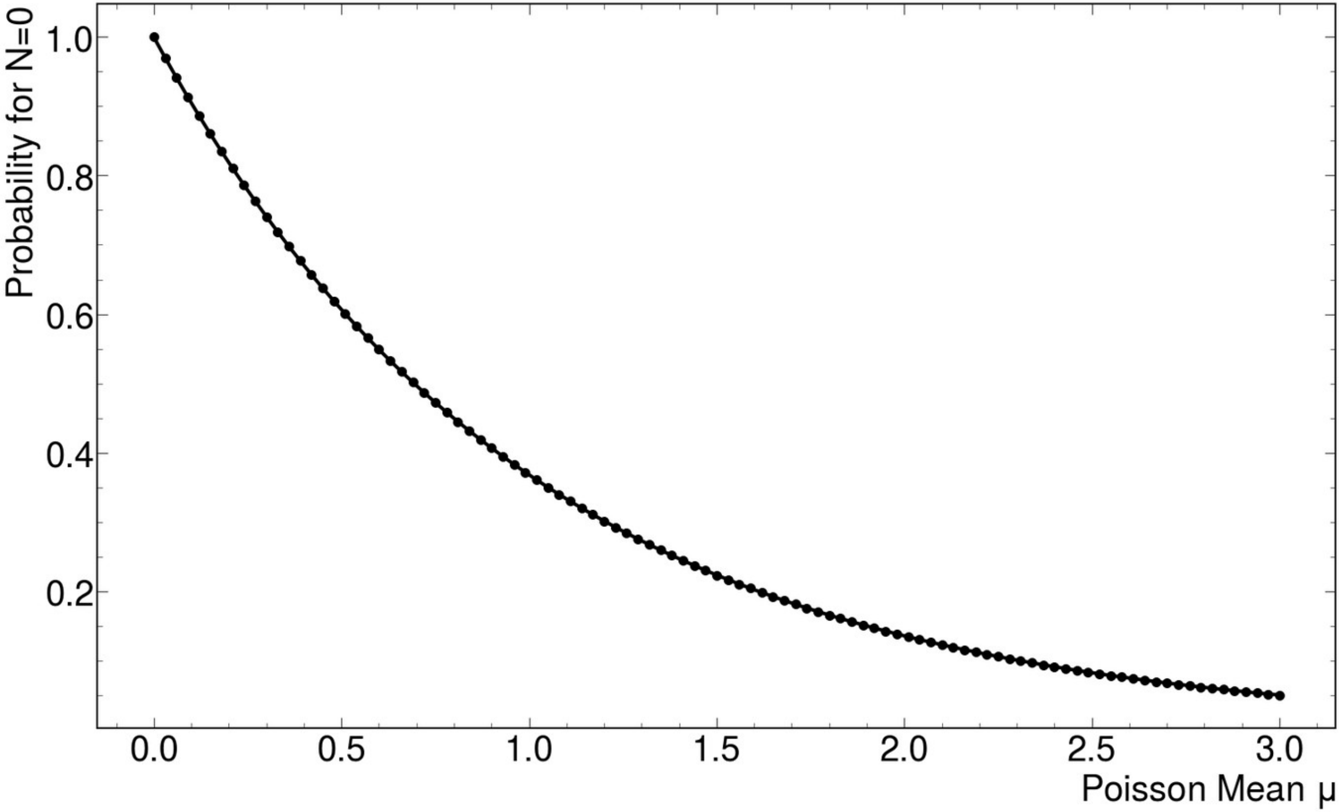


```
for i in range(ntrials):
    A_E_s = np.random.poisson(lam=(A+E))
    B_D_s = np.random.normal(loc=(B+D), scale=np.sqrt((B+D)))
    C_s = np.random.normal(loc=C, scale=np.sqrt(C))

    mu_F = ((A_E_s)/(B_D_s))*C_s
    F = np.random.poisson(lam=mu_F)
    distribution_F.fill(F)
```

Error when A+E = 0?

- If A+E = 0, we can't build a Poisson distribution for the toys
- We could just force A+E = 1, but that's very conservative



Not actually applicable under current min z0 cut