Sınır hesaplama ve istatistik

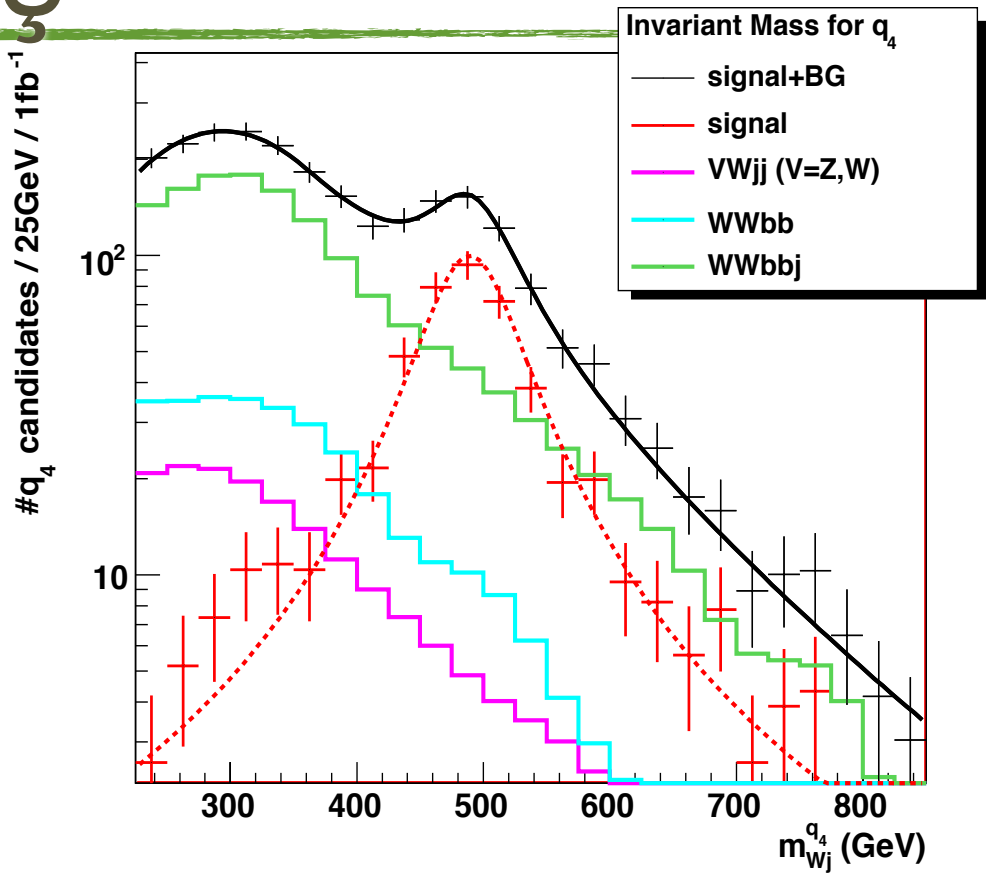
HPFBU -2012

Gökhan Ünel - UCI

11-19 Şubat 2012

Giriş

- LHC çalışmadan önce ve ilk veriler toplanırken keşif yapıvermeyi hayal ediyorduk.
  - ➡ sinyal olayları arkaplan üzerine rahatça çıkacaktı
- Bu durumda
  - ➡ elimizdeki tüm veriye bir eğri yakıştıracak
    - kenarlardan arkaplanı
    - farklılık gösteren yerden sinyali
  - ➡ Alıp, olay sayılarını hesaplayacaktık
    - sinyal tepesinden 2 hatapayı sol-sağ arası integral alarak
    - ölçümün istatistik anlamlılığını bulacaktık.
    - 2σ:sınır 3σ:sinyal 5σ: keşif

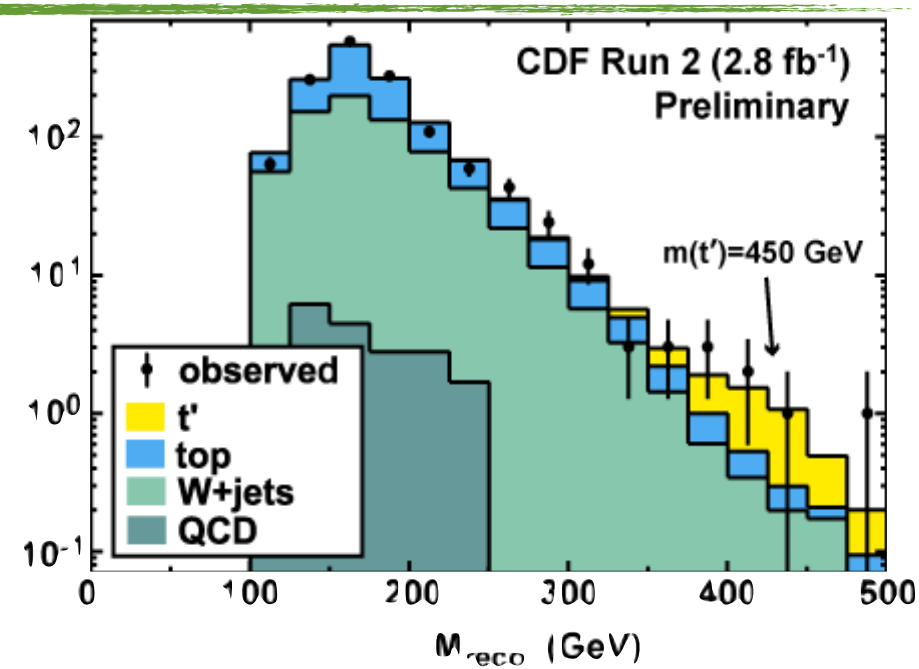


$$\mathcal{S} = \sqrt{2 \times [(s+b)\ln(1+\frac{s}{b}) - s]}$$

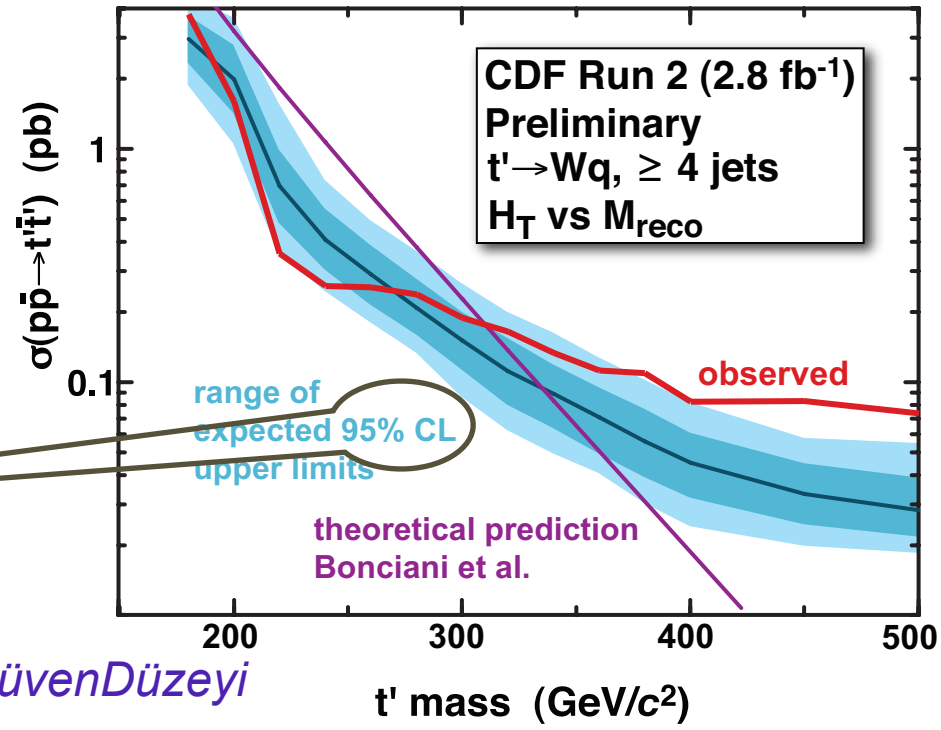
ama böyle olmadı

Karşılaştığımız durum

- LHC (daha önce de Tevatron)
  - ➡ kenar bölgesi SM ile uyumlu
  - ➡ sinyal bölgesinde çok az olay var
    - ölçüm hassasiyetimiz az
- başka bir yöntem lazım
  - ➡ daha çok MC'ya dayanarak
- Bu arada
  - ➡ eğer aradığımız değişken gauss gibi dağılım gösteriyorsa:

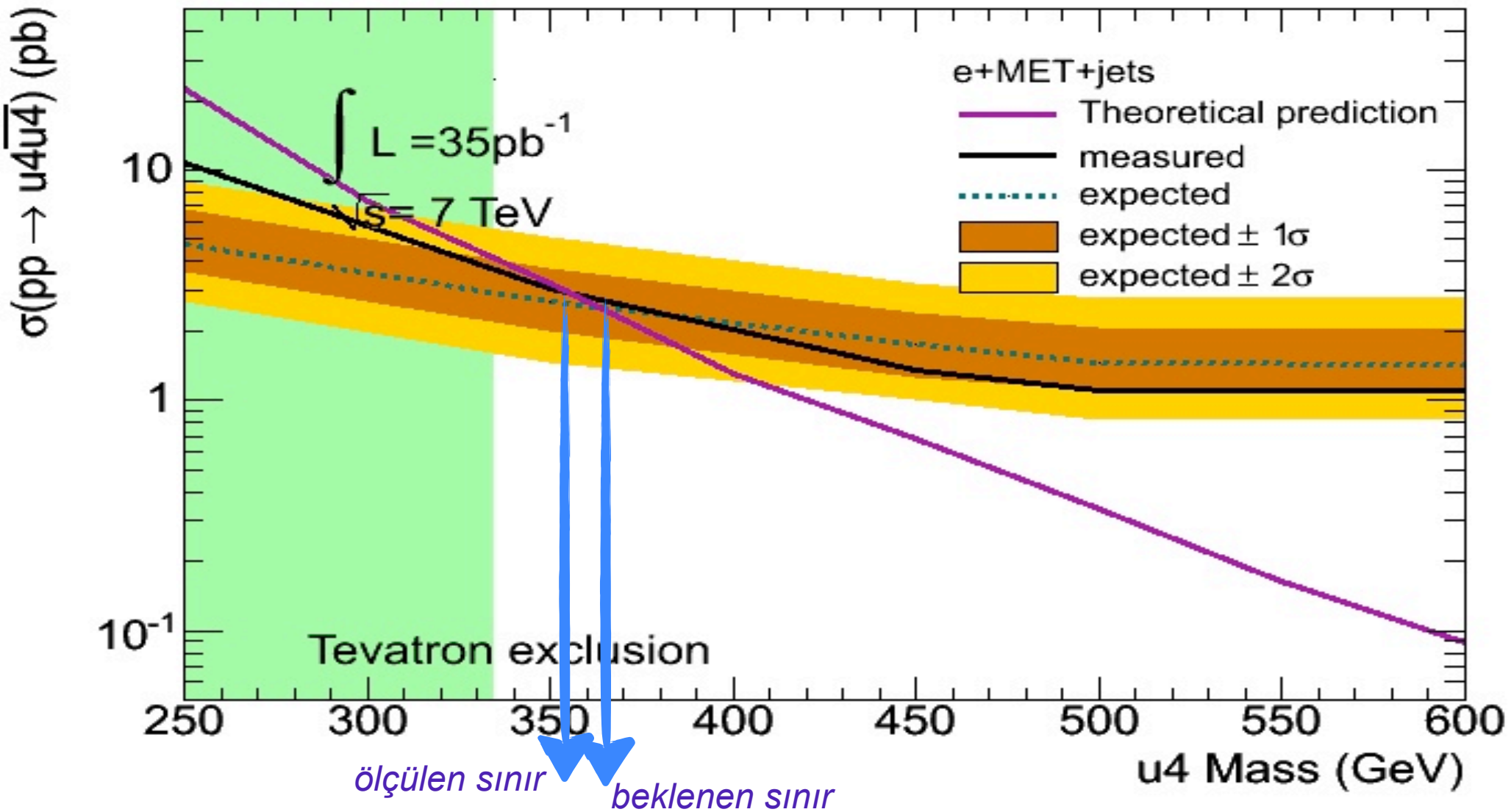


α	δ	α	δ
0.3173	1σ	0.2	1.28σ
4.55×10^{-2}	2σ	0.1	1.64σ
2.7×10^{-3}	3σ	0.05	1.96σ
6.3×10^{-5}	4σ	0.01	2.58σ
5.7×10^{-7}	5σ		



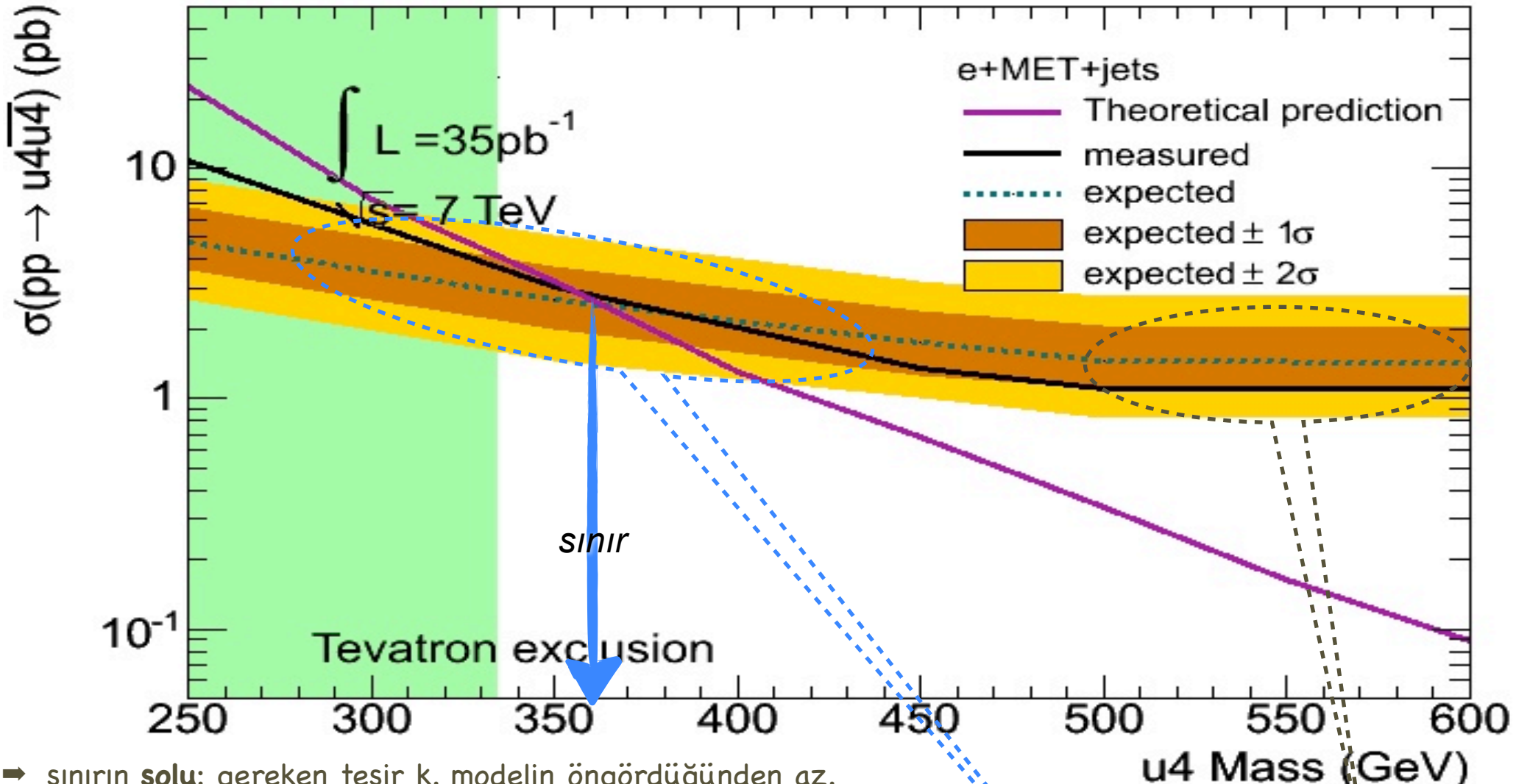
CL : ConfidenceLevel - GüvenDüzeyi
α : yanılma olasılığı

Çizime yakından bakalım



- theory: modelin bize verdiği tesir kesiti
- expected: MC'dan bulunan, 95% CL için gereken tesir kesiti
  - expected 1(2) : MC dağılımlarının 1(2) hatapayı (σ) değiştiği durum için bulunan, 95%CL için gereken tesir k.
- measured (observed) : deney verisi MC ile karşılaştırılarak bulunan 95%CL için gereken tesir k.

Çizime yakından bakalım

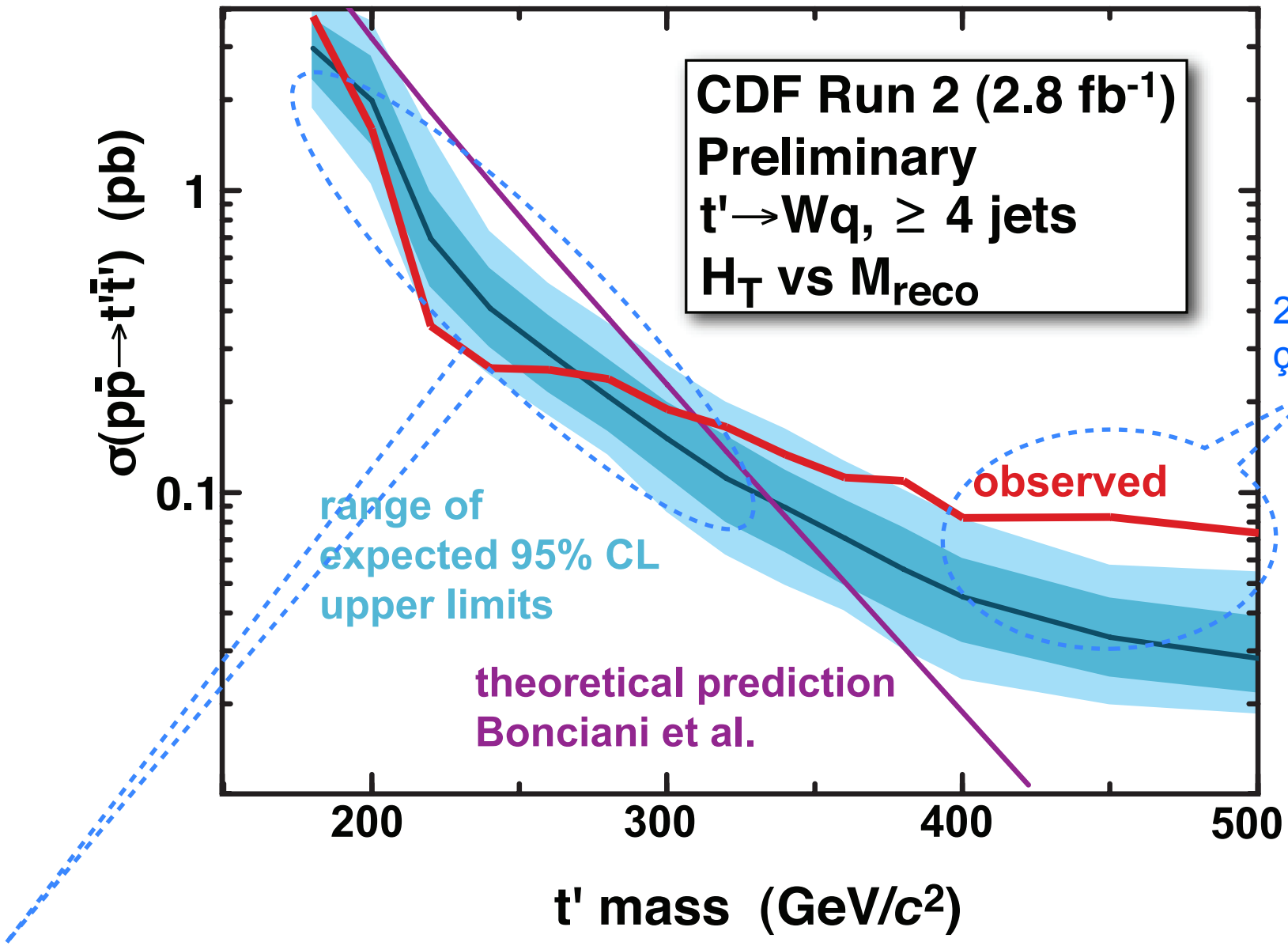


- sınırın **solu**: gereken tesir k. modelin öngördüğünden az. modelim bana gerekli olay sayısını temin edebilir, **ölçüm yapabilirim**
- sınırın **sağı**: gereken tesir k. modelin öngördüğünden çok. modelim bana yeteri kadar olay veremiyor, yapacağım ölçümler sınır koymak için **yetersiz**

Burada bir tuhaflık var. çok düz

Burası iyi: gözlem beklenti civarında oynuyor, 2hatapayı içinde kalıyor.

konuşalım



2hatapayından çok fazlalık var.

İyi: gözlem beklenti civarında oynuyor, 2hatapayı içinde kalıyor.

Kullanılan yöntemler

- girdiler - sonuçlar
  - ➡ kağıt kalem --> kuramsal tesir k.
  - ➡ sinyal MC, ardalan MC --> beklenen tesir k.
  - ➡ sinyal MC, ardalan MC, veri --> gözlenen tesir k.

- Soru
  - ➡ "veri" ve "eğreti-veri" (pseudodata) hangi durum ile uyumlu?
    - Null Hypothesis = Yok Sanrısı
    - Alt Hypothesis = Diğer Sanrı

- Ölçüm
  - ➡ ilgilendiğim değişkenin bir çok değeri için uyum testleri yap

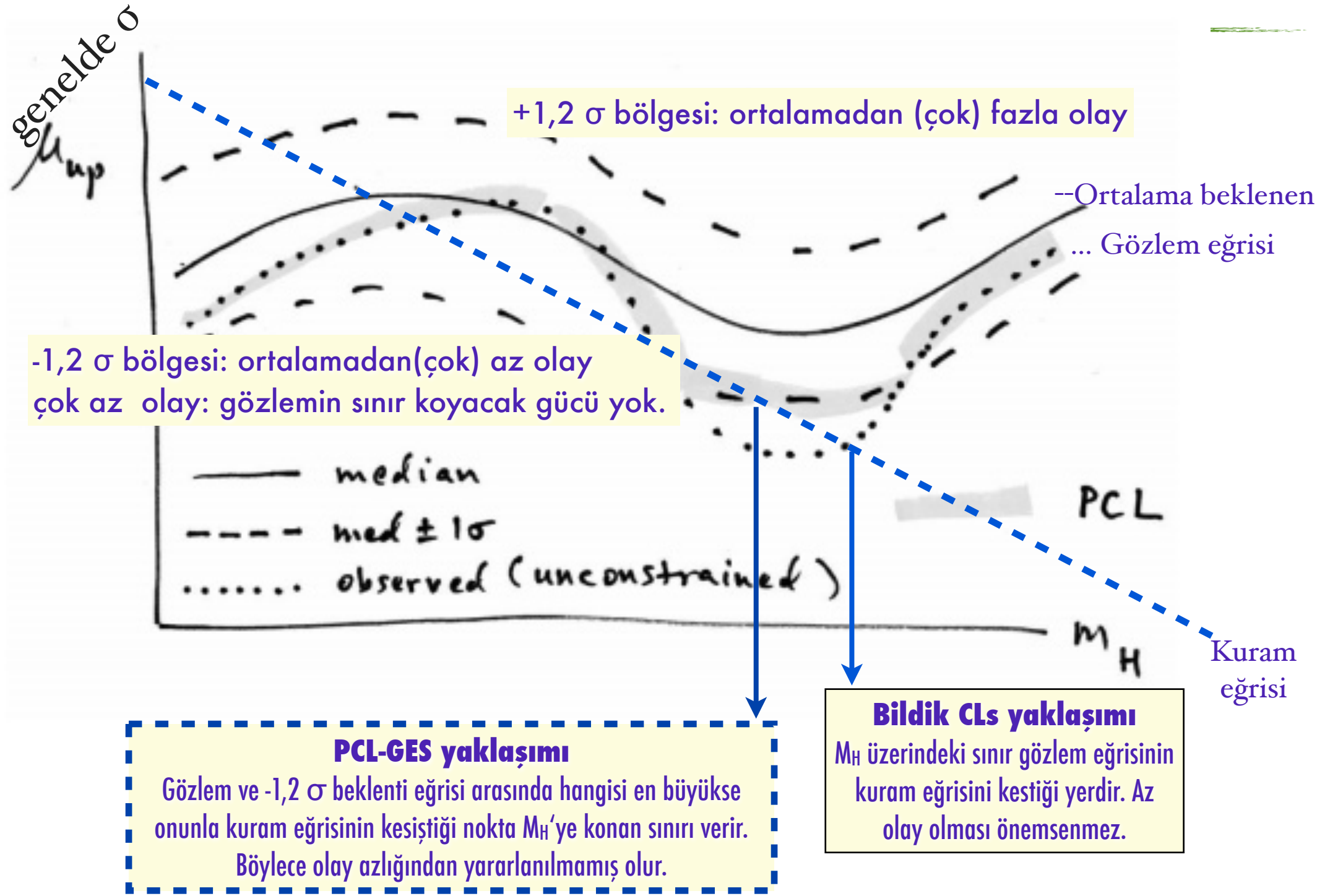
- Kullanılan kıstaslar
  - ➡ p-değeri: olasılık değeri, sanrı testi sonucu, CL (Güven Düzeyi)
  - ➡ CL_b : 1-p: genel ardalan uyumluluğu, CL sadece ardalan sanrısı için
  - ➡ CL_{s+b} : CL signal + ardalan sanrısı için, hırslı keşif yapmak istersek.
  - ➡ CL_s : CLsb/CLb. Ardalan oynamalarına karşı sarsılmaz (stable)
    - yani daha "normal" bir dağılımdır.

$$CL_s = \frac{p\text{-value of signal plus background hypothesis}}{1 - p\text{-value of hypothesis of background only}}$$

kullanılan hatalar veya bilinemeyenler

- İstatistik hatalar
- Systematik-düzenli hatalar veya bilinemeyenler (uncentainties)
  - ➡ Rate = Sıklık
    - genel şekil aynı,
    - her kutu yeni katsayı ile yukarı veya aşağı dalgalanabilir.
  - ➡ Shape = Şekil
    - genel şekil değişebilir,
    - her kutu farklı katsayı ile yukarı veya aşağı dalgalanabilir.

Gücü Engellenmiş Sınır (power constrained limit) yaklaşımı



PCL-GES yaklaşımı

Gözlem ve -1,2 σ beklenti eğrisi arasında hangisi en büyükse onunla kuram eğrisinin kesiştiği nokta M_H'ye konan sınırı verir. Böylece olay azlığından yararlanılmamış olur.

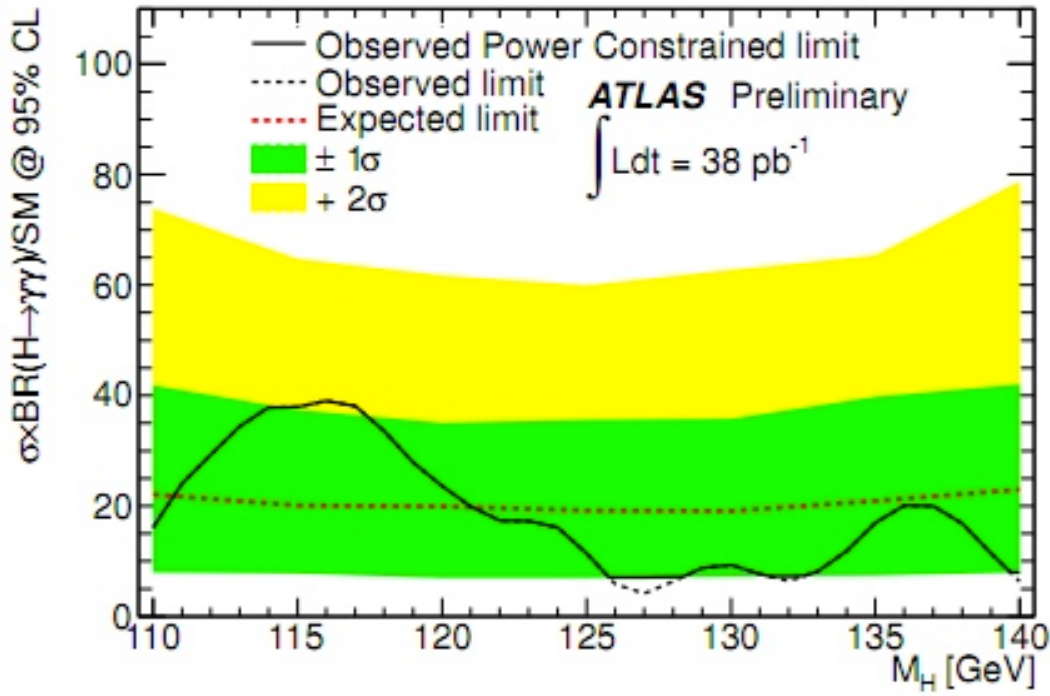
Bildik CLs yaklaşımı

M_H üzerindeki sınır gözlem eğrisinin kuram eğrisini kestiği yerdir. Az olay olması önemsenmez.

Örnekler ve notlar

m_H [GeV]	Expected limit		Observed limit	
	CL_{s+b}	CL_s	PCL_{s+b}	CL_s
110	22.1	27.9	16.1	24.0
115	20.1	25.5	37.9	39.7
120	19.9	24.4	23.6	27.2
130	19.1	24.0	9.3	18.3
140	23.0	29.9	8.0	20.7

PCL_{s+b} means the CL_{s+b} method with Power Constrained Limit technique for the computation of the observed limit.



- PCL hem CLs hem de CLsb için kullanılabilir
  - ➡ genelde CLsb'nin gereksiz iyimserliğini gidermek için kullanılır.
- PCL Kaynakları
  - ➡ http://cdsweb.cern.ch/record/1336758/files/ATLAS-CONF-2011-025.pdf
  - ➡ http://arxiv.org/pdf/1105.3166v1

işleri bilgisayarla yapalım

- En çok kullanılan sınır belirleme gereçleri:
  - **mclimit** : FNAL ve yakın zamanda ATLAS çözümlemelerinde kullanıldı ama ROOT kütüphanesinin parçası değil
  - **TLimit** : mclimit'in ROOT'a uyarlanmış hali ama artık yenilenmiyor ve bin-by-bin (şekil) sistematik hataları işlemiyor.
  - **RooStats** : ATLAS tarafından öneriliyor, ama TLimit'e oranla çok yavaş ve kullanıcı dostu değil. Şekil sistematikleri ekleme olarak histfactory alt bohçası içinde geliyor.
  - **fastpcl / fastcls** : UC Irvine grubunun yazılımı, şekil sistematik hatalarını işliyor ama ROOT (RooStats) kütüphanesinin parçası değil.
  - **RSLimit** : 

ayrıntılar sonra

Kütüphanelerin en basiti: TLimit

```
// now work the same numbers with TLimit class
  TLimitDataSource* mydatasource = new TLimitDataSource(sih,bgh,dth);
  TConfidenceLevel *myconfidence = TLimit::ComputeLimit(mydatasource,90000);
if (clsclsb) {
  double cls_tl=myconfidence->CLsb();
  double cls_tl_exp0= myconfidence->GetExpectedCLsb_b();
  double cls_tl_exp_m1s=myconfidence->GetExpectedCLsb_b(-1);
  double cls_tl_exp_p1s=myconfidence->GetExpectedCLsb_b(+1);
  double cls_tl_exp_m2s=myconfidence->GetExpectedCLsb_b(-2);
  double cls_tl_exp_p2s=myconfidence->GetExpectedCLsb_b(+2);
} else {
  double cls_tl=myconfidence->CLs();
  double cls_tl_exp0= myconfidence->GetExpectedCLs_b();
  double cls_tl_exp_m1s=myconfidence->GetExpectedCLs_b(-1);
  double cls_tl_exp_p1s=myconfidence->GetExpectedCLs_b(+1);
  double cls_tl_exp_m2s=myconfidence->GetExpectedCLs_b(-2);
  double cls_tl_exp_p2s=myconfidence->GetExpectedCLs_b(+2);
}
```

- tesir kesitine göre hazırlanmış sinyal, ardalan, ve veri histogramlarını alıp geriye CL=GüvenDüzeyi verir
  - ➡ CLs ve CLsb kullanılabilir.

ters sorun

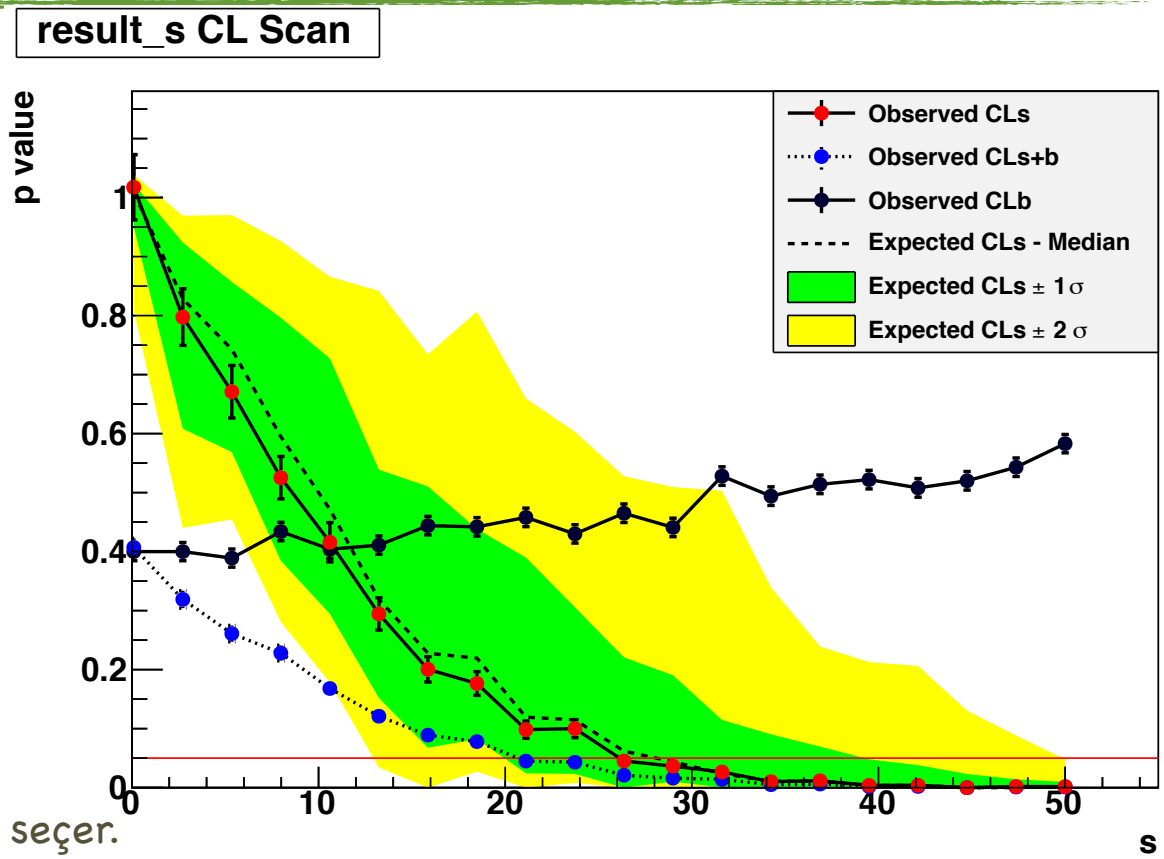
- durum
  - ➡ si ve bg histogramlarım : sih ve bgh olsun,
  - ➡ si tesir kesitim : x0 olursa
  - ➡ aldığım CL değerleri, CL0 olur.
- soru:
  - ➡ CL = 95% durumunu bana verecek si tesir kesiti nedir?
    - Aslında 0.05e bakılır.
    - buna `ters sorun' adını verdim
- olası cevap ve yöntem
  - ➡ kuramsal tesir kesitine x0 diyelim
  - ➡ CL=0.05 verecek olan tesir kesiti s*x0 olsun
    - s=1 den baslayip s'i arttıyım veya azaltayım
    - her adimda CL degerini tekrar hesap edeyim, 0.05`den farkına bakayim
    - taa ki fark sıfira inene kadar.

TÜRK MALI RSLimit yazılımı

- RooStats kütüphanesi etrafına bir kaplama(wrapper)
  - ➡ RooStat, RooFit komutlarını öğrenmekten bizi kurtarır.
    - "RooRealVar" gibi değişkenleri, PROD ve prod arasındaki farkları bilmeye gere.k yok..
  - ➡ HybridCalculator'u kullanır
    - modified frequentist yaklaşımı denir, ayrıntılar için bakınız: http://root.cern.ch/root/html/RooStats__HybridCalculatorOriginal.html
  - ➡ Basit kullanım
  - ➡ TLimit ve mclimit ile uyumlu sonuçlar verir.
- Gerektirir:
  - ➡ C++
  - ➡ Root 5.30/01 (RooFit ile derlenmiş olmalı) veya daha ileri
    - eski ROOT sürümleri farklı API kullanırdı.
- Sunar:
  - ➡ Gözlenen ve Beklenen GüvenDüzeyleri & üzerlerindeki hatalar
  - ➡ K-çarpanları for Gözlenen and Beklenen sınırlar (ters sorun)
  - ➡ Rate systematics kullanılabilir, Şekil systematics yakında

Kullanıcı seçenekleri

- CLs veya CLs+b olası (PCL de)
- Bir çok istatistik deneme olanağı
  1. Simple Likelihood Ratio,
  2. Ratio Of Profiled Likelihoods
  3. Profile Likelihood
  4. Number of Events
  - more can be easily added
- Sanrı testi - ters sorun çözümü
  - ➡ GüvenDüzeyini kullanıcı belirler (açılışta= 95%)
  - ➡ Sinyal tesir kesiti taraması
    - Adım boyunu ve tarama bölgesini kullanıcı seçer.
  - ➡ CLs, CLs+b, CLb çizimleri yapılır, pdf çıktı
- Discriminator histogram range setting
- ProofLite ile çok çekirdek kullanılabilir
- sinyal ve ardalan için oyuncakMC olay sayısını kullanıcı belirler.




```
Comparing results from TLimit vs RooSta
CLsb Obs TL: 0.4548   RSL:0.442
CLsb Exp TL: 0.1457   RSL:0.158

CLsb -1s TL: 0.4656   RSL:0.477
CLsb +1s TL: 0.02137   RSL:0.023

CLsb -2s TL: 0.7994   RSL:0.828
CLsb +2s TL: 0.001507   RSL:0.001
```

En basit kullanım

```
TFile  *bb = new TFile("examplefit.root");
 TH1F *sih = (TH1F*)((TH1F*)bb->Get("signal"))->Clone();
 TH1F *bgh = (TH1F*)((TH1F*)bb->Get("ttbar"))->Clone();
 TH1F *dth = (TH1F*)((TH1F*)bb->Get("data"))->Clone();
```

Adım1 :girdi histogramlarını okumak için kendi programınızı yazın

```
RSLimit u4limit(sih, bgh, dth);

u4limit.Initialize();

u4limit.TestHypothesis();
```

Adım2 : RSLimit değişkenini, 3 girdi histogramını kullanarak tamımla: sinyal, ardalan ve veri

Adım3 : RSLimit içayarları yap

Adım4 :Hesabı Çalıştır

```
double u4lim_obs    = u4limit.ExclusionLimitObs();
double u4lim_exp_m2s= u4limit.ExclusionLimitExp(-2);
double u4lim_exp_m1s= u4limit.ExclusionLimitExp(-1);
double u4lim_exp    = u4limit.ExclusionLimitExp(0);
double u4lim_exp_p1s= u4limit.ExclusionLimitExp(1);
double u4lim_exp_p2s= u4limit.ExclusionLimitExp(2);
```

Adım5 : Sonuçları al

iç özellikleri değiştirmek

- Kullanıcı iç özellikleri initialize komutundan önce değiştirmeli

```
RSLimit u4limit(sih, bgh, dth);


u4limit.VerbosityLevel(2);
u4limit.SetDiscriminator("m",100,500);
u4limit.UseProof(2);
u4limit.SetToyMC(5000,15000);
u4limit.SelectTestStats(1);
u4limit.SetCLsCLsb(0);
u4limit.Dump();



u4limit.Initialize();
u4limit.TestHypothesis();
```

RooFit/Stat sadece hataları yazacak

Only [100,500] range of the discriminant histogram will be used

Hesaplar 2 çekirdek üzerine dağıtılacak

5000 sinyal & 15000 ardalan MC olayı

0: Simple Likelihood Ratio
1: Ratio Of Profiled Likelihoods
2: Profile Likelihood (aka LHC)
3: Number Of Events

0: CL_s
1: CL_{s+b}

Show all pdfs, models, niusance parameters etc..

Bilinemezleri Eklemek

- Farklı bir sınıf başlatıcısı kullanılmalı
  - ➡ Kalanlar aynı

Sadece veri histogramıyla başlat

sinyal histogramını ekle, aşağı ve yukarı sıklık dalgalanmaları için yer, artı iki yer de şekil histogramlarına ayrılmış

```
RSLimit u4limit(dth);

u4limit.AddSignalHistogram(sih, 1.0, 1.2, NULL, NULL);

u4limit.AddBackgroundHistogram(bgh0, 0.8, 1.2, NULL, NULL);
u4limit.AddBackgroundHistogram(bgh1, 0.8, 1.2, NULL, NULL);
u4limit.AddBackgroundHistogram(bgh2, 1.0, 1.0, NULL, NULL);

u4limit.Initialize();
u4limit.TestHypothesis();
```

Ardalan histogramını ekle: aşağı ve yukarı sıklık dalgalanmaları,

1.0 : dalgalandırma

ters sorunu çözmek

- DoHTI işlevini testhypothesis komutundan önce doğru(=true) olarak vermek gerekli

```
RSLimit u4limit(dth);
u4limit.AddSignalHistogram(sih, 1.0, 1.2, NULL, NULL);
u4limit.AddBackgroundHistogram(bgh0, 0.8, 1.2, NULL, NULL);
u4limit.AddBackgroundHistogram(bgh1, 0.8, 1.2, NULL, NULL);
u4limit.AddBackgroundHistogram(bgh2, 1.0, 1.0, NULL, NULL);
u4limit.Initialize();
```

```
u4limit.DoHTI(true);

u4limit.TestHypothesis(10, 0.0, 50.0);

u4limit.DrawHTIPlot();
```

set HypoTest Inversion to TRUE

#Adıms, min and max values for scan, these values are the defaults

İstersek çizim yaparız, pdf olarak saklanır.

```
double u4lim_kf_m2s= u4limit.GetKFactorExp(-2);
double u4lim_kf_m1s= u4limit.GetKFactorExp(-1);
double u4lim_kf_med= u4limit.GetKFactorExp(0);
double u4lim_kf_p1s= u4limit.GetKFactorExp(1);
double u4lim_kf_p2s= u4limit.GetKFactorExp(2);
double u4lim_kf_obs= u4limit.GetKFactorObs();
```

Get the results

Yazılım

- Sürüm 0.7, OSX ve Linux ile denendi.
  - 07 Şubat 2012 tarihi
- Buradan indirin:
  - http://unel.web.cern.ch/unel/RSLimit.html
- İçindekiler
  - RSLimit.h, RSLimit.C : sınıf tanımlamaları
  - Readme : çalışmaya başlamak için gereken bilgiler
  - testRSLimit.C : kullanım örneği, tartıştıklarımızı gösterir
    - examplefit.root : denemeler için örnek root kütüğü
- Yapılacak işler - gönüllü var mı?
  - data fazla uyumluluk karşılaştırmaları
  - şekil sistematikleri eklenmeli
  - bir twiki sayfası yapılmalı
- Sizin önerileriniz

400	TLimit	RSLimit	mclimit
-2σ	0.91	0.95	0.94
-1σ	0.69	0.70	0.68
Exp	0.48	0.51	0.49
+1σ	0.35	0.39	0.36
+2σ	0.27	0.26	0.22
Obs	0.58	0.57	0.56