

Istatistik dersi ve Sınır hesaplama

HPFBU - 4

Gökhan Ünel - *UCI*
Sezen Sekmen - *Kyungpook Nat. Univ.*

Şubat 2015

sıklıkçılar ve bayesçiler tartışırken



- sıklıkçılar: "sadece ölçtüğüme inanırım!"
- bayesçiler: "önyargıma inanırım, çünkü sağduyudan gelir!"

Değiştirilmiş Sıklıkçı Yaklaşım (modified frequentist approach)

- hem sıklıkçı hem bayesçi fikirleri kullanır
- CLs yöntemi
 - ➔ BHÇ'de keşif yapmak için tercih edilen yol: veri \approx sinyal+ardalan VE veri \neq ardalan ise sinyali gördüm diyebilirim.
- Kavramlar
 - ➔ yok varsayımı: sinyal yoktur.
 - ➔ diğer varsayım: sinyal vardır.
 - ➔ gözlemlenebilirler: değişmez kütle, b-ışaretleme olasılığı, vb.
 - ➔ ölçme-istatistiği (test statistics): gözlemlenebilirleri ve kuram değişkenlerini kullanarak sonuçları ardalangibi'den sinyalgibi'ye doğru dizmenin bir ölçütü, Q, Q=0 ardalalan, Q=1 sinyal.
 - ➔ kurallar: ölçme-istatistiğinin hangi değerlerine sinyal gibi derim?
 - ➔ güven düzeyi, güven aralığı, güven sınırı: belli bir aralıkta, belli bir olasılıkla dışlama olayı ile ilgili

DSY -2- bazı eşitlikler

● ardaan (yok varsayımı) CL_b

$$CL_b = P_b(Q \leq Q_{obs}) = \int_{-\infty}^{Q_{obs}} \frac{dP_b}{dQ} dQ$$

→ Q_{obs} : ölçüm, gözlem, deney sonuçları

→ dP_b/dQ : sadece ardaan olan deneylerin olasılık dağılım işlevi (pdf), bunu öyle kurgulayım ki

- ▶ sonuçlar ardaan varsayımı ile uyumluysa $CL_b = 0$
- ▶ sonuçlar ardaan varsayımı ile uyumsuzsa $CL_b = 1$

● sinyal + ardaan (diğer varsayım) CL_{sb}

$$CL_{s+b} = P_{s+b}(Q \leq Q_{obs}) = \int_{-\infty}^{Q_{obs}} \frac{dP_{s+b}}{dQ} dQ,$$

→ dP_{sb}/dQ : sinyal + ardaan olan deneylerin olasılık dağılım işlevi (pdf), bunu öyle kurgulayım ki

- ▶ sonuçlar s+a varsayımı ile uyumsuzsa $CL_{sb} = 0$
- ▶ sonuçlar s+a varsayımı ile uyumluysa $CL_{sb} = 1$

● ve $CL_s \equiv CL_{sb} / CL_b$

→ CL güven düzeyinde sinyal varsayımını

dıştıalamak için gereken şart: $1 - CL_s \leq CL$.

	deney1 sinyal yok	deney2 sinyal az	deney3 sinyal çok
CL_{sb}	0.01	0.2	0.8
CL_b	0.1	0.3	0.9
CL_s	0.10	0.67	0.89
$1-CL_s$	0.90	0.33	0.11

basit deneyler

● Deney 1

- ⇒ elimde hiç sinyal yok, sadece istatistik dalgalanma var. bu deneyin sadece ardalana ile uyumsuzluğu az, yani CL_b sifira yakın, 0.1 diyelim. sinyal+ardalan ile ise daha da çok uyumsuz. 0.01 diyelim.
- ⇒ 90% ile sinyali dışlataladım. ($1-CL_s \leq CL$) güvenilir bir sonuç !

● Deney 2

- ⇒ elimde azıcık sinyal var. sadece ardalana varsayımı ile uyumsuzluğu önceki deneye göre arttı, 0.3 oldu. sinyal + ardalana ile uyumu arttı ama daha az arttı: çünkü az sinyal var: 0.2 diyelim.
- ⇒ sadece 33% ile sinyali dışlataladım. 33% pek güvenilir değil !

● Deney 3

- ⇒ elimde daha çok sinyal var. sırf ardalana ile neredeyse tamamen uyumsuzum, 0.9 diyelim. Ancak istatistik dalgalanmalar yüzünden tam da sinyal + ardalana ile uyumlu olamıyorum, 0.8 diyelim.
- ⇒ sadece 11% ile sinyali dışladım. yani 89% olasılıkla sinyal var olabilir !

dikkat: CL_s iki olasılığın oranıdır. Yani bazen 1'den büyük olabilir (ill defined).

Peki neden CL_{sb} veya CL_b kullanmıyoruz? Çünkü sadece sadece s+a varsayımı ile uyumlu olmak yetmez. Aynı zamanda yok varsayımı ile de uyumsuz olmak lazım. Bir tür normalleştirme yapıyoruz.

	<i>deney1</i> <i>sinyal yok</i>	<i>deney2</i> <i>sinyal az</i>	<i>deney3</i> <i>sinyal çok</i>
CL_{sb}	0.01	0.2	0.8
CL_b	0.1	0.3	0.9
CL_s	0.10	0.67	0.89
$1-CL_s$	0.90	0.33	0.11

bir örnek

● LHC (daha önce de Tevatron)

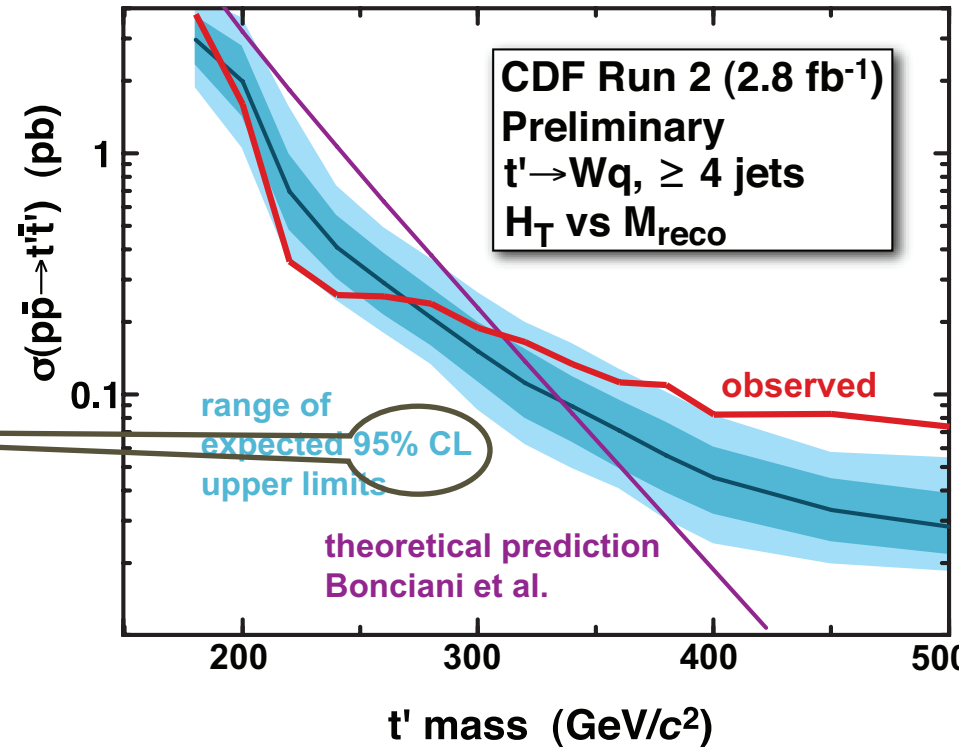
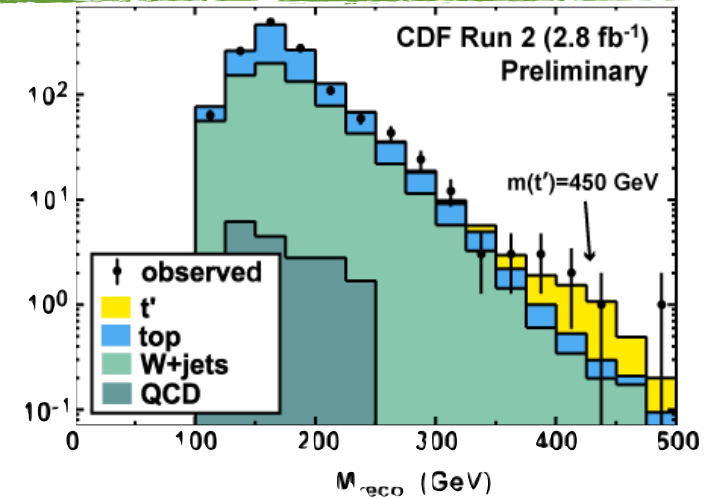
- ➔ kenar bölgesi SM ile uyumlu
- ➔ sinyal bölgesinde çok az olay var
 - ölçüm hassasiyetimiz az

● Bu arada

- ➔ eğer aradığımız değişken gauss gibi dağılım gösteriyorsa:

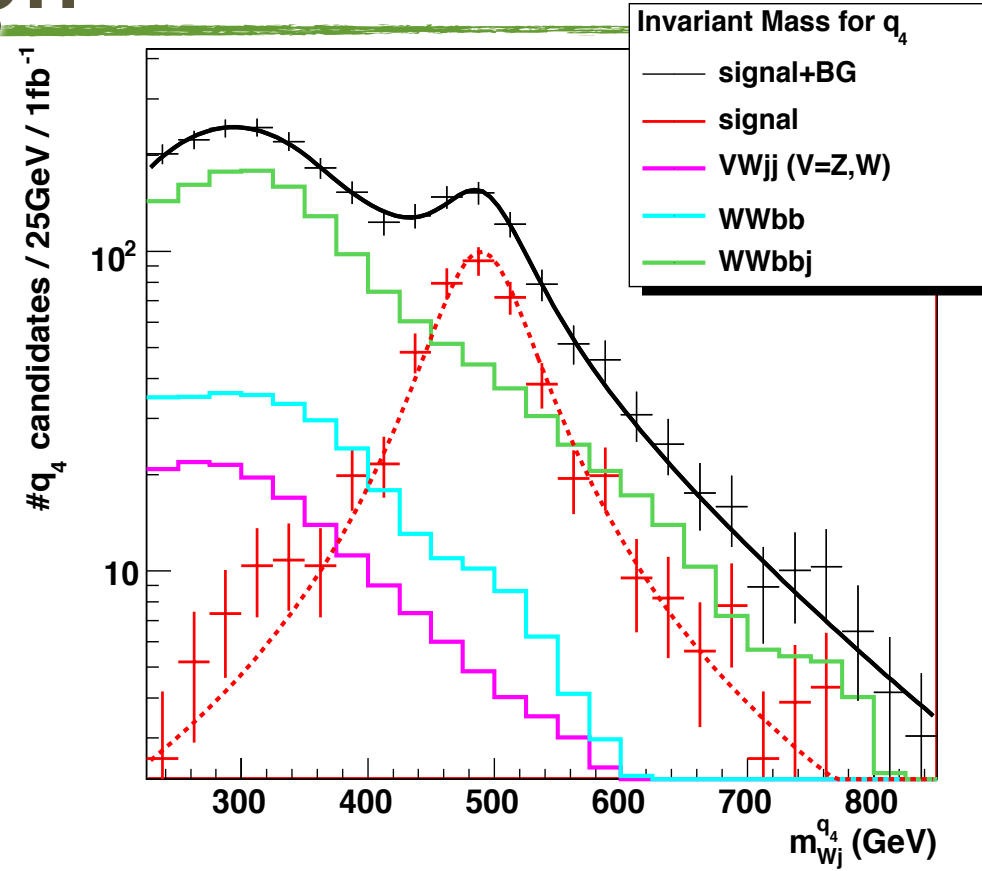
α	δ	α	δ
0.3173	1σ	0.2	1.28σ
4.55×10^{-2}	2σ	0.1	1.64σ
2.7×10^{-3}	3σ	0.05	1.96σ
6.3×10^{-5}	4σ	0.01	2.58σ
5.7×10^{-7}	5σ		

CL : ConfidenceLevel - GüvenDüzeyi
 α : yanılma olasılığı



Keşif

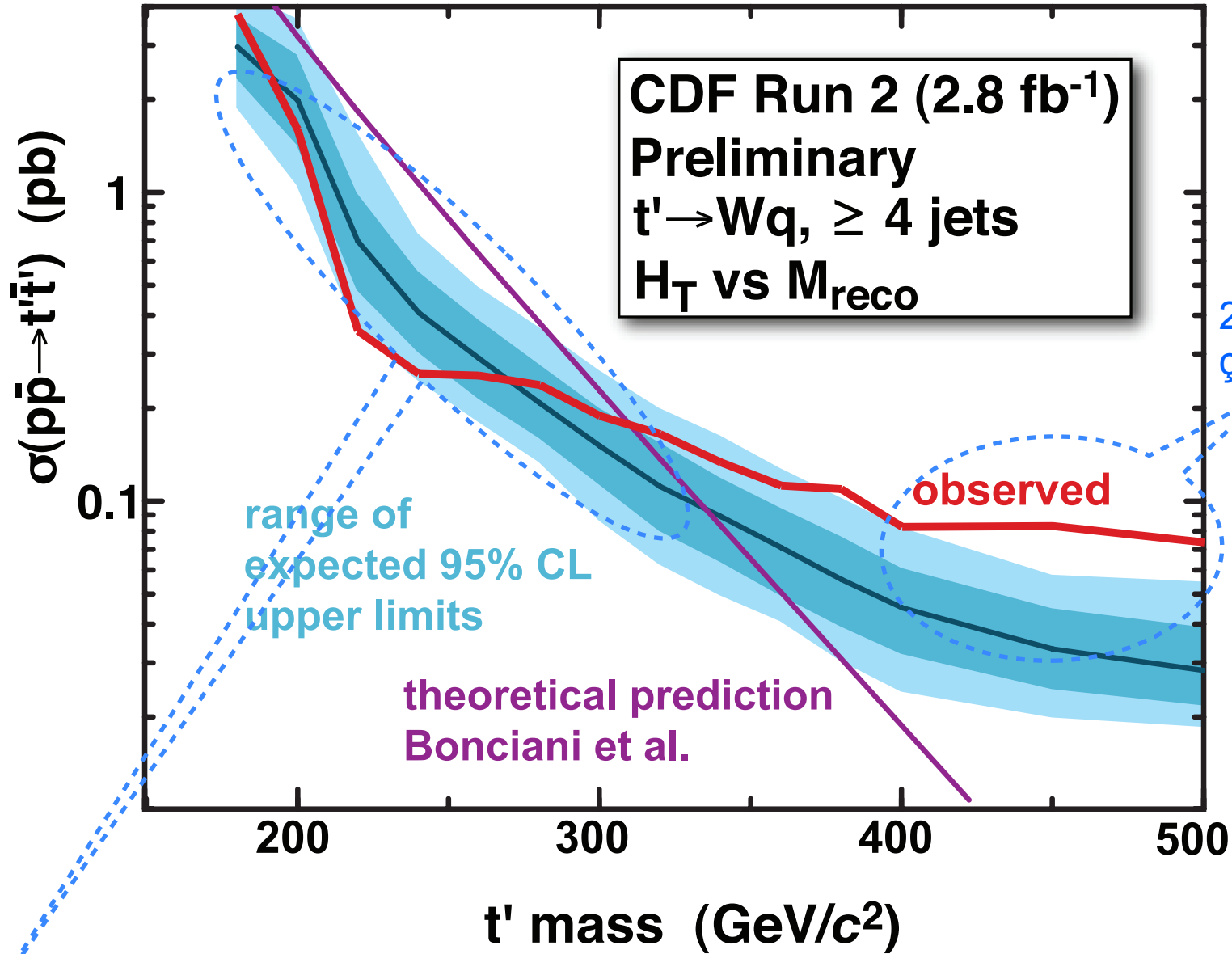
- LHC çalışmadan önce ve ilk veriler toplanırken keşif yapıvermeyi hayal ediyorduk.
 - ➔ sinyal olayları arkaplan üzerine rahatça çıkacaktı
- Bu durumda
 - ➔ elimizdeki tüm veriye bir eğri yakıştıracak
 - ▶ kenarlardan arkaplanı
 - ▶ farklılık gösteren yerden sinyali
 - ➔ Alıp, olay sayılarını hesaplayacaktık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



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

ama böyle olmadı

konuşalım



2hatapayından çok fazlalık var.

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

Kullanılan yöntemler

● girdiler - sonuçlar

- ➔ kağıt kalem --> kuramsal tesir k.
- ➔ sinyal MC, ardalın MC --> beklenen tesir k.
- ➔ sinyal MC, ardalın MC, veri --> gözlenen tesir k.

● Soru

- ➔ "veri" ve "eğreti-veri" (pseudodata) hangi durum ile uyumlu?
 - ▶ Null Hypothesis = Yok Varsayımı
 - ▶ Alt Hypothesis = Diğer Varsayım

● Ölçüm

- ➔ ilgilendiğim değişkenin bir çok değeri için uyum testleri yap

● Kullanılan kıstaslar

- ➔ p-değeri: olasılık değeri, varsayım testi sonucu, CL (Güven Düzeyi)
- ➔ CL_b : 1-p: genel ardalın uyumluluğu, CL sadece ardalın varsayımı için
- ➔ CL_{s+b} : CL signal + ardalın varsayımı için, hırslı keşif yapmak istersek.
- ➔ CL_s : CL_{s+b}/CL_b . 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 bilinmeyenler

● İstatistik hatalar

➔ yeterince olayım var mı? (hem MC hem de veri)

● Systematik-düzenli hatalar veya bilinmeyenler (uncertainties)

➔ Ölçtüğlerimi iyi ölçtüm mü? 2 şekilde kötü ölçebilirim:

➔ Rate = Sayma bilinmeyenleri olabilir

▶ genel şekil aynı,

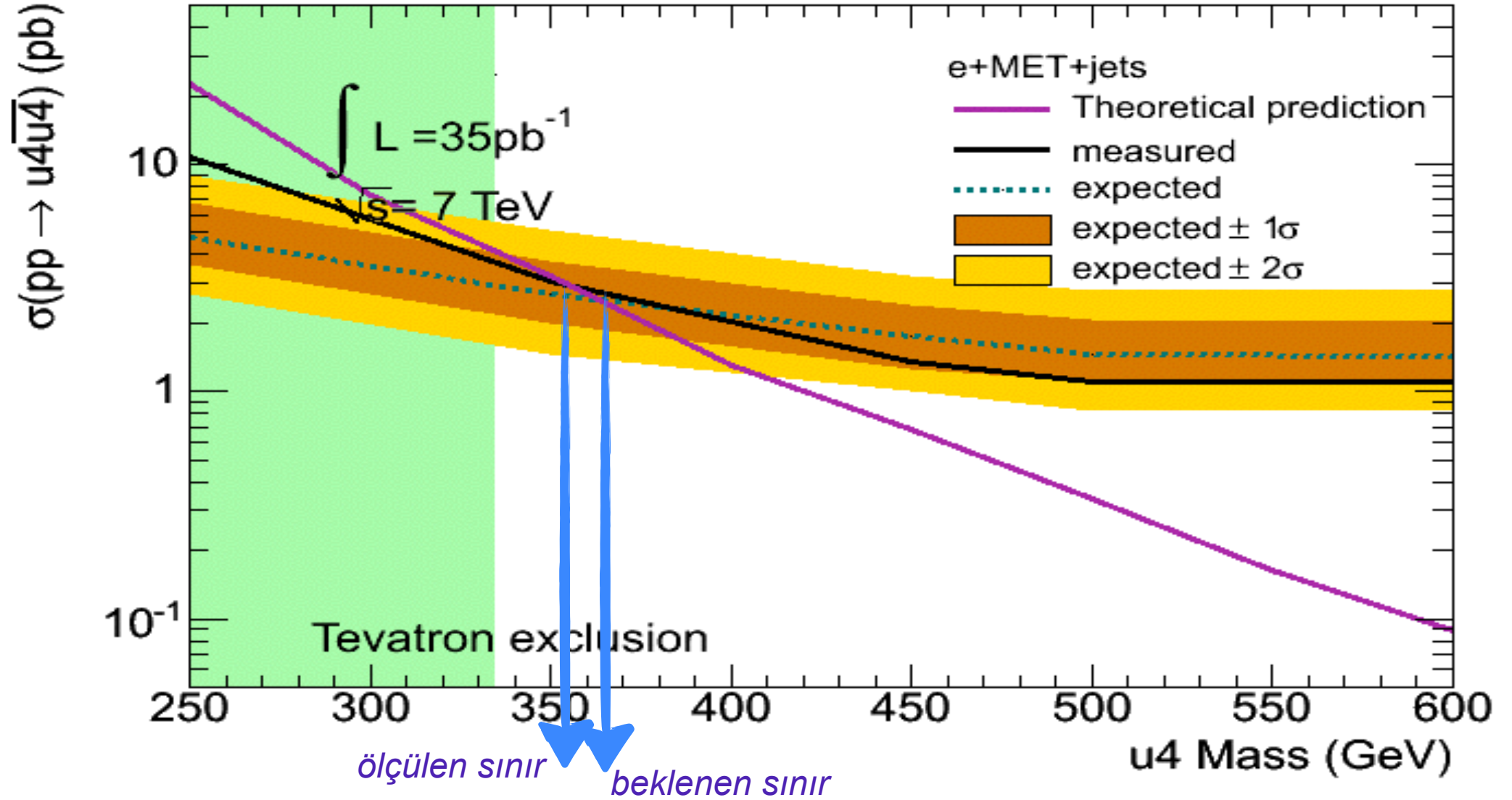
▶ histogramdaki her kutu yeni katsayı ile yukarı veya aşağı dalgalanabilir.

➔ Shape = Şekil bilinmeyenleri olabilir

▶ genel şekil değişebilir,

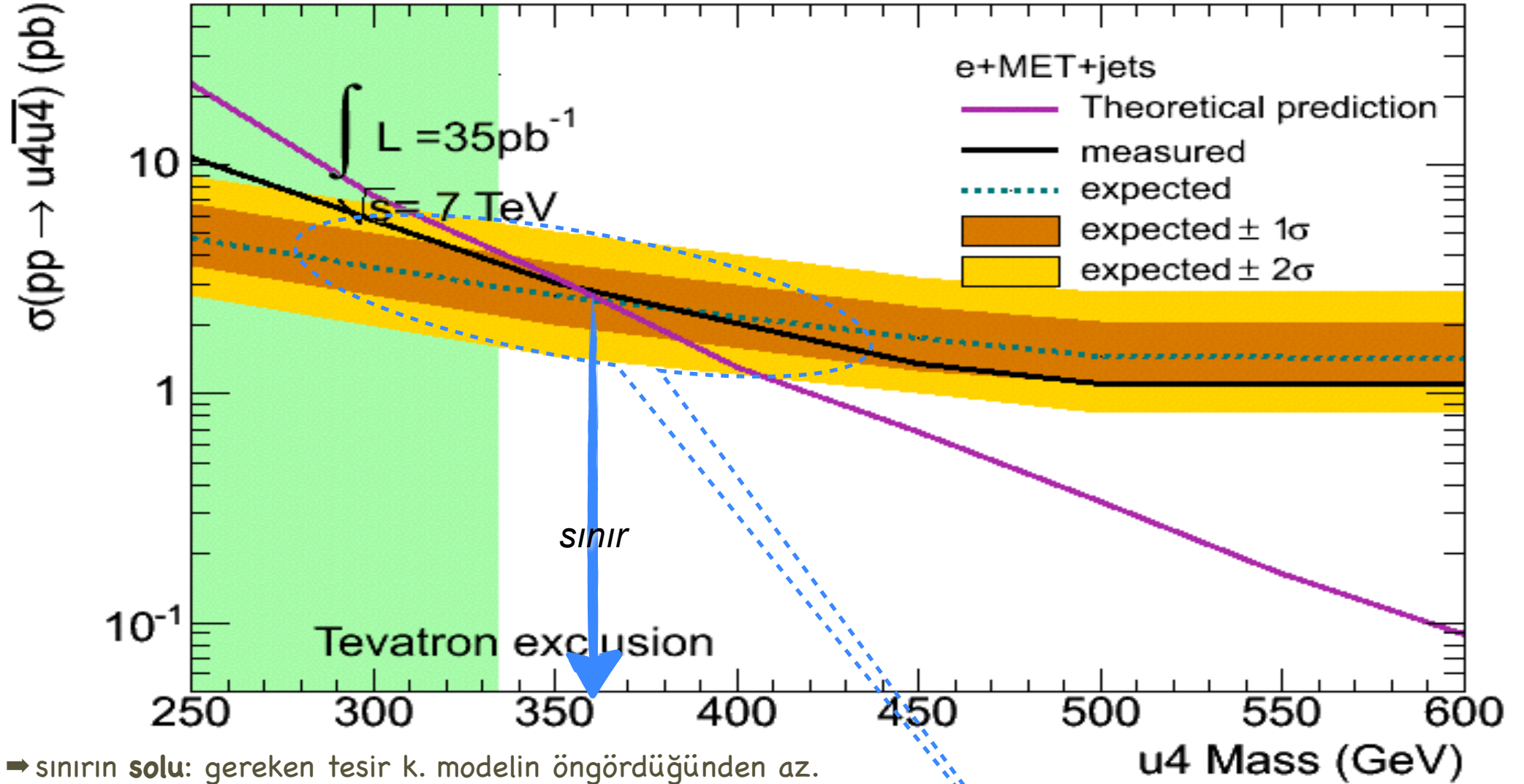
▶ histogramdaki her kutu farklı katsayı ile yukarı veya aşağı dalgalanabilir.

benzer bir 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.

benzer bir çizime yakından bakalım

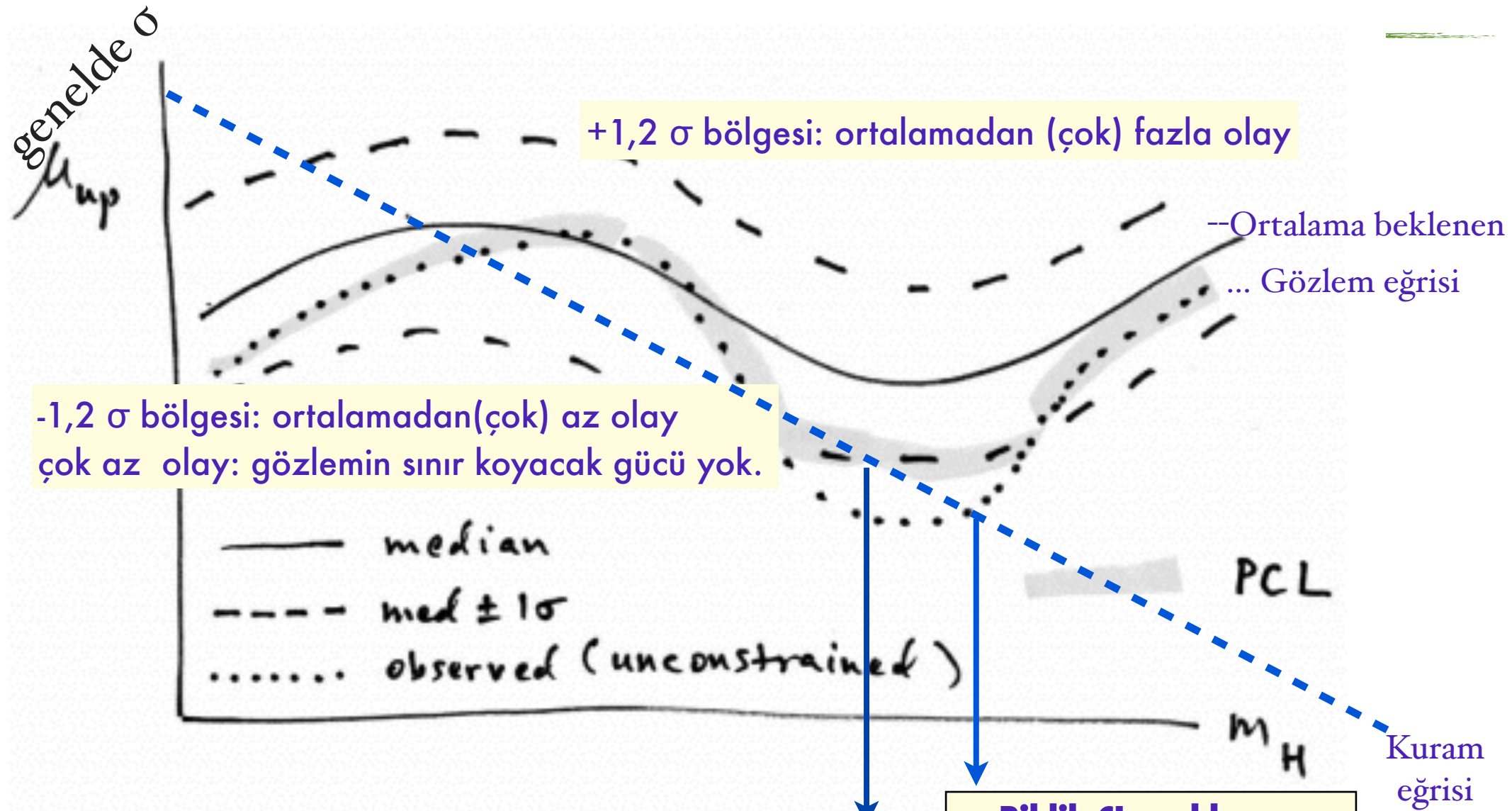


→ sınırın **solu**: gereken tesir k. modelin öngördüğünden az.
modelim bana gerekli olay sayısını temin edebilir, **ölçüm yapabilirim**

→ sınırın **sağı**: gereken tesir k. modelin öngördüğünden çok.
modelim bana yeteri kadar olay veremiyor, yapacağım ölçümler sınır koymak için **yetersiz**

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

Gücü Engellenmiş Sınır (power constrained limit) yaklaşımı



PCL-GES yaklaşımı

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

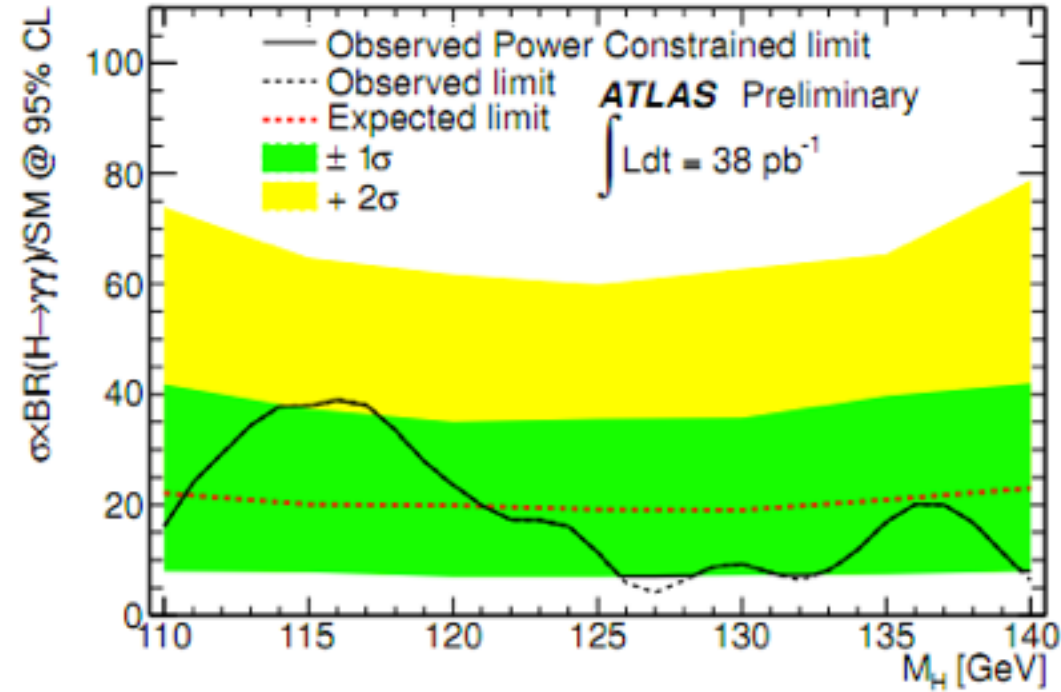
Bildik CLs yaklaşımı

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 çözümlmelerinde 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** : RooStats'ı basitleştirmek için bizim yaptığımız yazılım.

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.

- Kanal eklemek basittir

```
mydatasource->AddChannel(toplams,toplamb,toplamq);
if (sum_channels) { // read the other lepton channel and add.
  mydatasource->AddChannel(toplams_other,toplamb_other,toplamq_other);
}
```

Çalışma 1

```
{
  TH1F *hdata = new TH1F("hdata","Fake data histogram",10,0,10);
  TH1F *hbkg = new TH1F("hbkg" ,"Expected bg histogram",10,0,10);
  TH1F *hsig = new TH1F("hsig" ,"Expected signal histogram",10,0,10);
  TH1F *hspb = new TH1F("hspb" ,"Signal + bkg histogram",10,0,10);
```

```
TF1 *fpoly = new TF1("fpoly","[0]+[1]*x+[2]*x*x",0,10);
fpoly->SetParameters(3,20,-2);
fpoly->Draw();
hbkg->FillRandom("fpoly",10000);
fpoly->SetParameters(1,0,0);
hsig->FillRandom("fpoly",10000);
//hbkg->Draw();
hdata->FillRandom(hbkg,100);
hdata->Draw("e");
hbkg->Scale(100./10000);
hbkg->Draw("same");
hsig->Scale(5./10000);
hsig->SetLineColor(kRed);
hsig->Draw("same");
hspb->Add(hsig,hbkg);
hspb->SetLineColor(kGreen);
hspb->Draw("hsame");
}
```

- 1) TLimit kullanarak 5000 olaylı bir MC ile CLb, CLsb, CLs değerlerini beklenen ve gözlenen durumlar için hesap edin.
- 2) sinyali biraz artırın azaltın ve bunun etkisini CL değerlerinde görün.

ters sorun

● durum

- ➔ si ve bg histogramlarımız : sih ve bgh olsun,
- ➔ si tesir kesitim : x_0 olursa
- ➔ aldığımız CL değerleri, CLO olur.

● soru:

- ➔ CL = 95% durumunu bana verecek si tesir kesiti nedir?
 - ▶ Aslında 0.05e bakılır.
 - ▶ buna `ters sorun' adı verilir

● olası cevap ve yöntem

- ➔ kuramsal tesir kesitine x_0 diyelim
- ➔ CL=0.05 verecek olan tesir kesiti $s \cdot x_0$ olsun
 - ▶ $s=1$ den başlayip s 'i arttıyım veya azaltayım
 - ▶ her adımda CL degerini tekrar hesap edeyim, 0.05'den farkına bakayım
 - ▶ taa ki fark sifira inene kadar.

RSLimit yazılımı

● RooStats kütüphanesi etrafına bir sarma(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.

● Gerekirir:

- ➔ C++
- ➔ Root 5.30/01 (RooFit ile derlenmiş olmalı) veya daha ileri
 - ▶ eski ROOT sürümleri farklı API kullanırdı.

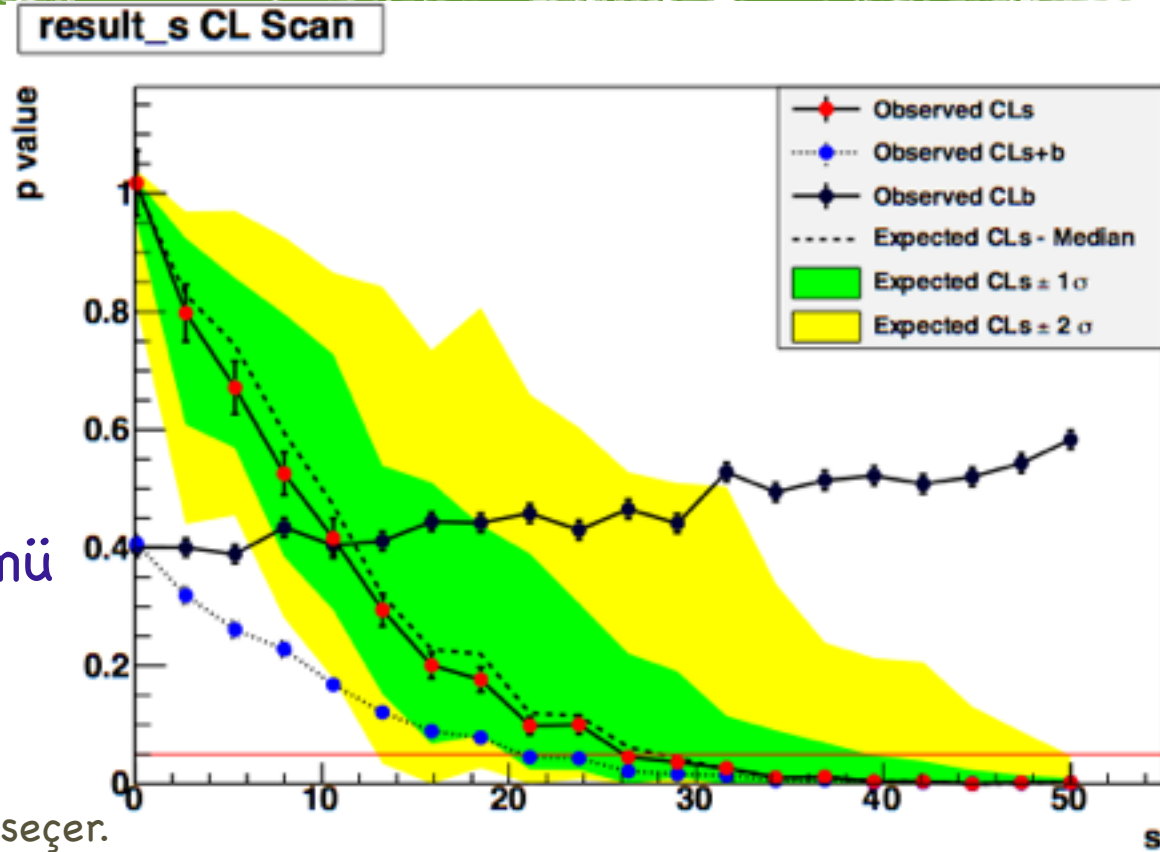
● Sunar:

- ➔ 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
- Varsayım testi - ters sorun çözümü
 - ➔ GüvenDüzeyini kullanıcı belirler (açılışt= 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 ardalın 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 tanımla: sinyal, aralan 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);
```

RooFit/Stat sadece hataları yazacak

```
u4limit.VerboesityLevel(2);
```

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

```
u4limit.SetDiscriminator("m", 100, 500);
```

Hesaplar tüm çekirdekler üzerine dağıtılacak, 2 eğreti deney yap. (?)

```
u4limit.UseProof(2);
```

```
u4limit.SetToyMC(5000, 15000);
```

5000 sinyal & 15000 aralan MC olayı

```
u4limit.SelectTestStats(1);
```

```
u4limit.SetCLsCLsb(0);
```

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

```
u4limit.Dump();
```

*0: CL_s
1: CL_{s+b}*

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

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

Show all pdfs, models, nuisance parameters etc..

Bilinemezleri Ekleme

- 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);
```

Ardalan histogramını ekle: aşağı ve yukarı sıklık dalgalanmaları,

1.0 : dalgalandırma

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

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();

```

*set HypoTest
Inversion to TRUE*

```

u4limit.DoHTI(true);
u4limit.TestHypothesis(10, 0.0, 50.0);
u4limit.DrawHTIPlot();

```

*#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 0.8, OSX ve Linux ile denendi.

➔ 22 Şubat 2012 tarihli, readme'yi okuyun.

- Buradan indirin:

➔ <http://unel.web.cern.ch/unel/RSLimit.html>

Çalışma 2

- İçindekiler

➔ RSLimit.h, RSLimit.C : sınıf tanımlamaları

➔ Readme : çalışmaya başlamak için gereken bilgiler

➔ testRSLimit.C : kullanım örneği, tartışıklarımızı gösterir

▶ examplefit.root : denemeler için örnek root kütüğü

- Yapılacak işler - gönüllü var mı?

➔ data fazla uyumluluk karşılaştırmaları

➔ şekil sistematiikleri eklenmeli

➔ kanal ekleme değişiklikleri yapılmalı

➔ bir twiki sayfası yapılmalı

- Sizin önerileriniz

400	<i>TLimit</i>	<i>RSLimit</i>	<i>mclimit</i>
-2 σ	0.91	0.95	0.94
-1 σ	0.69	0.7	0.68
<i>Exp</i>	0.48	0.51	0.49
+1 σ	0.35	0.39	0.36
+2 σ	0.27	0.26	0.22
<i>Obs</i>	0.58	0.57	0.56