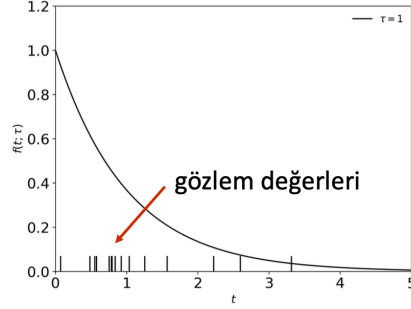


# Olasılık ve İstatistik

## Ödev: Üstel olarak dağıtılmış verilerle istatistik

Haftanın ilerleyen günlerinde;  
Deneyin n kez tekrarlandığını  
ve şu verileri verdiğini  
varsayalım:  $t_1, \dots, t_n$ .



Veri değerlerini kullanarak, ortalama ömür  $\tau$ 'yi tahmin edin.

Tahmin içindeki istatistiksel belirsizliği niceleyin.

Ortalama ömür için üst/alt sınırları bildirin.

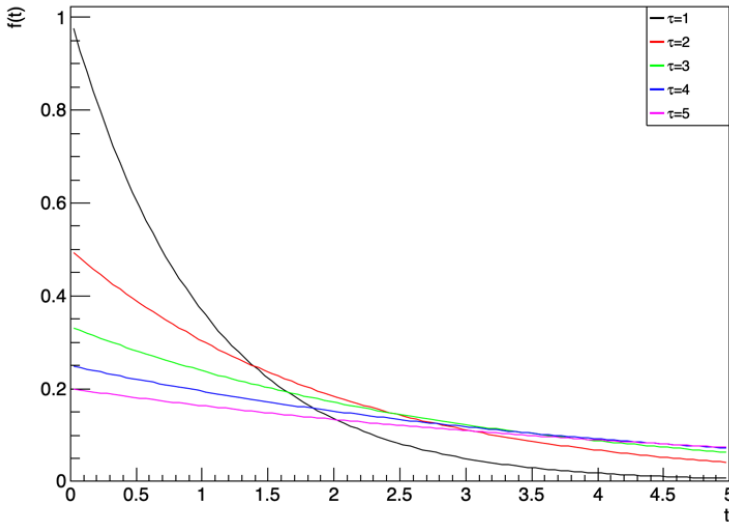
ocakir - zsh

### Çözüm:

Üstel olasılık dağılım fonksiyonu

$$f(t; \tau) = (1/\tau)e^{-t/\tau}$$

Farklı  $\tau$  parametreleri için aşağıdaki gibi çizilebilir.



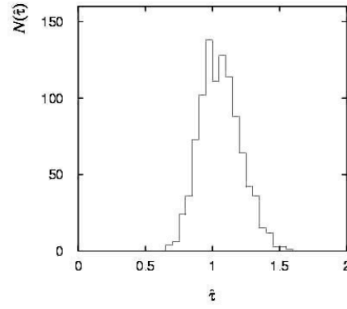
Parametremizi tahmin ettikten sonra şimdi onun 'istatistiksel hatasını', yani tüm ölçümü birçok kez tekrarlırsak tahminlerin ne kadar yaygın olarak dağılacağını bildirmemiz gerekiyor.

Bunu yapmanın bir yolu, tüm deneyi bir Monte Carlo programıyla birçok kez simüle etmek olacaktır (MC için ML tahmini kullanın).

Üstel örnek için, tahminlerin örneklem varyansından şunu buluruz:

$$\hat{\sigma}_{\hat{\tau}} = 0.151$$

Tahminlerin dağılımının yaklaşık olarak Gauss olduğunu unutmayın – büyük örneklem limitinde ML için (neredeyse) her zaman doğrudur.



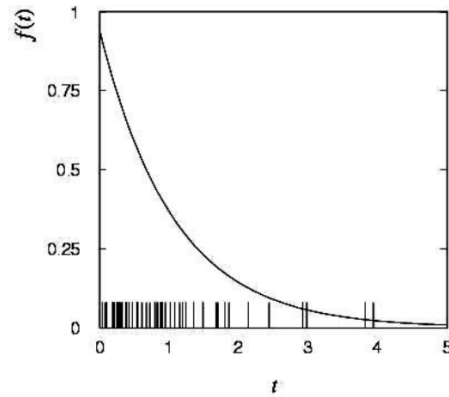
Maksimum değerini bulun  $\frac{\partial \ln L(\tau)}{\partial \tau} = 0$ ,

$$\rightarrow \hat{\tau} = \frac{1}{n} \sum_{i=1}^n t_i$$

Monte Carlo test için:  
50 değer üretin  
 $\tau = 1$  kullanın:

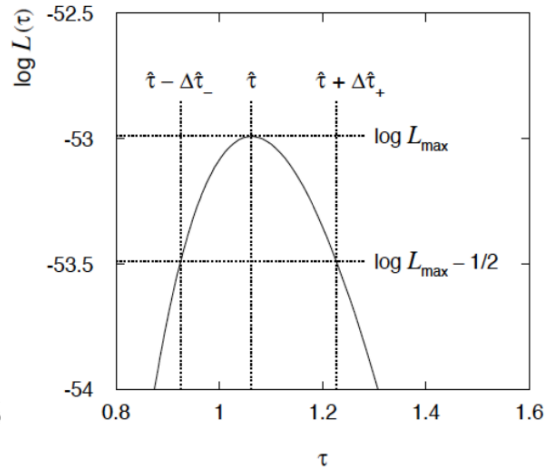
ML tahmin ediciyi buluruz:

$$\hat{\tau} = 1.062$$



Üstel ile ML örneği:

$$\begin{aligned}\hat{\tau} &= 1.062 \\ \Delta\hat{\tau}_- &= 0.137 \\ \Delta\hat{\tau}_+ &= 0.165 \\ \hat{\sigma}_{\hat{\tau}} &\approx \Delta\hat{\tau}_- \approx \Delta\hat{\tau}_+ \approx 0.15\end{aligned}$$



Sonlu örneklem büyüklüğü ( $n = 50$ ) olduğundan  $\ln L$ 'de tam olarak parabolik değil.

Buradaki grafikler ve hesaplamalar pyROOT veya Python ile yapılabilir. Örnek bir kod aşağıda verilmiştir ( $\tau = 1$ ,  $N = 1000$ ).

```
import ROOT
import numpy as np
import matplotlib.pyplot as plt
```

```
random = ROOT.TRandom3()
tau = 1.0
n_entries = 50
```

```
hist = ROOT.TH1F("hist", "", 100, 0, 10)
for i in range(n_entries):
    value = -tau * ROOT.TMath.Log(1-random.Uniform(0,1)) #
Inverse transform sampling
    hist.Fill(value)
```

```
hist.Scale(1/hist.GetMaximum())
```

```
tau_2 = np.sqrt( hist.GetMean() )
f1 = ROOT.TF1("f1", "(1/[0])*exp(-x/[0])", 0, 10)
f1.SetParameter(0, tau_2)
```

```
c = ROOT.TCanvas()
hist.Draw("hist")
hist.GetXaxis().SetTitle("t")
hist.GetYaxis().SetTitle("f(t)")
```

```
f1.Draw("same")
c.Draw()
```

```
import ROOT
```

```
random = ROOT.TRandom3()
tau_values = np.arange(0.8, 1.6, 50)
n_entries1 = 1000
n_entries2 = 1000
```

```
canvas = ROOT.TCanvas("canvas", "E", 800, 600)
hist = ROOT.TH1F("hist", "#tau dist", 100, 0., 2)
hist_ = ROOT.TH1F("hist1", "", 100, 0., 5.)
for _ in range(n_entries1):
    tau = random.Uniform(0.8, 1.6)
```

```
    for i in range(n_entries2):
        value = -tau * ROOT.TMath.Log(1-random.Uniform(0, 1))
        hist_.Fill(value)
    hist_.Scale(1/hist_.GetMaximum())
```

```
hist.Fill(hist_.GetMean())
```

```
hist.Draw()
hist.GetAxis().SetTitle("N(#tau)")
hist.GetAxis().SetTitle("#tau")
```

```
canvas.Draw()
```

```
N = 10000
ts = -1*np.log(1-np.random.uniform(0, 1, N))
taus = np.linspace(0.9, 1.1, N)
```

```
lnL = []
for tau in taus:
    lnL.append( -N*np.log(tau)-np.sum(ts)/tau )
```

```
mle_tau = taus[np.argmax(lnL)]
lnL_max = max(lnL)
```

```
threshold = lnL_max - 0.5
```

```
tau_min = taus[lnL >= threshold][0]
```

```
tau_max = taus[lnL >= threshold][-1]
```

```
lower_error = mle_tau - tau_min
```

```
upper_error = tau_max - mle_tau
```

```
plt.plot(taus, lnL, color="black")
```

```
plt.axvline(mle_tau, color="red", label=r"MLE ( $\hat{\tau}$ )" + f" = {mle_tau:.4f}")
```

```
plt.axvline(tau_min, color="green", linestyle='--', label=f"Alt Sınır = {tau_min:.4f}")
```

```
plt.axvline(tau_max, color="blue", linestyle='--', label=f"Üst Sınır = {tau_max:.4f}")
```

```
plt.xlabel(r' $\tau$ ')
```

```
plt.ylabel('ln(L)( $\tau$ )')
```

```
plt.legend(loc="best")
```

```
plt.grid()
```

```
plt.show()
```

