Procesamiento y Análisis de Datos Astronómicos

7.- Pruebas de hipótesis

R. Gil-Hutton

Marzo 2020

Actividades:

Práctica 6:

- La función de distribución uniforme entre dos valores reales a y b es $f(x; a, b) = (b - a)^{-1}$ para $a \leq x \leq b$. Para esta distribución de probabilidades, cuánto valen μ y σ^2?.
- Un conjunto de 300000 vectores tienen sus tres componentes distribuidos de manera normal con $\sigma^2 = 1$ y $\mu = 0$. Simular la distribución de los módulos de los vectores, calcular cuánto valen μ y σ^2 en este caso y verificar que tienen una distribución Maxwelliana.

Actividades:

Práctica 6 (cont.):

- Repita el último ejercicio de la práctica anterior con muestras de 21 galaxias, pero ahora trabajando con la mediana. Ajuste a los resultados la función de distribución correspondiente.

Actividades:

En una función de distribución uniforme tenemos que $f(x; a, b) = (b - a)^{-1}$ para $a \leq x \leq b$, su valor medio será:

$$\mu = \frac{1}{b-a} \int_a^b x \, dx = \frac{1}{2(b-a)} \, x^2 |_a^b = \frac{b^2 - a^2}{2(b-a)} = \frac{1}{2}(a+b)$$

Actividades:

y su varianza:

$$\sigma^2 = \frac{1}{b-a}\int_a^b (x-\mu)^2 dx =$$

$$= \frac{1}{3(b-a)}(x-\mu)^3\Big|_a^b =$$

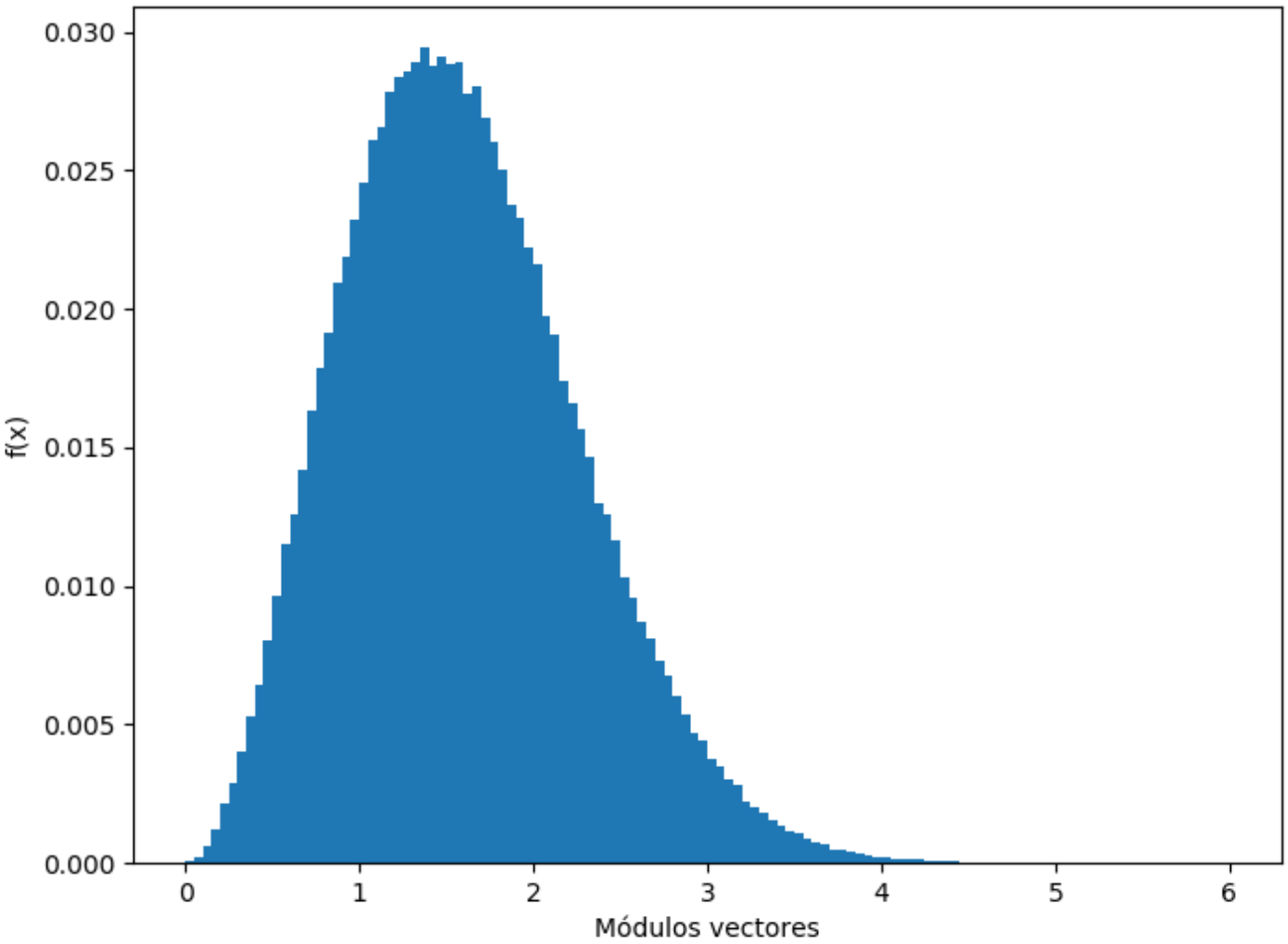
$$= \frac{1}{3(b-a)}\left[\left(b-\frac{a+b}{2}\right)^3 - \left(a-\frac{a+b}{2}\right)^3\right] =$$

$$= \frac{1}{3(b-a)}\left[\left(\frac{b-a}{2}\right)^3 - \left(\frac{a-b}{2}\right)^3\right] = \frac{(b-a)^2}{12}$$

Actividades:

```
In [4]: cx=np.random.standard_normal(300000)

In [5]: cy=np.random.standard_normal(300000)

In [6]: cz=np.random.standard_normal(300000)

In [7]: vv=np.sqrt(cx**2+cy**2+cz**2)

In [8]: np.min(vv),np.max(vv)
Out[8]: (0.03210902898291624, 5.512825447247643)

In [9]: his,lim=np.histogram(vv,bins=120,range=(0,6))

In [10]: his=his/np.sum(his)

In [11]:
```

Actividades:



Actividades:

En una distribución maxwelliana tenemos que:

$$f(x|a) = \sqrt{\frac{2}{\pi}} \frac{x^2}{a^3} \exp\left(-\frac{x^2}{2a^2}\right)$$

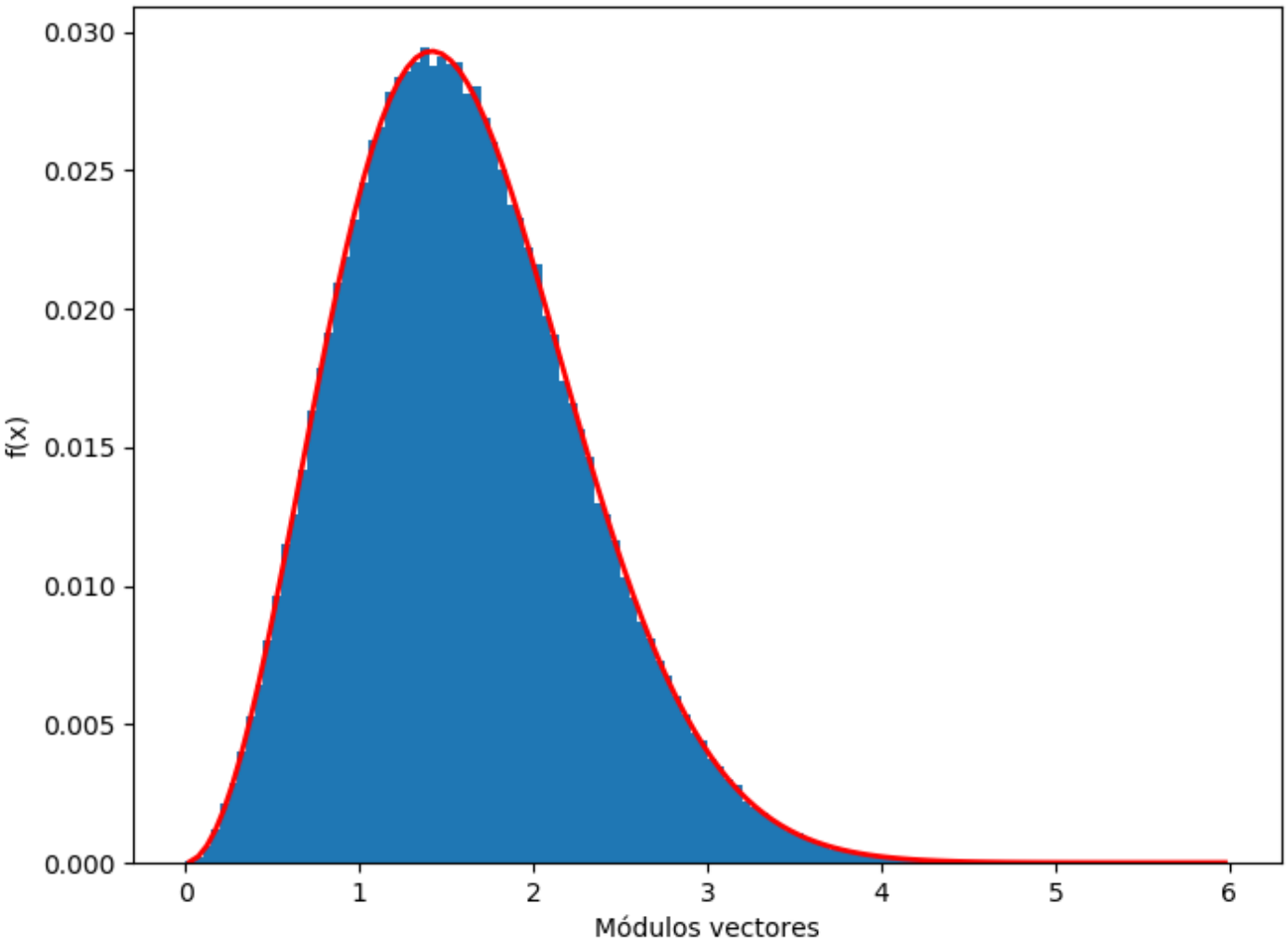
y su valor medio y varianza es:

$$\mu = 2a\sqrt{\frac{2}{\pi}}, \quad y \quad \sigma^2 = \frac{a^2(3\pi - 8)}{\pi}$$

Actividades:

```
In [12]: xx=lim[1:]-0.025

In [13]: vmed=np.mean(vv)

In [14]: aa0=vmed/2.*np.sqrt(np.pi/2.)

In [15]: vvar=np.var(vv,ddof=1)

In [16]: aa1=np.sqrt(vvar*np.pi/(3.*np.pi-8))

In [17]: aa0,aa1
Out[17]: (1.0012689580800735, 1.0015641596127944)

In [18]: aa=(aa0+aa1)/2.

In [19]: fmax=np.sqrt(2./np.pi)*xx**2/aa**3*np.e**(-xx**2/2./aa**2)

In [20]: fmax=fmax/np.sum(fmax)

In [21]:
```

Actividades:



Actividades:

Pama muestras de $N = 21$ elementos la mediana corresponde al elemento 11 de la muestra ordenada (índice 10 en Python), entonces el estadístico de orden 11 tienen una distribución:

$$g_{(11)}(x) = \frac{21!}{10!\ 10!} f(x)[F(x)]^{10}\ [1 - F(x)]^{10}$$

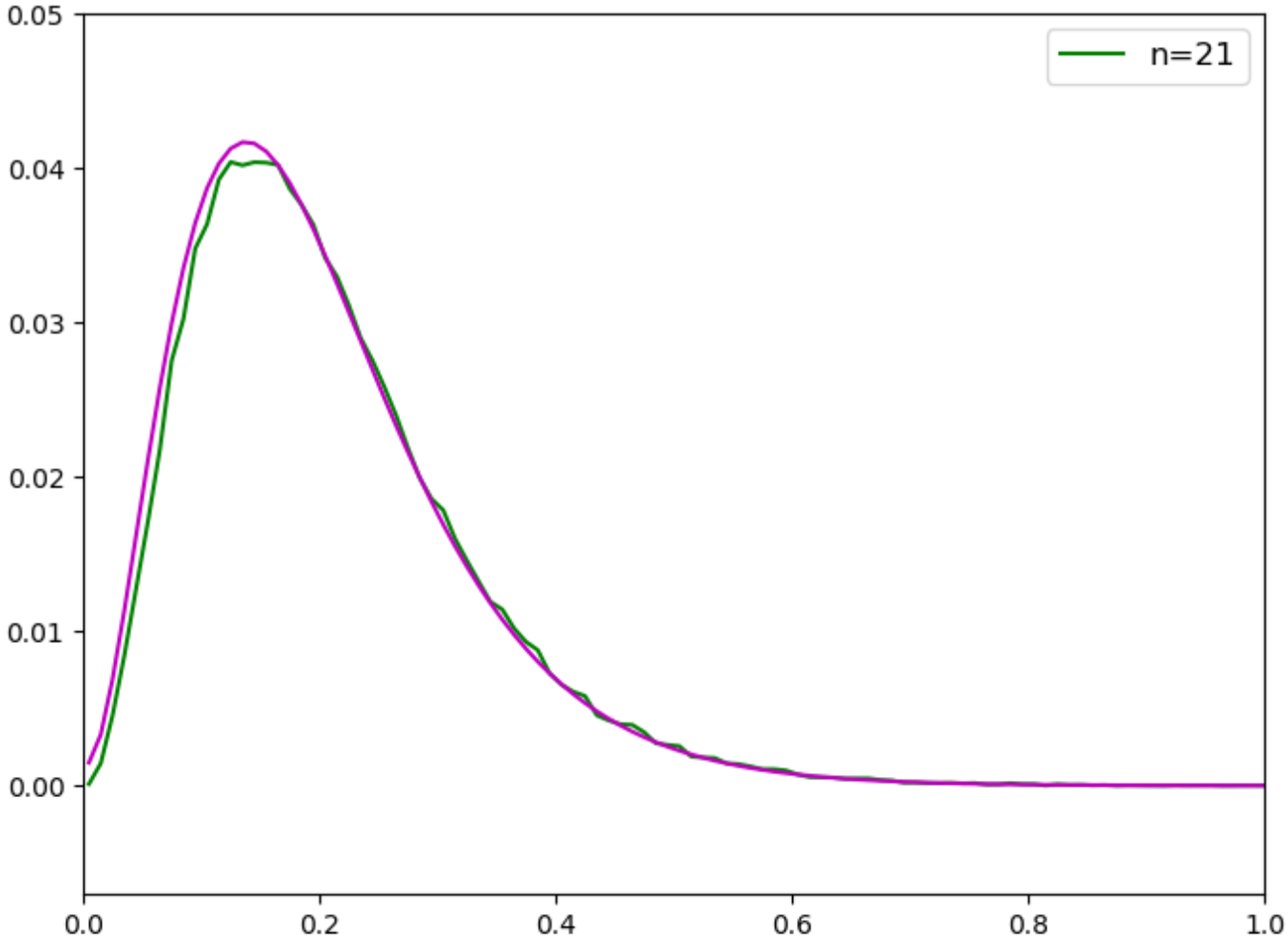
Actividades:

```
import numpy as np

def prmed(x,ac,n,p):
  """
  Extrae p elementos de n valores cada uno del array x tomados al azar,
obtiene la mediana y  el histograma correspondiente. El ancho del bin es
de 0.05 para 600 bins entre 0 y 6.
  """
  vx=[]
  for ii in range(p):
      kk=np.random.random(n)
      sx=[]
      for jj in range(n):
          vp=0
          while(kk[jj] > ac[vp]):
            vp+=1
          sx.append(x[vp])
      vx.append(np.median(sx))
  vx=np.array(vx)
  pes=np.ones(len(vx))/float(p)
  his=np.histogram(vx,bins=600,range=(0,6),weights=pes)
  return his
```

Actividades:

```
In [67]: xx=np.linspace(0.001,6,600)

In [68]: yy=xx**(-0.5)*np.exp(-xx)

In [69]: yy=yy/np.sum(yy)

In [70]: ac=np.cumsum(yy,dtype=float)

In [71]: his0=prmed(xx,ac,21,100000)

In [72]: import scipy.special as spc

In [73]: g21=spc.factorial(21)/spc.factorial(10)/spc.factorial(10)*ac**10*(1.-ac
    ...: )**10*yy
```

Actividades:



Trabajando con distribuciones:

Scipy.stats tiene la posibilidad de definir distintos tipos de distribuciones y operar sobre ella mediante diferentes métodos. Las distribuciones disponibles más conocidas son:

```
Algunas distribuciones discretas:
   binom             -- Binomial
   poisson           -- Poisson

Algunas distribuciones contínuas:
   chi2              -- Chi-squared
   cosine            -- Cosine
   expon             -- Exponential
   f                 -- F (Snecdor F)
   gamma             -- Gamma
   beta              -- Beta
   maxwell           -- Maxwell
   norm              -- Normal (Gaussian)
   t                 -- Student's T
   uniform           -- Uniform
```

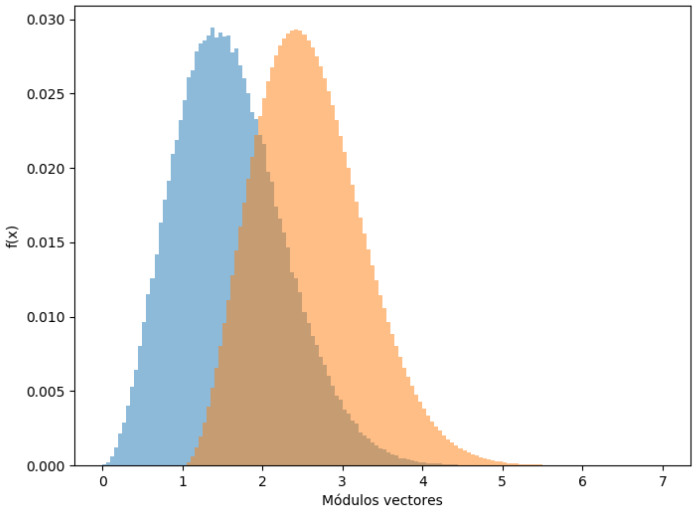
Trabajando con distribuciones:

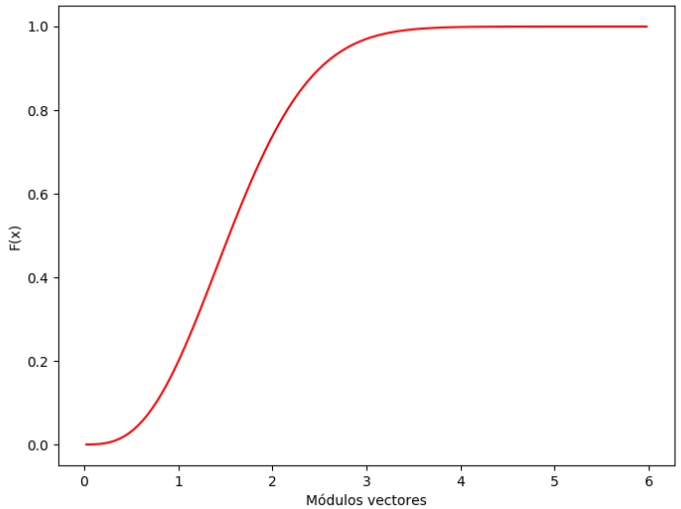
Los métodos disponibles más usuales son:

```
Algunos métodos genéricos:
rvs(loc=0, scale=1, size=1, random_state=None)
    Genera valores aleatorios.
pdf(x, loc=0, scale=1)
    Función de distribución de probabilidades.
cdf(x, loc=0, scale=1)
    Función acumulativa de distribución de probabilidades.
ppf(q, loc=0, scale=1)
    Función de punto porcentual.
stats(loc=0, scale=1, moments='mv')
    Media('m'), varianza('v'), sesgo('s'), y/0 kurtosis('k').
fit(data, loc=0, scale=1)
    Estimación de parámetros para un conjunto de datos.
interval(alpha, loc=0, scale=1)
    Extremos del rango con un porcentaje alpha de la distribución.
```

Trabajando con distribuciones:

```
In [104]: import scipy.stats as sts

In [105]: sts.maxwell.fit(vv)
Out[105]: (-0.001056142887115674, 1.0018765954121829)

In [106]: aa
Out[106]: 1.0014165588464339

In [107]: sts.maxwell.fit(vv,floc=0.)
Out[107]: (0.0, 1.0013179682454365)

In [108]: hsim=sts.maxwell.pdf(xx,loc=0.,scale=1.00187659)

In [109]: hsim=hsim*0.05

In [110]: hcum=sts.maxwell.cdf(xx,loc=0.,scale=1.00187659)
```





Pruebas de hipótesis:

Es usual tener la necesidad de comparar nuestras muestras para tomar decisiones:

- comparar dos muestras nuestras entre sí.
- comparar nuestra muestra con la de otro investigador.
- comparar nuestra muestra con algún modelo de la literatura.
- comparar nuestra muestra con nuestro propio modelo.

Pruebas de hipótesis:

	Pruebas paramétricas	Pruebas no Paramétricas
Clásicas	Modelo conocido Pob. conocida Muchos datos Escalas ordinales	Modelo desconocido Distr. errores desconocida Pocos datos Escalas nominales o en categorías
Bayesianas	Modelo conocido Proceso de recolección y errores conocidos	No existen

Pruebas de hipótesis:

- Las pruebas clásicas son las más usadas en diversas aplicaciones.
- Si no conocemos la distribución, tenemos datos anómalos o muy pocos datos se requiere el uso de métodos clásicos no paramétricos.
- Las pruebas clásicas tratan de rechazar una hipótesis a un cierto nivel de significancia (es un proceso de eliminación).
- Las pruebas bayesianas son prácticas cuando se puede modelar el proceso de recolección de datos.
- Una prueba clásica trabaja con la distribución de probabilidades de un estadístico, mientras que una bayesiana trabaja con distribución de probabilidades de una hipótesis.

Pruebas de hipótesis:

La metodología tradicional es la siguiente:

- Se obtiene la o las muestras a cotejar.
- Se define una hipótesis nula H_0 (por ejemplo, "El valor medio de la población es menor que el de la muestra") y una hipótesis alternativa H_1 opuesta ("El valor medio de la población es mayor o igual al de la muestra").
- Se elije una prueba a realizar y se decide **a priori** el nivel de significancia α.
- Al elegir la prueba se determina una distribución de muestreo en función de un estadístico de prueba.

Pruebas de hipótesis:

- Se encuentra la región de rechazo que es una fracción α del área total de la distribución de muestreo.
- Al efectuar la prueba, H_0 se rechaza si se obtiene para el estadístico de prueba una probabilidad de ocurrencia menor que α.
- Si H_0 es rechazado **no quiere decir** que H_1 es automáticamente aceptado. El proceso sirve solo para rechazar H_0, no para probar H_1.
- Los valores usados habitualmente para α son 0,10; 0,05 o 0,01.

Pruebas de hipótesis:

También se utiliza para estimar una prueba de hipótesis al denominado valor de probabilidad o valor p. El procedimiento es el siguiente:

- Se define una hipótesis nula H_0 donde se asume que el resultado que se quiere comprobar no es real (por ejemplo, "los dos valores medios no son iguales" cuando uno sospecha que si lo son), y una hipótesis alternativa opuesta, H_1.
- Se define un estadístico de prueba y se compara con una distribución de probabilidades.
- El valor p es la probabilidad de observar el valor de la prueba si H_0 es verdadera o, visto de otra manera, evalúa la evidencia en contra de esa hipótesis.

Pruebas de hipótesis:

- Si el valor p es menor que un cierto nivel de significancia α fijado por el usuario **a priori** se dice que el resultado es estadísticamente significativo y H_0 se debe rechazar.
- Que un resultado sea estadísticamente significativo quiere decir que es improbable que suceda por azar.
- Un valor p menor a 0,05 indica una fuerte evidencia en contra de H_0 porque habría una probabilidad menor al 5 % que H_0 sea correcta y nos lleva a aceptar H_1.
- Un valor p superior a 0,05 indica una fuerte evidencia a favor de H_0 y al rechazo de H_1.

Pruebas de hipótesis:

Al efectuar una prueba de hipótesis se pueden cometer dos tipos de errores:

- Errores de tipo I, son aquellos en los que se rechaza H_0 siendo esta verdadera:

$$\alpha = p(\textit{rechazar } H_0 | H_0 \textit{ es verdadera})$$

- Errores de tipo II, son aquellos en los que no se rechaza H_0 siendo esta hipótesis falsa:

$$\beta = p(\textit{aceptar } H_0 | H_0 \textit{ es falsa})$$

Además, se define la Potencia del test como la probabilidad de detectar que H_0 es falsa:

$$(1 - \beta) = p(\textit{rechazar } H_0 | H_0 \textit{ es falsa})$$

Usualmente se evalúa solo el error de tipo I fijando un nivel de significancia α.

Pruebas de hipótesis:

Por otra parte, al fijar un nivel de significancia podemos tener pruebas de una o dos colas, dependiendo de lo que se quiera probar.

Por ejemplo, si decidimos que H_0 corresponde a "el valor medio de la población es igual a un cierto valor μ_0" y **se fija** $\alpha = 0{,}05$, tenemos que H_1 puede corresponder a $\mu > \mu_0$, $\mu < \mu_0$ y $\mu \neq \mu_0$, produciendo pruebas de hipótesis de una cola por arriba, de una cola por abajo y de dos colas, respectivamente.

Pruebas de hipótesis:

Estadístico de prueba con metodología tradicional:

Para una muestra de una variable aleatoria tomada de una distribución normal con media μ y varianza σ^2 ($N(\mu;\sigma^2)$) se quiere verificar H_0 ($\mu = \mu_0$) contra H_1 ($\mu > \mu_0$).
Si la varianza es conocida, el estadístico de prueba es $Z = \frac{\bar{x}-\mu_0}{\sigma/\sqrt{n}}$ que se distribuye como $N(0;1)$.
Por ejemplo, si $n = 100$, $\mu_0 = 24$, $\sigma^2 = 9$, y considerando que $\alpha = 0{,}05$:

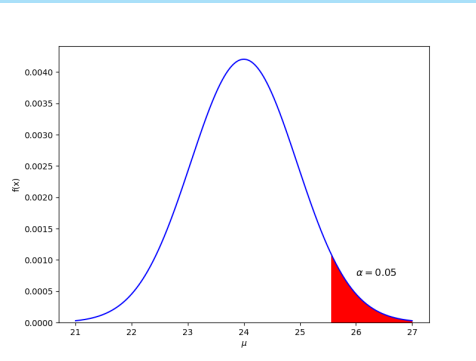
$$p(\bar{x} \leq c) = \frac{c - \mu_0}{\sqrt{0{,}9}} = 1 - \alpha = 0{,}95$$

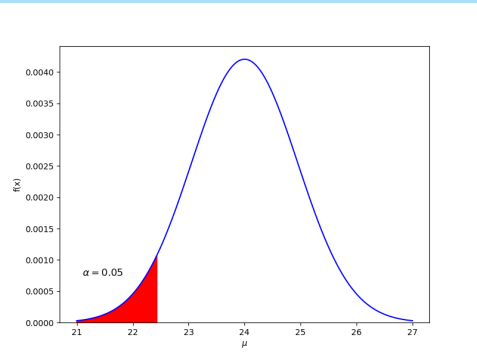
que para $N(0;1)$ corresponde a $Z = 1{,}645$ y entonces $c = 1{,}645 * \sqrt{0{,}9} + \mu_0 = 25{,}56$; por lo cual si $\bar{x} > 25{,}56$ se rechaza H_0.

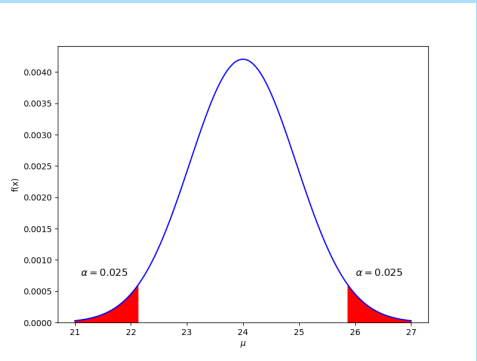
Pruebas de hipótesis:

Estadístico de prueba con metodología tradicional:

```
In [118]: sts.norm.ppf(0.95,loc=0,scale=1)
Out[118]: 1.6448536269514722

In [119]: sts.norm.ppf(0.05,loc=0,scale=1)
Out[119]: -1.6448536269514729

In [120]: sts.norm.ppf(0.975,loc=0,scale=1)
Out[120]: 1.959963984540054

In [121]: sts.norm.ppf(0.025,loc=0,scale=1)
Out[121]: -1.9599639845400545
```







Pruebas de hipótesis:

Ejemplo: Comparación de varianzas

Tenemos dos muestras, con $N_1 = 8$ y $N_2 = 7$ elementos, tomadas de distribuciones normales con varianzas $\sigma_1^2 = 106$ y $\sigma_2^2 = 84$, respectivamente, y queremos comprobar la hipótesis H_0 "las varianzas de ambas poblaciones son iguales" contra la hipótesis alternativa H_1 "la varianza de la población 1 es mayor que la correspondiente a la población 2".
En Python no existe esta prueba de hipótesis por lo que debemos construirla manualmente trabajando a partir de una distribución F.

Pruebas de hipótesis:

Proceso:

1. Calcular las varianzas de ambas muestras.
2. Fijar el nivel de significancia (por ejemplo, $\alpha = 0{,}05$).
3. Calcular el estadístico de prueba $v = \sigma_1^2/\sigma_2^2$.
4. Determinar el valor crítico C a partir de $p(V \leq C) = 1 - \alpha$ usando una distribución F con $(N_1 - 1; N_2 - 1)$ grados de libertad.
5. Si $v \leq C$, H_0 no se rechaza.

```
In [42]: v=106./84.

In [43]: v
Out[43]: 1.2619047619047619

In [44]: sts.f.ppf(0.95,7,6)
Out[44]: 4.2066584878692055
```

Pruebas paramétricas en Python:

- scipy.stats.ttest_1samp()
- scipy.stats.ttest_ind()
- scipy.stats.ttest_ind_from_stats()
- scipy.stats.ttest_rel()
- scipy.stats.f_oneway()
- scipy.stats.bartlett()
- scipy.stats.levene()

Pruebas paramétricas:

- scipy.stats.ttest_1samp(): Prueba T de Student de dos colas para compara la media de la muestra con la de la población, asumiendo distribución normal. La H_0 es "la media de la muestra es igual a la de la población":

```
In [56]: datos=sts.norm.rvs(loc=10,scale=5.,size=(50,))

In [57]: sts.ttest_1samp(datos,10)
Out[57]: Ttest_1sampResult(statistic=0.06686317970631839, pvalue=0.9469627058390
672)
```

Pruebas paramétricas:

- scipy.stats.ttest_ind(): Prueba T de Student de dos colas donde se comparan las medias de dos muestras y se asumen distribuciones normales. La H_0 es "las medias de dos muestras independientes son iguales". Si se indica que las varianzas de las poblaciones no son iguales se realiza una prueba de Welsh para resolver la hipótesis.

```
In [64]: datos1=sts.norm.rvs(loc=10,scale=5.,size=(100,))

In [65]: datos2=sts.norm.rvs(loc=10,scale=10,size=(100,))

In [66]: sts.ttest_ind(datos1,datos2,equal_var=False)
Out[66]: Ttest_indResult(statistic=-1.8080333861334814, pvalue=0.072679020915735
85)
```

Pruebas paramétricas:

- scipy.stats.ttest_ind_from_stats(): Prueba T de Student de dos colas donde se conocen $\bar{x}$ y s para dos muestras y se asumen distribuciones normales. La H_0 es "las medias de dos muestras independientes son iguales":

```
In [68]: sts.ttest_ind_from_stats(mean1=10.,std1=15.,nobs1=30,mean2=12.,std2=10.
    ...: ,nobs2=20)
Out[68]: Ttest_indResult(statistic=-0.5229453222747296, pvalue=0.603418025993309
8)
```

Pruebas paramétricas:

- scipy.stats.ttest_rel(): Prueba T de Student de dos colas donde dos muestras independientes están emparejadas una a una y la H_0 es "las medias de las dos muestras son iguales". Se asumen distribuciones normales e igual número de elementos:

```
In [84]: datos1=sts.norm.rvs(loc=5,scale=10,size=(100,))

In [85]: datos2=(sts.norm.rvs(loc=8,scale=10,size=(100,))+sts.norm.rvs(scale=0.5
    ...: ,size=(100,)))

In [86]: sts.ttest_rel(datos1,datos2)
Out[86]: Ttest_relResult(statistic=-3.2642513187272173, pvalue=0.001507471303923
9184)
```

Pruebas paramétricas:

- scipy.stats.f_oneway(): Prueba de Análisis de la Varianza de una vía de clasificación (ANOVA) para varias muestras independientes con igual varianza, donde H_0 es "las muestras provienen de poblaciones con igual media". Se asumen distribuciones normales:

```
In [40]: datos1=sts.norm.rvs(loc=5,scale=10,size=(10,))

In [41]: datos2=sts.norm.rvs(loc=4,scale=10,size=(20,))

In [42]: sts.f_oneway(datos1,datos2)
Out[42]: F_onewayResult(statistic=5.044115968392863, pvalue=0.0327850868808436)
```

Pruebas paramétricas:

- scipy.stats.bartlett(): Es una prueba paramétrica para verificar varianzas iguales en las poblaciones de donde se obtuvieron las muestras. Para poblaciones significativamente no normales scipy.stats.levene() es más robusto.
  En esta prueba H_0 es "las muestras provienen de poblaciones con varianzas iguales":

```
In [196]: datos1=sts.norm.rvs(loc=1.,scale=2.,size=100)

In [197]: datos2=sts.norm.rvs(loc=1.,scale=3.,size=100)

In [198]: sts.bartlett(datos1,datos2)
Out[198]: BartlettResult(statistic=9.477404513885897, pvalue=0.002080180576855118)
```

Pruebas paramétricas:

- scipy.stats.levene(): Es una prueba paramétrica para verificar varianzas iguales en las poblaciones de donde se obtuvieron las muestras. Es una prueba más robusta que scipy.stats.bartlett() para poblaciones que se desvían de una normal. Hay tres modos de hacer la prueba que son 'median', 'mean' y 'trimmed' que son recomendados para poblaciones no normales, poblaciones simétricas y poblaciones con colas, respectivamente.
  En esta prueba H_0 es "las muestras provienen de poblaciones con varianzas iguales":

```
In [200]: datos1=sts.norm.rvs(loc=1.,scale=2.,size=100)

In [201]: datos2=sts.norm.rvs(loc=1.,scale=3.,size=100)

In [202]: datos3=sts.maxwell.rvs(loc=0.,scale=2.,size=100)

In [203]: sts.levene(datos1,datos2,center='mean')
Out[203]: LeveneResult(statistic=8.927057619613695, pvalue=0.003165169514914902)

In [204]: sts.levene(datos1,datos3,center='median')
Out[204]: LeveneResult(statistic=15.068414179429766, pvalue=0.00014119308326542316)
```

Pruebas no paramétricas en Python:

- scipy.stats.chisquare()
- scipy.stats.kstest()
- scipy.stats.ks_2samp()
- scipy.stats.mannwhitneyu()
- scipy.stats.ranksums()
- scipy.stats.wilcoxon()
- scipy.stats.kruskal()
- scipy.stats.ansari()
- scipy.stats.fligner()
- scipy.stats.mood()

Pruebas no paramétricas:

- scipy.stats.chisquare(): Compara datos de una muestra que se pueden agrupar en intervalos de clase con valores obtenidos de un modelo arbitrario. Es un test de bondad de ajuste basado en una distribución χ^2.
  En este caso H_0 es "el número de elementos en cada intervalo de clase coincide con el del modelo". En más del 80 % de los intervalos de clase debe haber al menos 5 elementos:

```
In [54]: datos=[6,9,13,15,15,11,7]

In [55]: frec=[7,11,14,16,15,10,6]

In [56]: sts.chisquare(datos,f_exp=frec)
Out[56]: Power_divergenceResult(statistic=0.9070887445887446, pvalue=0.988890448
3258015)
```

  Si no se incluye las frecuencias de la población en 'f_exp' se asume una distribución uniforme igual al valor medio de la muestra.

Pruebas no paramétricas:

- scipy.stats.kstest(): Prueba de bondad de ajuste de Kolmogorov-Smirnov para una muestra. Compara la distribución acumulativa de una muestra con la correspondiente a una distribución contínua y estima la mayor desviación entre ellas.
  La H_0 es "las dos distribuciones acumulativas son iguales". Este test permite trabajar con una H_1 de una cola por arriba, una cola por abajo y dos colas:

```
In [92]: datos=sts.norm.rvs(size=100)

In [93]: sts.kstest(datos,'norm')
Out[93]: KstestResult(statistic=0.057630162229666215, pvalue=0.8940216471182763)

In [94]: sts.kstest(datos,'norm',alternative='less')
Out[94]: KstestResult(statistic=0.057630162229666215, pvalue=0.4957557985103209)

In [95]: sts.kstest(datos,'laplace')
Out[95]: KstestResult(statistic=0.07023860063483306, pvalue=0.712288564042646)

In [96]: sts.kstest(datos,'maxwell')
Out[96]: KstestResult(statistic=0.6899875474818046, pvalue=0.0)
```

Pruebas no paramétricas:

- scipy.stats.ks_2samp(): Prueba de bondad de ajuste de Kolmogorov-Smirnov para dos muestras. Compara las distribuciones acumulativas de ambas muestras asumiendo que fueron adquiridas de una población cuya distribución es contínua y estima la mayor desviación entre ellas.
  La H_0 es "las dos muestras fueron adquiridas de la misma población con una distribución contínua". Este test permite trabajar solo con una H_1 de dos colas:

```
In [99]: datos1=sts.norm.rvs(loc=1., scale=2, size=200)

In [100]: datos2=sts.norm.rvs(loc=1.5, scale=1.5, size=300)

In [101]: sts.ks_2samp(datos1,datos2)
Out[101]: Ks_2sampResult(statistic=0.14500000000000002, pvalue=0.011408832695404
68)

In [102]: datos3=sts.norm.rvs(loc=1.1,scale=2.,size=300)

In [103]: sts.ks_2samp(datos1,datos3)
Out[103]: Ks_2sampResult(statistic=0.07166666666666666, pvalue=0.553298670180372
2)
```

Pruebas no paramétricas:

- scipy.stats.mannwhitneyu(): Prueba de rango de Mann-Whitney para dos muestras o Prueba U: se ordena el conjunto de ambas muestras y sumar los índices de la menor de ellas para usar como el estadístico de prueba U. Las muestras deben tener más de 20 elementos debido a la aproximación usada.
  La H_0 es "las dos muestras tienen la misma distribución continua" contra la H_1 de que sean diferentes o que difieran por una traslación (dos o una cola):

```
In [108]: datos1=sts.norm.rvs(loc=1., scale=2, size=200)

In [109]: datos2=sts.norm.rvs(loc=1.1,scale=2.,size=300)

In [110]: sts.mannwhitneyu(datos1,datos2,alternative='two-sided')
Out[110]: MannwhitneyuResult(statistic=25751.0, pvalue=0.0072680906119697375)

In [111]: sts.mannwhitneyu(datos1,datos2,alternative='less')
Out[111]: MannwhitneyuResult(statistic=25751.0, pvalue=0.0036340453059848688)

In [112]: sts.mannwhitneyu(datos1,datos2,alternative='greater')
Out[112]: MannwhitneyuResult(statistic=25751.0, pvalue=0.9963728172465885)
```

Pruebas no paramétricas:

- scipy.stats.ranksums(): Prueba de rango de Mann-Whitney para dos muestras. El procedimiento es igual al empleado en scipy.stats.mannwhitneyu() pero no hay limitaciones respecto del número de elementos, pero no distingue entre elementos iguales en las muestras.
  La H_0 es "las dos muestras tienen la misma distribución continua" contra la H_1 de que difieran por una traslación:

```
In [153]: a1
Out[153]: [20, 37, 55, 50, 64, 41]

In [154]: b1
Out[154]: [75, 21, 72, 71, 85, 43, 34, 65, 90, 35]

In [155]: sts.ranksums(a1,b1)
Out[155]: RanksumsResult(statistic=-1.301582746911937, pvalue=0.19305906087643554)
```

Pruebas no paramétricas:

- scipy.stats.wilcoxon(): Prueba de rango con signo de Wilcoxon para dos muestras apareadas. Comprueba si las diferencias entre los elementos de ambas muestras son simétricas respecto de cero. Las muestras deben tener más de 20 elementos debido a la aproximación usada. La H_0 es "las dos muestras son iguales" contra la H_1 de que sean diferentes. Si se ingresa una sola muestra se asume que contiene las diferencias entre las dos muestras.

```
In [139]: datos1
Out[139]: array([10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10])

In [140]: datos2
Out[140]: array([ 8, 10, 10,  9,  8,  8,  7, 13,  9,  8,  7, 10, 11,  9, 12,  9])

In [141]: dif=datos1-datos2

In [142]: sts.wilcoxon(datos1,datos2)
Out[142]: WilcoxonResult(statistic=23.0, pvalue=0.11094011568029877)

In [143]: sts.wilcoxon(dif)
Out[143]: WilcoxonResult(statistic=23.0, pvalue=0.11094011568029877)
```

Pruebas no paramétricas:

- scipy.stats.kruskal(): Prueba de Kruskal-Wallis o prueba H para dos o más muestras. El procedimiento compara las medianas de las muestras, las cuales deben tener al menos 5 elementos. Es una versión no paramétrica de ANOVA. La H_0 es "las medianas de las muestras son iguales" contra la H_1 de que difieran entre si:

```
In [172]: datos1=sts.norm.rvs(loc=1.,scale=2.,size=20)

In [173]: datos2=sts.norm.rvs(loc=2,scale=2.5,size=10)

In [174]: datos3=sts.norm.rvs(loc=4.,scale=1.5,size=10)

In [175]: sts.kruskal(datos1,datos2)
Out[175]: KruskalResult(statistic=1.1148387096774144, pvalue=0.291032533885933)

In [176]: sts.kruskal(datos1,datos3)
Out[176]: KruskalResult(statistic=7.681935483870959, pvalue=0.005577631440258802)
```

Pruebas no paramétricas:

- scipy.stats.ansari(): Es la prueba de Ansari-Bradley para dos muestras. Comprueba si los paŕametros de escala de las distribuciones de donde se obtienen las muestras son iguales.
  En este caso H_0 es "las muestras provienen de poblaciones con escalas iguales":

```
In [211]: datos1=sts.norm.rvs(loc=1.,scale=2.,size=100)

In [212]: datos2=sts.norm.rvs(loc=0.,scale=3.,size=100)

In [213]: sts.ansari(datos1,datos2)
Out[213]: AnsariResult(statistic=5535.0, pvalue=0.0177796143664449 26)
```

Pruebas no paramétricas:

- scipy.stats.fligner(): Es la prueba de Fligner-Killeen para varias muestras. Comprueba si las varianzas de las distribuciones de donde se obtienen las muestras son iguales. Hay tres modos de hacer la prueba que son similares a las opciones indicadas en scipy.stats.levene(). La H_0 es "las muestras provienen de poblaciones con varianzas iguales":

```
In [219]: datos1=sts.norm.rvs(loc=1.,scale=2.,size=100)

In [220]: datos2=sts.norm.rvs(loc=0.,scale=3.,size=100)

In [221]: datos3=sts.norm.rvs(loc=2.,scale=2.5,size=100)

In [222]: sts.fligner(datos1,datos2,datos3,center='mean')
Out[222]: FlignerResult(statistic=5.901577477169236, pvalue=0.0522984398790532)

In [223]: sts.fligner(datos1,datos2,datos3,center='median')
Out[223]: FlignerResult(statistic=5.829206524242981, pvalue=0.05422554009169919)
```

Pruebas no paramétricas:

- scipy.stats.mood(): Es la prueba de Mood para dos muestras. Comprueba si los parámetros de escala de las distribuciones de donde se obtienen las muestras son iguales.
  Aquí H_0 es "las muestras provienen de poblaciones con escalas iguales":

```
In [226]: datos1=sts.norm.rvs(loc=1.,scale=2.,size=100)

In [227]: datos2=sts.norm.rvs(loc=0.,scale=3.,size=100)

In [228]: datos3=sts.norm.rvs(loc=2.,scale=2.,size=100)

In [229]: sts.mood(datos1,datos2)
Out[229]: (-4.2336708397048834, 2.2990738080205764e-05)

In [230]: sts.mood(datos1,datos3)
Out[230]: (-0.2536265613914764, 0.7997840744256501)
```

Otras pruebas de hipótesis en Python:

- scipy.stats.shapiro()
- scipy.stats.anderson()
- scipy.stats.anderson_ksamp()
- scipy.stats.jarque_bera()
- scipy.stats.normaltest()

Otras pruebas de hipótesis en Python:

- scipy.stats.shapiro(): Es la prueba de Shapiro-Wilk que comprueba si la muestra fue tomada de una distribución normal.
  La H_0 es "la muestra proviene de una distribución normal":

```
In [234]: datos1=sts.norm.rvs(loc=1.,scale=2.,size=100)

In [235]: datos2=sts.maxwell.rvs(loc=1.,scale=2.,size=100)

In [236]: sts.shapiro(datos1)
Out[236]: (0.9872949123382568, 0.45775875449180603)

In [237]: sts.shapiro(datos2)
Out[237]: (0.975009024143219, 0.054044511169195175)
```

Otras pruebas de hipótesis en Python:

- scipy.stats.anderson(): Es la prueba de Anderson-Darling que comprueba si la muestra fue tomada de una distribución particular. Permite pruebas contra diferentes tipos de distribuciones. La función devuelve el estadístico de la prueba y los valores críticos para diferentes niveles de significancia.
  En este caso, H_0 es "la muestra proviene de la distribución particular que se prueba":

```
In [246]: datos1=sts.norm.rvs(loc=0.,scale=1.,size=100)

In [247]: datos2=sts.maxwell.rvs(loc=1.,scale=2.,size=100)

In [248]: sts.anderson(datos1,'norm')
Out[248]: AndersonResult(statistic=0.6516358837367022, critical_values=array([0.555,
 0.632, 0.759, 0.885, 1.053]), significance_level=array([15. , 10. ,  5. ,  2.5,  1.
 ]))

In [249]: sts.anderson(datos2,'norm')
Out[249]: AndersonResult(statistic=0.5497273667601092, critical_values=array([0.555,
 0.632, 0.759, 0.885, 1.053]), significance_level=array([15. , 10. ,  5. ,  2.5,  1.
 ]))
```

Otras pruebas de hipótesis en Python:

- scipy.stats.anderson_ksamp(): Es la prueba de Anderson-Darling para múltiples muestras. Comprueba si las muestras fueron tomadas de la misma distribución sin especificar cual es. La función devuelve el estadístico de la prueba, los valores críticos para diferentes niveles de significancia y el nivel de significancia aproximado para el cual H_0 es rechazado.
  La H_0 es "las muestras provienen de la misma distribución":

```
In [256]: datos1=sts.norm.rvs(loc=0.,scale=1.,size=100)

In [257]: datos2=sts.norm.rvs(loc=0.5,scale=1.5,size=100)

In [258]: datos3=sts.norm.rvs(loc=0.2,size=100)

In [259]: sts.anderson_ksamp([datos1,datos2,datos3])
Out[259]: Anderson_ksampResult(statistic=2.2470445981404055, critical_values=array([
0.44925884, 1.3052767 , 1.9434184 , 2.57696569, 3.41634856]), significance_level=0.0
35932172916725395)
```

Otras pruebas de hipótesis en Python:

- scipy.stats.jarque_bera(): Es una prueba de bondad de ajuste para una muestra que comprueba si la kurtosis y sesgo se ajusta a una distribución normal. Como la prueba está basado en una distribución χ^2 se requiere que la muestra contenga más de 2000 elementos.
  H_0 es "la kurtosis y sesgo de la muestra se ajusta a una distribución normal":

```
In [180]: datos1=sts.norm.rvs(size=10000)

In [181]: datos2=sts.maxwell.rvs(size=10000)

In [182]: sts.jarque_bera(datos1)
Out[182]: (3.695873501256187 4, 0.1575619207077078)

In [183]: sts.jarque_bera(datos2)
Out[183]: (330.7827347367894, 0.0)
```

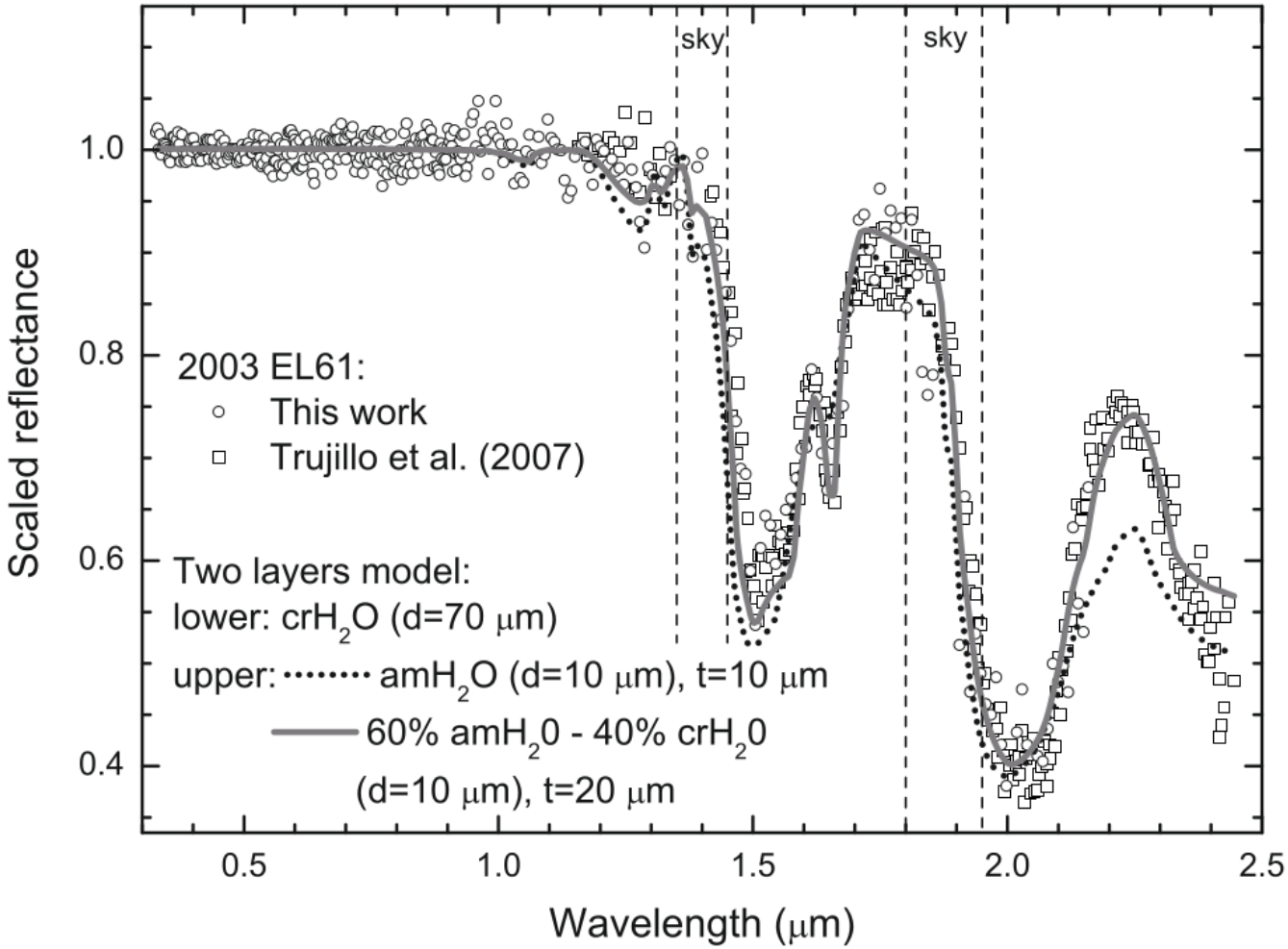
Otras pruebas de hipótesis en Python:

- scipy.stats.normaltest(): Prueba de D'Agostino para ver si una muestra de más de 20 elementos difiere de una distribución normal considerando los valores de la kurtosis y el sesgo para generar el test.
  La H_0 es "la muestra proviene de una distribución normal":

```
In [48]: datos1=sts.norm.rvs(loc=5,scale=10,size=(30,))

In [49]: sts.normaltest(datos1)
Out[49]: NormaltestResult(statistic=0.29675464279131825, pvalue=0.8621057626151527)
```

Otras pruebas:

- **Prueba de aleatoriedad de una muestra:**



Otras pruebas posibles:

- **Prueba de aleatoriedad de una muestra (cont.):**
  Esta prueba simple permite estimar el grado de aleatoriedad de una muestra en referencia a un valor medio, una función de ajuste, etc.
  La prueba primero determina cuantos valores estan por arriba (Ma) y por debajo (Md) de un valor de referencia, siendo $N = Ma + Md$. Con estos valores se encuentra el número de corridas de valores similares, r, que se usa como estadístico de prueba.
  Por ejemplo, si se indica con "+" y "-" valores por arriba o debajo del valor de referencia, respectivamente, una serie "$++--+-+++--+--+--$" tendría $Ma = 8$, $Md = 10$ y $r = 10$.
  En esta prueba la H_0 sería "la muestra es al azar" y la prueba permite pruebas de una y dos colas.

Otras pruebas posibles:

- **Prueba de aleatoriedad de una muestra (cont.):** Para resolver la prueba se recurre a la función de probabilidades:

$$f(R) = 2 \left(\begin{array}{c} \text{Ma - 1} \\ \text{R/2 - 1} \end{array} \right)^2 / \left(\begin{array}{c} \text{2Ma} \\ \text{Ma} \end{array} \right)$$

para un r par, y:

$$f(R) = 2 \left(\begin{array}{c} \text{Ma - 1} \\ \text{R/2 - 1/2} \end{array} \right) \left(\begin{array}{c} \text{Ma - 1} \\ \text{R/2 - 3/2} \end{array} \right) / \left(\begin{array}{c} \text{2Ma} \\ \text{Ma} \end{array} \right)$$

para r impar.

Otras pruebas posibles:

- **Prueba de aleatoriedad de una muestra (cont.):** Estos valores están listados en tablas de probabilidades ya calculadas. Para el caso de un número de elementos $N > 20$ es posible utilizar una distribución normal para la variable:

$$Z = \frac{R - \mu_r}{\sigma_r}$$

con media:

$$\mu_r = \frac{2MaMd}{Ma + Md} + 1$$

y varianza:

$$\sigma_r^2 = \frac{2MaMd(2MaMd - N)}{N^2(N-1)}$$

Actividades:

Práctica 7:

- Si se extraen muestras de quasares del survey de Parkes (Masson & Wall 1977) seleccionándolos según si su espectro en radio es plano o no, se obtienen los siguientes valores:
  flat=0,1,0,0,1,0,1,8,7,11,6,7,5,10,6,5,3,0
  noflat=0,0,0,0,0,1,0,0,2,2,1,3,9,4,5,4,3,0
  para sus respectivos histogramas que corresponden a bines de 0.50 de ancho centrados en magnitudes que van desde 12.75 hasta 21.25. Si asumimos que H_0 es "ambas muestras tienen la misma distribución" contra H_1 "las muestras están desplazadas", decida que prueba de hipótesis no paramétrica utilizaría e indique a que conclusión llega.

Actividades:

Práctica 7 (cont.):

- Repita el ejercicio anterior utilizando una prueba no paramétrica diferente. Compare los resultados.
- Agrege a la muestra de quasares con espectro no plano 10 elementos más al azar utilizando un generador arbitrario y repita la prueba. Compare los resultados.
- Desplace la muestra de quasares con espectro no plano media magnitud hacia valores más bajos y repita la prueba. Comente sus conclusiones.

Actividades:

Práctica 7 (cont.):

- Supongamos que adquirimos una muestra con los siguientes valores:
  [44, 23, 77, 38, 68, 41, 98, 5, 27, 74, 36, 33, 80, 27, 74, 39, 34, 33, 59, 55]
  y queremos comprobar la hipótesis H_0 "la mediana de la población de donde se obtuvo es de 30". Elija una prueba de hipótesis y comente los resultados.
- Utilizando la muestra del ejercicio anterior compruebe el grado de aleatoriedad respecto de un valor medio de 40 para H_0: "la muestra es aleatoria".

Actividades:

Práctica 7 (cont.):

- Utilizando una de las variables de su archivo de datos como población, tome dos muestras al azar de 100 elementos cada una y compruebe que pertenecen a la mismma población.

Entrega

Para la próxima clase

Por consultas: ricardo.gil-hutton@conicet.gov.ar
Grupo de Ciencias Planetarias - CUIM 2