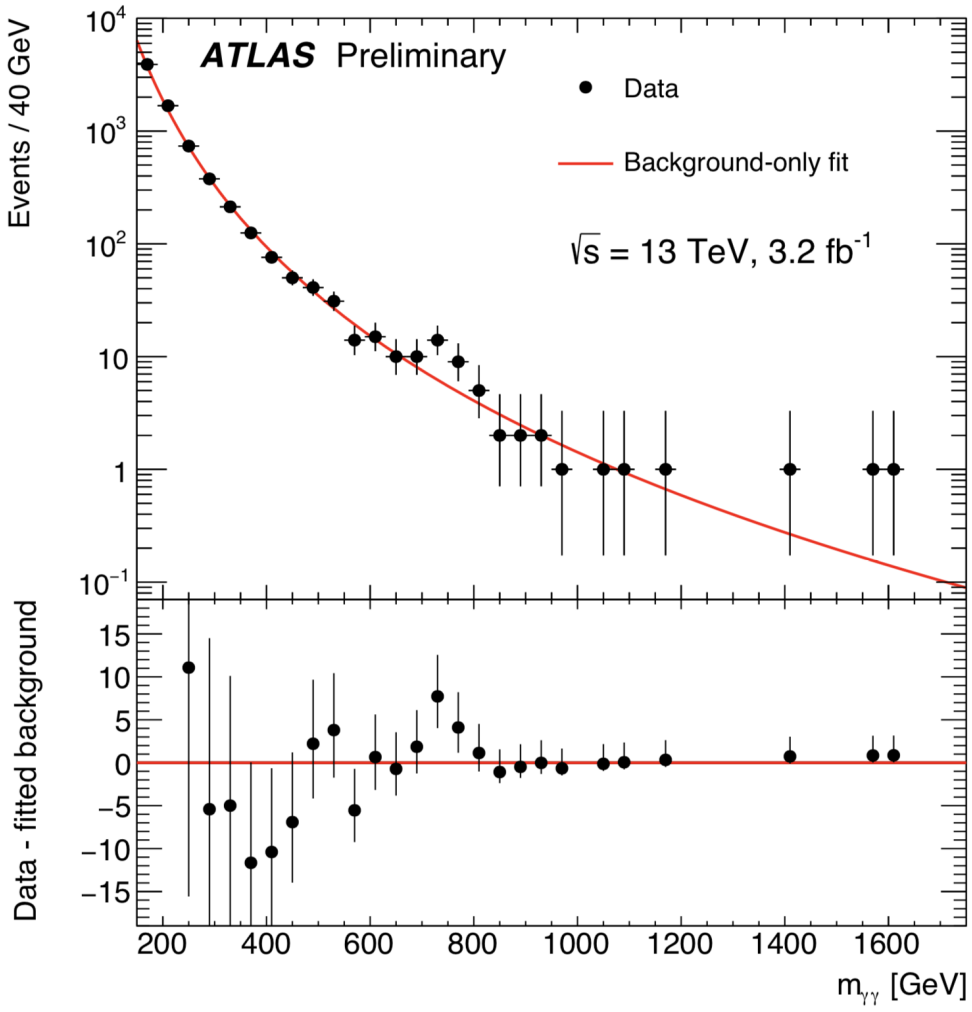
In [4]:
```
%matplotlib inline

import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import warnings
warnings.filterwarnings('ignore')
from pylab import rcParams
rcParams['figure.figsize'] = 12, 9
```

Статистика для физиков

Основными задачами экспериментаторов являются измерения различного рода величин и открытие новых феноменов. При этом всегда возникает необходимость оценить правильность результата, и как он согласуется с теорией.

Пример из [статьи (http://inspirehep.net/record/1410174/files/ATLAS-CONF-2015-081.pdf?version=1)](http://inspirehep.net/record/1410174/files/ATLAS-CONF-2015-081.pdf?version=1)



Какова вероятность, что выброс порожден обычным фоновым процессом, а не тем процессом, что нам интересен (новой физикой)?

Теория вероятности

Вероятность

Что такое вероятность? На данный вопрос довольно сложно дать однозначный ответ. Выделяют несколько трактовок этого понятия:

- частотный подход (Frequentist) - процентная доля интересуемых событий

$$p = \lim_{N \to \infty} \frac{n}{N}$$

- субъективный (Bayesian) - степень уверенности в чем-то

> Он станет президентом с вероятностью 60%

Законы теории вероятности

Несмотря на разные трактовки, законые теории вероятности одни и те же. Стоит отметить, что язык, которым пользуются физики значительно отличается от языка, которым пользуются статистики.

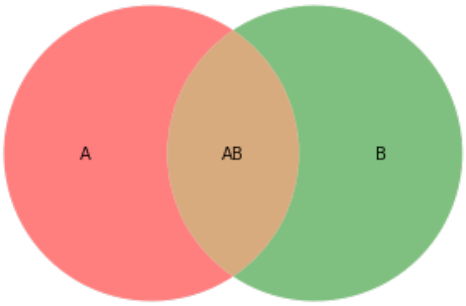
> **Событие** - физики под этим обычно подразумевают результат эксперимента, у которого может быть несколько исходов. Как пример, бросок кости или столкновение двух протонов в коллайдере. Статистики под событием обычно подразумевают подмножество в простанстве всех возможных значений (сумма двух кубов больше 5 или $p_T(\gamma) > 40GeV$).

Пусть **A** и **B** некоторые события. Тогда будем обозначать

- $A \cup B$ или $A + B$ как наступление события либо A, либо B, либо A и B одновременно
- $A \cap B$ или AB как наступление, одновременно, событий A и B

Для любых двух событий A и B верно

$$P(A + B) = P(A) + P(B) - P(AB)$$



И если события **несовместны (взаимно исключают друг друга)**, то есть при наступлении события A, событие B произойти не может

$$P(A + B) = P(A) + P(B)$$

Условная вероятность

Пусть

$$P(A) = \frac{n_A}{N} \quad P(AB) = \frac{n_{AB}}{N}$$
$$P(AB) = \frac{n_A}{N}\frac{n_{AB}}{n_A} = P(A)\frac{n_{AB}}{n_A}$$

Условная вероятность - это вероятность события **B**, при условии, что событие **A** уже произошло

$$P(B|A) = \frac{P(AB)}{P(A)}$$

Отсюда можно получить сразу же следствие. Если на появления события **B** не влияет появление события **A**, то данные события считаются независимыми

$$P(B|A) = P(B)$$
$$P(AB) = P(A)P(B)$$

Формула полной вероятности

Если считать, что множество событий $\{A_i\}_{i=1}^N$ несовместны

$$P(B) = \sum_i P(A_i B) = \sum_i P(A_i)P(B|A_i)$$

данный процесс можно назвать маргинализацией (Marginalisation) и он является довольно частым явление в физике. Нередко нас может интересовать только наступление события $P(B)$. Данной операцией мы можем сосредоточиться только на событие B, усреднив по A.

Теорема Байеса

Если подумать, то мы можем записать $P(AB)$ двумя способами

$$P(AB) = P(A)P(B|A) = P(B)P(A|B)$$

что приводит нас к

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- **P(A)** - априорная вероятность гипотезы или события A
- **P(B)** - полная вероятность события B
- **P(A|B)** - вероятность гипотезы A при наступлении события B (апостериорная вероятность)
- **P(B|A)** - вероятность наступления события B при истинности гипотезы A (правдоподобие)

Например, пусть у нас есть детектор, который умеет регистрировать мюоны с вероятностью $P(+|\mu) = 0.95$, то есть 5% мюонов будут им пропущены. Небольшая доля пионов регистрируется как мюон с вероятностью $P(+|\pi) = 0.05$. Нам нужно посчитать вероятность того, что зарегистрированная частица - это мюон.

Для этого нам нужно знать долю пионов и мюонов в поток частиц. Пусть это будет $P(\mu) = 0.04$ и $P(\pi) = 0.96$.

Теперь по формуле

$$P(\mu|+) = \frac{P(+|\mu)P(\mu)}{P(+)}$$
$$P(+) = P(+|\mu)P(\mu) + P(+|\pi)P(\pi)$$

In [5]:
```
0.95 * 0.04 / (0.95 * 0.04 + 0.05 * 0.96)
```

Out[5]:
```
0.4418604651162791
```

Случайные величины

Пока мы говорили об абстрактной вероятности. Рассмотрим же её теперь подробнее.

> **Случайная величина** - это просто переменная, которая представляется собой численные исходы какого-либо события.

Случайные величины бывают:

- дискретными
- непрерывными

Дискретные случайные величины

Если существет лишь конечное число исходов некого события **A**, то каждому конкретному исходу A_i можно приписать вероятность

$$P(A = A_i) = P_i$$

Множество вероятностей P_i называют дискретным распределением вероятности. И достаточно очевидно, что

$$\sum_i P(A = A_i) = 1$$

Это означает, что вероятность наступления хоть какого-нибудь события равна 1.

Хорошим примером дискретной случайной величины является бросок кубика или вероятность конкретного процесса при столкновении двух протонов в коллайдере.

Непрерывные случайные величины

Если случайная величина может принимать любые значения из некоторого непрерывного интервала значений, то такую величину принято называть непрерывной.

Примером может служить длина комнаты, температура за окном, импульс частицы и т.д.

Здесь у нас возникают небольшие проблемы, ведь если

$$P(X = x) \neq 0$$

тогда сумма по всем x даст нам

$$\sum_x P(X = x) = \infty$$

что не будет иметь смысла.

Обойдем эту проблему следующим образом. Вместо конкретного значения x мы расммотрим вероятность того, что X находится в каком-то малом интервале

$$P(x < X < x + dx) = f(x)dx$$

где $f(x)$ будем называть функцией плотности вероятности (probability density function). **Физики нередко называют это распределением вероятности.**

Отсюда

$$P(x_1 < X < x_2) = \int_{x_1}^{x_2} f(x)dx = F(x_2) - F(x_1)$$

где $F(x)$ - это функция вероятности (Cumulative distribution function)

$$F(x) = \int_{-\infty}^{x} f(x)dx$$

При этом всегда имеет место условие нормировки

$$\int^{\infty} f(x)dx = 1$$

Основные характеристики распределений

- Математическое ожидание или среднее (expected value or mean), которое принято обозначать μ

$$\mu = \langle X \rangle = E[X] = \sum_x xP(X = x), \qquad \text{дискретный случай}$$

$$\mu = \langle X \rangle = E[X] = \int_{-\infty}^{\infty} xf(x), dx \qquad \text{непрерывный случай}$$

- Математическое ожидание или среднее (mean) от некоторой функции случайно величины $h(X)$

$$E[f(X)] = \sum_x h(x)P(X = x), \qquad \text{дискретный случай}$$

$$E[f(X)] = \int_{-\infty}^{\infty} h(x)f(x), dx \qquad \text{непрерывный случай}$$

- Дисперсия (variance) или среднеквадратичная ошибка (mean square error) которую принято обозначать σ^2

$$\sigma^2 = D[x] = Var[x] = E[(x - \mu)^2] = E[x^2] - (E[x])^2$$

> $E[x^2]$ - называют средним квадратическим (root mean square), а σ - стандартным отклонение (standard deviation)

- Ковариация

$$Cov(X, Y) = E[(X - E[X])(Y - E[Y])]$$

- Коэффициент корреляции
- Медиана
- Коэффициент ассиметрии
- и другие

Основные распределения

Биномиальное распределение

Пусть мы проводим последовательно ряд одинаковых экспериментов, в котором результат - это успех или провал (Bernoulli trial). Если p - это вероятность успеха, тогда $q = 1 - p$ - это вероятность провала.

Наример, мы рассматриваем столкновение двух протонов, тогда p - это вероятность того, что событие для нас интересно, q - нам не интересно.

Из этого мы можем записать вероятность получения k успехов при конкретной поледовательности экспериментов S длиною N

$$P(k|p,S,N) = p^k(1-p)^{N-k}$$

Обычно нас не интересует последовательность успехов или провалов, нам именно интересно сколько их. Следовательно, нам нужно проссумировать по всем последовательностям

$$P(k|p,N) = \sum_S P(k|p,S,N) = \frac{N!}{k!(N-k)!}p^k(1-p)^{N-k} = C_N^k p^k(1-p)^{N-k}$$

Это распределение называют биномиальным.

Характеристики бинамиального распределения:

- $\mu = pN$
- $\sigma^2 = p(1-p)N$

Обобщением биномиального распределения является - мультиномиальное распределение. В этом случае вместо двух исходов их становится K

$$P(k_1,k_2,\dots,k_K|p_1,\dots,p_K,N) = \frac{N!}{k_1!k_2!\dots k_K!}p_1^{k_1}p_2^{k_2}\dots p_K^{k_K}$$

$$\sum_{i=1}^{K} k_i = N$$

$$\sum_{i=1}^{K} p_i = 1$$

Распределение Пуассона

А что если у нас число экспериментов достаточно велико? Мы можем тогда получить предельную форму предыдущего распределения в виде

$$P(k|\lambda) = \frac{\lambda^k}{k!}e^{-\lambda}$$

где $\lambda = pN$ - среднее число событий за какой-то интервал.

Обычно распределение Пуассона относят к случаю, когда вам нужно определить некоторое число событий за фиксированный интервал времени. Так вот, число этих событий как раз будет подчинятся распределению Пуассона. В этом смысле λ является средним количеством событий в этом интервале времени.

В нашем случае, под это распределение подходит почти любой эксперимент на LHC. Например количество событий за день или количество частиц в каком-то конкретном процессе.

- $\mu = \lambda$
- $\sigma^2 = \lambda$

Равномерное распределение

Самое простейшее распределение - это равномерное. Для непрерывной случайной величины оно имеет вид

$$f(x|a,b) = \begin{cases} \frac{1}{a-b}, & x \in [a,b) \\ 0, & x \notin [a,b) \end{cases}$$

- $\mu = \frac{a+b}{2}$
- $\sigma^2 = \frac{(b-a)^2}{12}$

Гамма распределение

Обобщение распределения Пуассона на непрерывную случайную величину. При этом $x \geq 0$

$$f(x|k,\theta) = \frac{x^{k-1}}{\theta^k \Gamma(k)} e^{-\frac{x}{\theta}}$$

- $\mu = k\theta$
- $\sigma^2 = k\theta^2$

Экспоненциальное распределение

Довольно важным классом распределений является экспоненциальное. Оно обозначает вероятность наступления события через время t. Например: вероятность распада частицы через время t.

$$f(x|\lambda) = \lambda e^{-\lambda x}$$

- $\mu = \frac{1}{\lambda}$
- $\sigma^2 = \frac{1}{\lambda^2}$

Распределение Коши / Брейта-Вигнера

К необычным распределениям можно отнести распределение Коши

$$f(x|x_0,\gamma) = \frac{1}{\pi\gamma}\frac{\gamma^2}{(x-x_0)^2+\gamma^2}$$

У данного распределения нельзя посчитать среднее значение и дисперсию.

Нормальное распределение

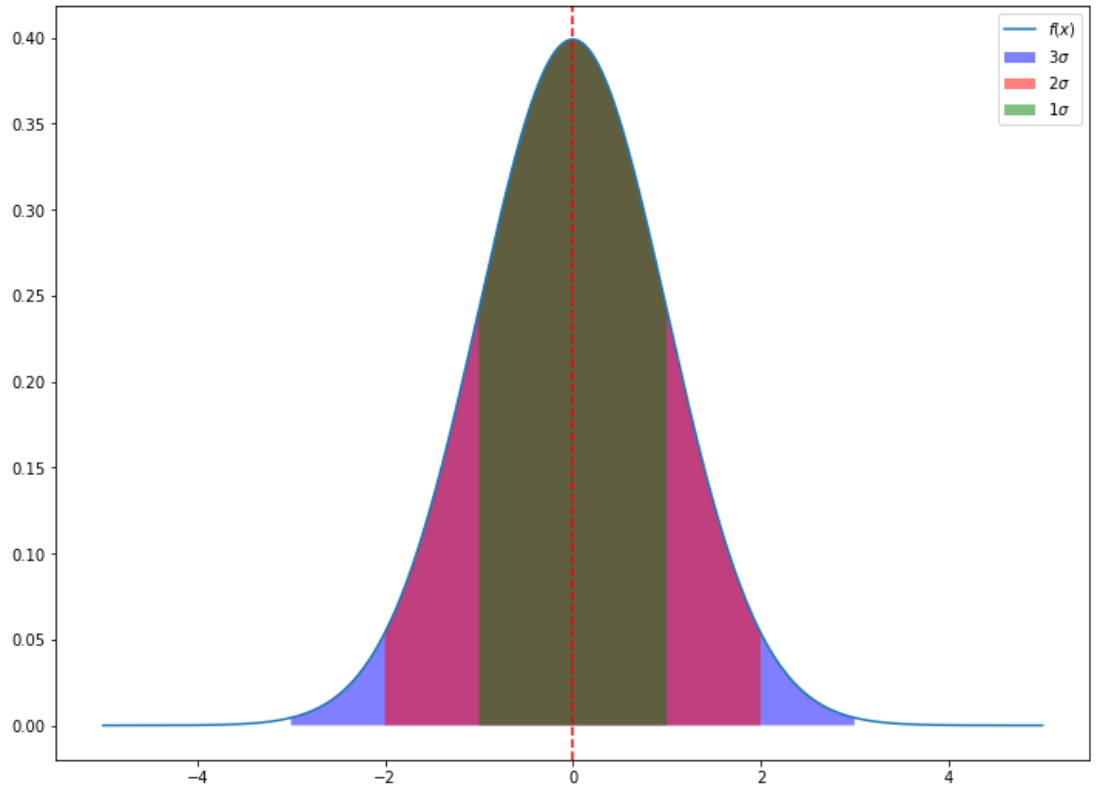
И, наконец, один из самых выжнейших классов распределений

$$f(x|\mu,\sigma^2) = \mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- $\mu = \mu$
- $\sigma^2 = \sigma^2$

In [6]:
```
from scipy.stats import norm

x = np.linspace(-5, 5, 1000)

xs = np.linspace(-3, 3, 100)
plt.fill(np.concatenate([xs, xs[::-1]]),
         np.concatenate([norm.pdf(xs), np.zeros(xs.shape)]),
         alpha=.5, fc='b', label="$3 \sigma$")

xs = np.linspace(-2, 2, 100)
plt.fill(np.concatenate([xs, xs[::-1]]),
         np.concatenate([norm.pdf(xs), np.zeros(xs.shape)]),
         alpha=.5, fc='r', label="$2 \sigma$")

xs = np.linspace(-1, 1, 100)
plt.fill(np.concatenate([xs, xs[::-1]]),
         np.concatenate([norm.pdf(xs), np.zeros(xs.shape)]),
```

```
        alpha=.5, fc='g', label="$1 \sigma$")
plt.axvline(0, c='r', ls='--')
plt.plot(x, norm.pdf(x), label="$f(x)$");
plt.legend()
plt.show()
```



Оно на столько канонично, что иногда вероятности выражают через квантили данного распределения.

> Квантиль - это некое число x_α, значение которого случайная величина не превышает с некоторой наперед заданной вероятностью α
>
> $$P(X \leq x_\alpha) = \alpha$$
> $$F(x_\alpha) = \alpha$$

Квантили нормального распределения обычно смотрять в виде

$$P(\mu - k_\alpha\sigma < X < \mu + k_\alpha\sigma] = \alpha$$

k_α	α
1	68.3%
2	95.4%
3	99.7%
4	$1 - 6.5 \cdot 10^{-3}$ %
5	$1 - 5.7 \cdot 10^{-5}$ %

Для удобства вероятности различного рода величин иногда записывают в количествах σ нормального распределения. При этом сами эти величины могут быть из любого распределения.

$$k_\alpha = \sqrt{2}\,\mathrm{erf}^{-1}(\alpha)$$

Иногда называют это Z-масштаб (Z scale).

In [7]:
```
from scipy.special import erfinv

np.sqrt(2) * erfinv(0.683), np.sqrt(2) * erfinv(0.954), np.sqrt(2) * erfinv(0.9
```

Out[7]:
```
(1.0006418287624492, 1.995393310167825, 2.967737925341772)
```

Центральная предельная теорема

Если не очень строго, то

> Пусть **X** - случайная величина, распределенная по любому закону с конечными средним и дисперсией. Тогда сумма достаточно большого числа этих случайных величин
>
> $$S = \sum_{i=1}^{N} X_i$$
>
> будет вести себя как случайная величина, распределенная по закону $\mathcal{N}(n\mu, n\sigma^2)$

Таким образом, биномиального распределение как сумма случайных событий Бернулли при больших N стремится к нормальному распределению, это же касается распределения Пуассона при больших λ.

χ^2-распределение

Пусть у нас $x_i \sim \mathcal{N}(0, 1)$, тогда их сумма

$$\sum_{i=0}^{K} x_i^2 \sim \chi_K^2$$

где K - число степеней свободы

- $\mu = K$
- $\sigma^2 = 2K$

Статистика

> **Статистика (statistic)** - любая вычислимая функция из выборки данных $X = x_1, x_2, \dots, x_n, \dots, x_N$. При этом физики нередко могут называть статистикой параметризированные функции от выборки данных.

Примерами статистик могут быть:

- выборочный r-й момент
$$m_r = \frac{1}{N} \sum_{n=1}^{N} x_n^r$$
- выборочное среднее
$$\bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n$$
- Несмещенная выборочная дисперсия

$$s^2 = \frac{\sum_{n=1}^{N}(x_n - \hat{x})^2}{N-1}$$

- χ^2
- другие

Достаточно очевидно, что любая статистика является случайной величиной.

Литература

- Идье В., Драйард Д., Джеймс Ф., Рус М., Садуле Б. Статистические методы в экспериментальной физике
- Худсон Д., Статистика для физиков
- Luca Lista, Statistical Methods for Data Analysis in Particle Physics
- https://arxiv.org/abs/1905.12362 (https://arxiv.org/abs/1905.12362)
- http://pdg.lbl.gov/2020/reviews/rpp2020-rev-statistics.pdf (http://pdg.lbl.gov/2020/reviews/rpp2020-rev-statistics.pdf)

Оценка параметров распределений

На практике, вам в лучшем случае известно только семейство функций для распределения. Параметры же распределения для вас будут не известны и их нужно как-то найти.

Большинство стандартных распределений параметризированы 1 или 2 параметрами. На практике, придется работать с распределениями с достаточно большим числом параметров.

Чтобы оценить значения параметров, обычно вводят функцию оценки этих параметров, которая является функцией данных

$$\hat{\vec{\theta}} = t(\mathbf{X})$$

Результат данной функции также будет случайной величиной.

Способов построения данной функций довольно много, и у каждого есть свои преимущества и недостатки. К ним можно отнести метод максимального правдоподобия, методы машинного обучение и т.д.

Правдоподобие

Обычно, мы можем представить один исход эксперимента, как набор случайных величин $\vec{x} = x_1, \dots, x_K$. Здесь $\vec{x}$ - обозначает событие, которые описывается несколькими числовыми характеристиками (возможно взаимосвязанными), например: p_T и заряд.

Распределение для $\vec{x}$ обычно учитывает множество эффектов: теорию, ошибку инструмента и прочее. Эти эффекты учитываются через параметры $\vec{\theta}$ этого распределения, которые изначально обычно неизвестны. Такую функцию плотности вероятности называют правдоподобием

$$L(\vec{\theta}|\vec{x}) = P(\vec{x}|\vec{\theta}) = f(\vec{x}|\vec{\theta})$$

Она показывает вероятность получения $\vec{x}$ при заданных значениях параметров $\vec{\theta}$.

Если наш эксперимент состоит из нескольких последовательных **независимых** исходов $\mathbf{X} = \vec{x}_1, \dots, \vec{x}_N$, распределенных по одному и тому же закону, то мы легко можем записать функцию правдоподобия для этой последовательности:

$$L(\vec{\theta}|\mathbf{X}) = \prod_n P(\vec{x}_n|\vec{\theta})$$

То есть, фактически, функция правдоподобия показывает вероятность получить наши данные

исходя из предположения, что эти данные получены из распределения $L(\vec{\theta}|\mathbf{X})$.

Часто удобно работать с логарифмом функции правдоподобия

$$\mathcal{L}(\vec{\theta}|\mathbf{X}) = \ln L(\vec{\theta}|\vec{x}) = \sum_n \ln P(\vec{x}_n|\vec{\theta})$$

Байесов вывод

Функция правдоподобия идеально вписывается в теорему Байеса. В этом случае, мы считаем параметры распределения случайными величинами с некоторыми априорными вероятностями $P(\vec{\theta}) = \pi(\vec{\theta})$. Тогда сразу можем записать апостериорную вероятность для параметров

$$P(\vec{\theta}|\mathbf{X}) = \frac{P(X|\theta)\pi(\theta)}{P(X)} = \frac{L(\vec{\theta}|\mathbf{X})\pi(\vec{\theta})}{\int L(\vec{\theta}|\mathbf{X})\pi(\vec{\theta})\, d\vec{\theta}}$$

Если априорные вероятности параметров независимы, то $\pi(\vec{\theta}) = \pi_1(\theta_1) \cdot \ldots \cdot \pi_K(\theta_K)$.

В априорную вероятноятность можно закладывать все, что мы знаем о значении данного параметра. Например, параметр может быть только больше нуля:

$$\pi(\theta) = \begin{cases} 1, & \theta \geq 0 \\ 0, & \theta < 0 \end{cases}$$

Так как знаменатель является константой и не зависит от параметров модели, его можно не писать, а в конце просто вывести из условия нормировки

$$P(\vec{\theta}|\mathbf{X}) \sim L(\vec{\theta}|\mathbf{X})\pi(\vec{\theta})$$

Отлично, но как же нам искать значение параметров? У нас есть функция распределения для всех возможных значений параметров, а следовательно, мы можем найти наиболее вероятные значения параметров (это максимум апостериорной функции распределения) и даже построить доверительный интервал.

Интересной особенностью теоремы Байса можно отметить, что мы можем факторизовать априорную вероятность и правдоподобие. Как пример, если мы разобьем последовательность событий $\mathbf{X}$ на три группы $\mathbf{X}_1$, $\mathbf{X}_2$ и $\mathbf{X}_3$, то мы можем записать это как

$$P(\vec{\theta}|\mathbf{X}) \propto L(\vec{\theta}|\mathbf{X})\pi(\vec{\theta})$$
$$\Rightarrow$$
$$P(\vec{\theta}|\mathbf{X}) \propto L(\vec{\theta}|\mathbf{X_1})L(\vec{\theta}|\mathbf{X_2})L(\vec{\theta}|\mathbf{X_3})\pi(\vec{\theta})$$

Мы можем расписать правдоподобие в произведение, предполагая независимость каждого события. Здесь если присмотреться, то можно заметить, что

$$P(\vec{\theta}|\mathbf{X}_1) \propto L(\vec{\theta}|\mathbf{X}_1)\pi(\vec{\theta})$$
$$P(\vec{\theta}|\mathbf{X}_2, \mathbf{X}_1) \propto L(\vec{\theta}|\mathbf{X}_2)P(\vec{\theta}|\mathbf{X}_1)$$
$$P(\vec{\theta}|\mathbf{X}_3, \mathbf{X}_2, \mathbf{X}_1) \propto L(\vec{\theta}|\mathbf{X}_3)P(\vec{\theta}|\mathbf{X}_2, \mathbf{X}_1)$$

Таким образом, добавляя новые измерения, мы меняем апостериорную информацию, исходя из полученного "опыта".

Доверительный интервал по Байесу

Апостериорное распределение для параметров распределения отражает нашу степень уверенности в каком-либо конкретном значении параметра при наличии у нас какого-то опыта или эксперимента. Из этого распределения мы можем построить интервал и указать вероятность попадания параметра в этот интервал.

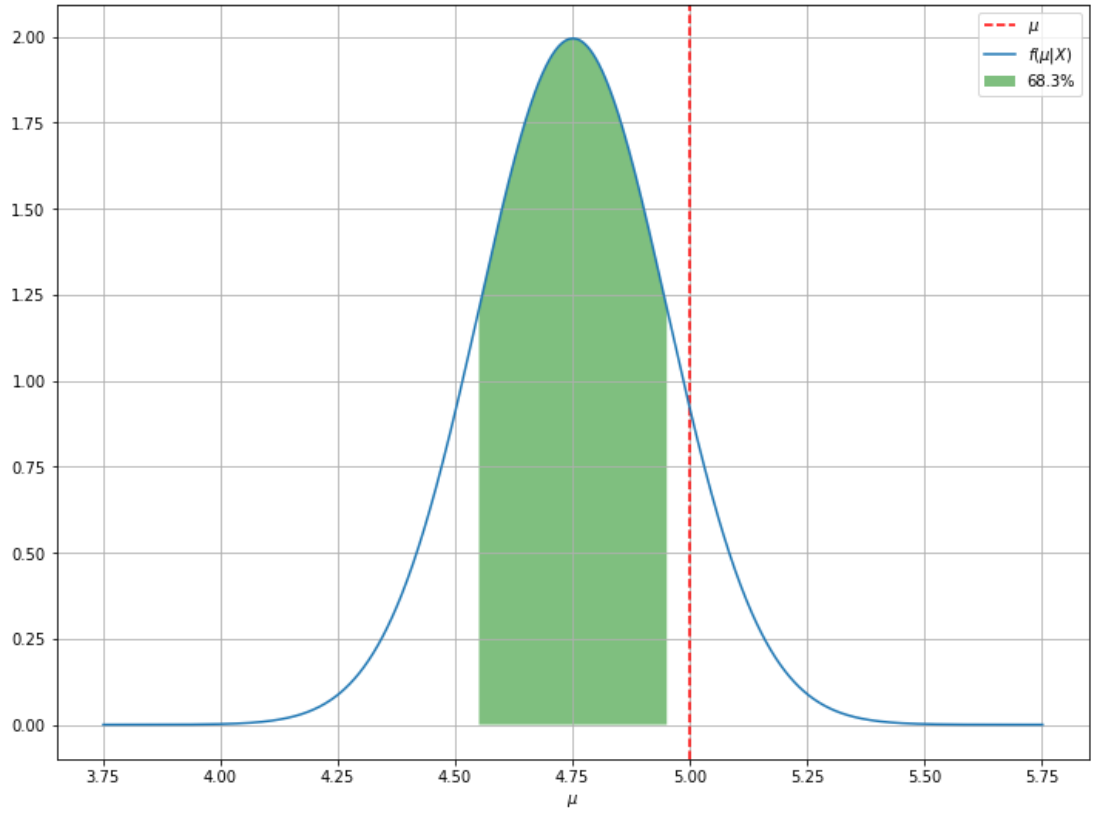
$$P(\theta_b \leq \theta \leq \theta_u) = \int_{\theta_b}^{\theta_u} P(\theta|\mathbf{X})\, d\theta = 1 - \alpha$$

где α - уровень доверия, указывает на сколько сильно нам хотелось бы ошибаться.

Такой интервал называют доверительным. В англоязычной литературе доверительный интервал по Байесу называют **credible interval**.

Пусть $\alpha = 0.317$, что соответсвует вероятности в 1σ или 68.3%. Покажем как выглядит этот

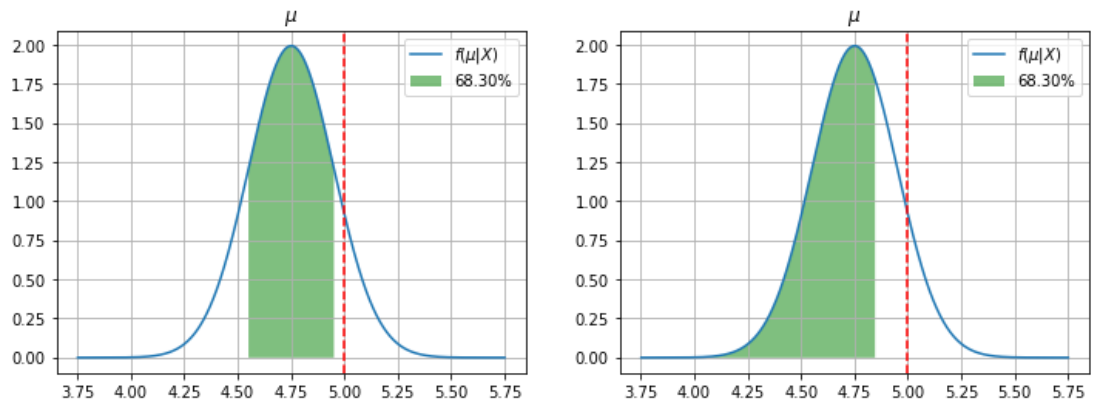
In [8]:
```
from scipy.stats import norm

# Параметры распределений
mean = 5
sigma2 = 2

# Здесь второй параметр - это стандартное отклонение, а не дисперсия
pdf = norm(mean, np.sqrt(sigma2))

# сколько точек будем брать
N = 50

# Генерируем данные
samples = pdf.rvs(size=N)

# Для простоты будем считать, что дисперсия известна и фиксирована, а все значе
# берем самое вероятное значение за основную оценку
h_mu = samples.mean()

x1s = np.linspace(h_mu-np.sqrt(sigma2/N), h_mu + np.sqrt(sigma2/N), 1000)

plt.axvline(5, c='r', ls='--', label="$\mu$")

mu = np.linspace(h_mu - 5*np.sqrt(sigma2/N), h_mu + 5*np.sqrt(sigma2/N), 1000)
plt.plot(mu, norm.pdf(mu, h_mu, np.sqrt(sigma2/N)), label="$f(\mu|X)$")
plt.fill(np.concatenate([x1s, x1s[::-1]]),
         np.concatenate([norm.pdf(x1s, h_mu, np.sqrt(sigma2/N)),
                         np.zeros(x1s.shape)]),
         alpha=.5, fc='g', label="68.3%")
plt.xlabel("$\mu$")
plt.legend()
plt.grid()
plt.show()
```



Хм, тут мы должны задумать, ведь ничего не мешает выбрать другие пределы интегрирования.

In [9]:
```
f = norm(h_mu, np.sqrt(sigma2/N))
alpha = 0.317
```

```
plt.subplot(221)
plt.title("$\mu$")

x1s = np.linspace(f.ppf(alpha/2), f.ppf(1-alpha/2), 1000)
plt.fill(np.concatenate([x1s, x1s[::-1]]),
         np.concatenate([f.pdf(x1s), np.zeros(x1s.shape)]),
         alpha=.5, fc='g', label="%.2f%%" % ( (1-alpha) * 100))

plt.plot(mu, f.pdf(mu), label="$f(\mu|X)$")
plt.axvline(5, c='r', ls='--')
plt.grid()
plt.legend()


plt.subplot(222)
plt.title("$\mu$")

x1s = np.linspace(mu.min(), f.ppf(1 - alpha), 1000)
plt.fill(np.concatenate([x1s, x1s[::-1]]),
         np.concatenate([f.pdf(x1s), np.zeros(x1s.shape)]),
         alpha=.5, fc='g', label="%.2f%%" % ( (1-alpha) * 100))

plt.plot(mu, f.pdf(mu), label="$f(\mu|X)$")
plt.axvline(5, c='r', ls='--')
plt.grid()
plt.legend()

plt.subplot(223)
plt.title("$\mu$")

x1s = np.linspace(f.ppf(alpha), mu.max(), 1000)
plt.fill(np.concatenate([x1s, x1s[::-1]]),
         np.concatenate([f.pdf(x1s), np.zeros(x1s.shape)]),
         alpha=.5, fc='g', label="%.2f%%" % ( (1-alpha) * 100))

plt.plot(mu, f.pdf(mu), label="$f(\mu|X)$")
plt.axvline(5, c='r', ls='--')
plt.grid()
plt.legend()

plt.subplot(224)
plt.title("$\mu$")

x1s = np.linspace(f.ppf(3*alpha/4), f.ppf(1-alpha/4), 1000)
plt.fill(np.concatenate([x1s, x1s[::-1]]),
         np.concatenate([f.pdf(x1s), np.zeros(x1s.shape)]),
         alpha=.5, fc='g', label="%.2f%%" % ( (1-alpha) * 100))

plt.plot(mu, f.pdf(mu), label="$f(\mu|X)$")
plt.axvline(5, c='r', ls='--')
plt.grid()
plt.legend()

plt.show()
```



Вероятность попадания в этот интервал везде одна и та же. Какой из них выбрать? Обычно выбирают такой интервал, чтобы вероятности внутри интервала были больше вероятностей вне интервала (обычно это соответствует наименьшей ширине интервала). Ниже представлены примеры наиболее часто используемые формы для доверительного интервала

$$\int_{-\infty}^{\theta_u} P(\theta|\mathbf{X})\,d\theta = 1-\alpha$$

$$\int_{\theta_l}^{\infty} P(\theta|\mathbf{X})\,d\theta = 1-\alpha$$

$$\int_{\theta_0-\delta}^{\theta_0+\delta} P(\theta|\mathbf{X})\,d\theta = 1-\alpha$$

$$\int_{-\infty}^{\theta_0-\delta_2} P(\theta|\mathbf{X})\,d\theta = \frac{\alpha}{2}, \int_{\theta_0+\delta_1}^{\infty} P(\theta|\mathbf{X})\,d\theta = \frac{\alpha}{2}$$

Если x распределен по нормальному закону (или симметричное распределение), то кратко доверительный интервал записывают как

$$x \pm \sigma_x,$$

Если интервал не симметричен

$$x^{+\delta_1}_{-\delta_2}$$

И еще раз: **доверительный интервал по Байесу показывает нашу степень в веры в то, что данный параметр принимает некое истинное значение в указанном интервале с определенной вероятностью. Формально, все значения внутри интервала подходят.**

Вид интервала зависит от выбора априорной функции распределения!

Частотный вывод

В этом случае подход внешне остается тем же, но смысл, который в него вкладывается совершенно другой. В этом подходе считается, что параметры распределения фиксированы (не являются случайной величиной), мы их просто пока еще не знаем.

Таким образом, нам нужен метод, который позволяет найти оценку неизвестного параметра. Рассмотрим один из возможных методов.

Метод максимального правдоподобия

На помощь приходит снова функция правдоподобия. Помним, что она показывает вероятность получения данных при конкретных значениях параметров.

Хорошо, что дальше? А дальше, мы ищем ее максимум, то есть максимизируем вероятность получения выборки (ищем наиболее правдоподобный вид функции)

$$\hat{\vec{\theta}} = \arg\max_{\vec{\theta}} L(\vec{\theta}|\mathbf{X})$$

Байесов вывод начинает совпадать с методом максимального правдоподобия, если априорную вероятность выбрать равномерной.

При этом, **считается, что область наблюдений X не зависит от параметров $\vec{\theta}$!**

Доверительный интервал в частотном выводе

Доверительный интервал в частотном подходе записывается также, но несет в себе совершенно

другой смысл. На английском языке этот интервал называют **confidence interval**.

Помним, что в частотном подходе значения параметров фиксированы, просто они нам не известны. А это значит, что говорить о вероятности попадания значения данного параметра в фиксированный интервал не имеет смысла. Мы либо попали (тогда вероятность 100%), либо не попали (в этом случае 0%).

То есть мы не можем говорить, что истинное значение параметра находится в этом интервале с какой-то вероятностью.

Что же тогда значит доверительный интервал в этом подходе?

На самом деле, доверительный интервал означает следующее:

> Если мы будем повторять эксперимент множество раз, и для каждого эксперимента мы будем строить свой доверительный интервал, то в p процентах случаев этот интервал накроет истинное значение параметра.

Таким образом, истинное значение параметра действительно остается константой (хоть и не известной нам), а вот границы интервала $\theta_l(\theta)$ и $\theta_u(\theta)$ становятся случайной величиной.

Гипотезы

Чтобы было проще понять, что такое доверительный интервал в частотном подходе удобно ввести понятие статистической гипотезы.

Любое утверждение касательно наблюдаемой случайной величины является гипотезой. Различают два вида гипотез

- Простая гипотеза - предположение о каком-то конкретном законе распределения (то есть все параметры считаются известными)
- Сложная гипотеза - предположение о том, что случайная величина принадлежит некому семейству распределений (есть незафиксированные параметры)

Обычно вводят гипотезу H_0 - некое простое утверждение. В противовес ей вводят гипотезу H_1 - противоположное утверждение.

Доверительный интервал (продолжение)

Попробуем в этом разобраться по шагам на примере нормального распределения, предполагая, что нам известна дисперия. Из курса мат.статистики мы можем узнать, что оценка среднего $\hat{\mu}$ распределена по нормальному закону $\mathcal{N}\left(\mu, \frac{\sigma^2}{N}\right)$.

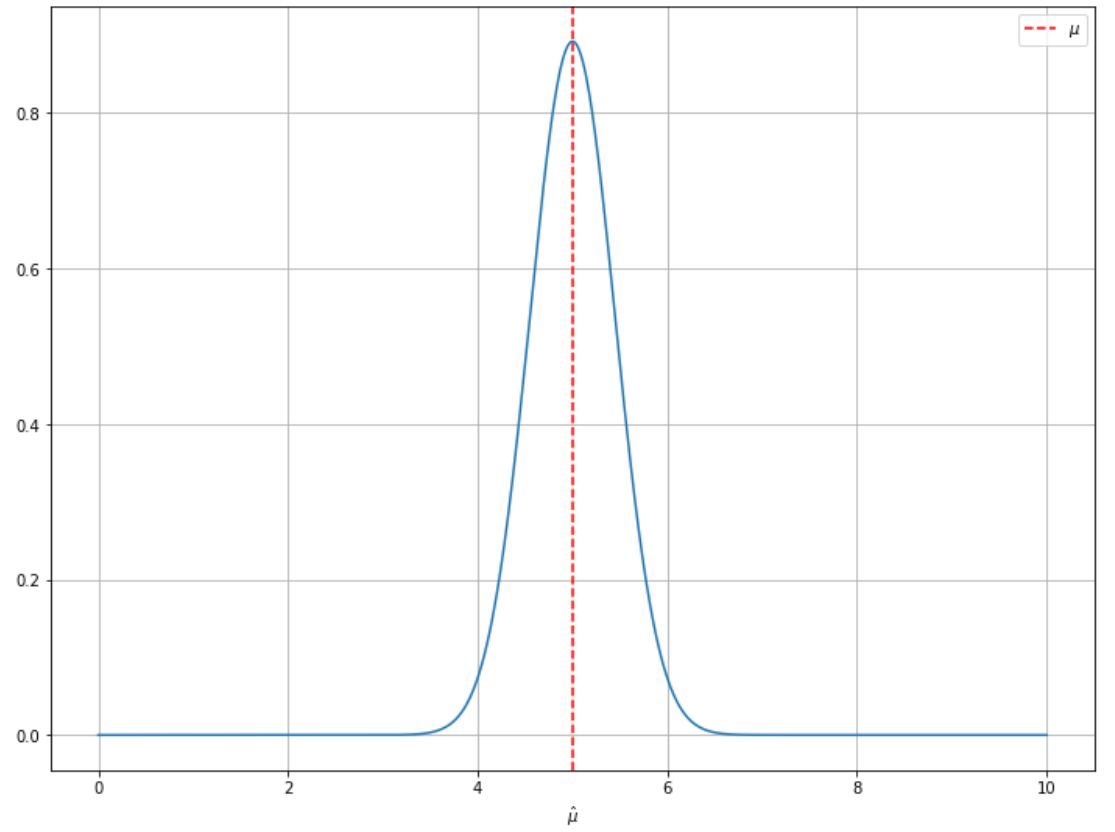
Таким образом мы можем посчитать вероятность получения любого значения $\hat{\mu}$ в зависимости от различных значений параметров μ и σ^2. Тут есть большая проблема, на практике, мы обычно никогда не знаем истинного значения параметров распределения. Все, что мы можем делать - это строить гипотезы.

Для каждой гипотезы, у нас есть распределение

$$P(\hat{\mu}|\mu) = \mathcal{N}\left(\hat{\mu}|\mu, \frac{\sigma^2}{N}\right)$$

In [10]:
```
# напомним, как выглядит распределение для оценки среднего при каком-то заданно

x = np.linspace(0, 10, 1000)
plt.plot(x, norm.pdf(x, 5, np.sqrt(2/10)))
plt.axvline(5, c='r', ls='--', label="$\mu$")
plt.xlabel("$\hat{\mu}$")
```

```
plt.grid()
plt.legend()
plt.show()
```

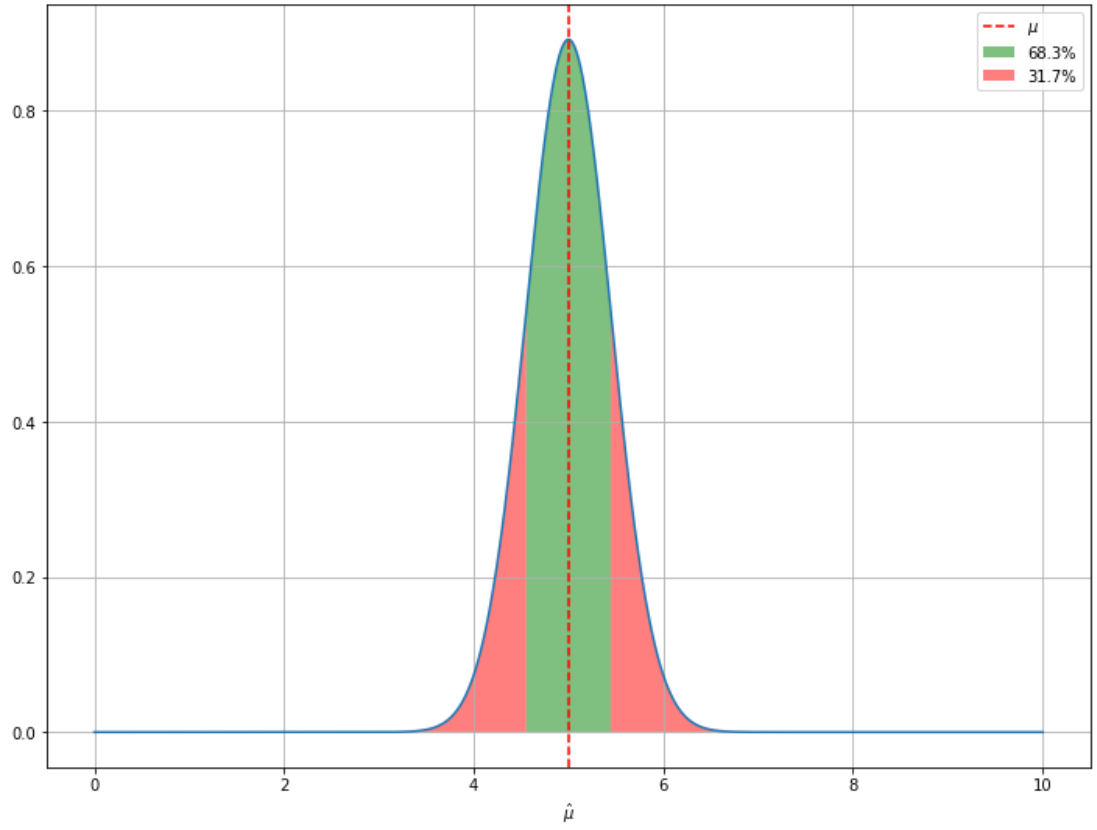


Выберем какой-то уровень α, пусть это будет 31.7% (то есть $1 - \alpha$ = 68.3%). Это соответсвует отклонению в 1σ. В данном контексте этот уровень будет определять нам область в распределение куда должен попасть $\hat{\mu}$ с заданной вероятностью для данной гипотезы:

$$P(|\mu - \hat{\mu}| \leq k_\alpha) = 1 - \alpha$$
$$P(|\mu - \hat{\mu}| > k_\alpha) = \alpha$$

Таким образом, k_α - определяет некую границу, куда с вероятностью $(1 - \alpha)$ будут попадать измеряемые значения $\hat{\mu}$ при условии, что гипотеза μ верна.

In [11]:
```
x1s = np.linspace(5-np.sqrt(2/10), 5+np.sqrt(2/10), 1000)

x1l = np.linspace(x.min(), x1s.min(), 1000)
x1r = np.linspace(x1s.max(), x.max(), 1000)

plt.fill(np.concatenate([x1s, x1s[::-1]]),
         np.concatenate([norm.pdf(x1s, 5, np.sqrt(2/10)), np.zeros(x1s.shape)])
         alpha=.5, fc='g', label="68.3%")

plt.fill(np.concatenate([x1l, x1l[::-1]]),
         np.concatenate([norm.pdf(x1l, 5, np.sqrt(2/10)), np.zeros(x1l.shape)])
         alpha=.5, fc='r', label="31.7%")

plt.fill(np.concatenate([x1r, x1r[::-1]]),
         np.concatenate([norm.pdf(x1r, 5, np.sqrt(2/10)), np.zeros(x1r.shape)])
         alpha=.5, fc='r')

plt.plot(x, norm.pdf(x, 5, np.sqrt(2/10)))
plt.axvline(5, c='r', ls='--', label="$\mu$")
plt.xlabel("$\hat{\mu}$")
plt.legend()
plt.grid()
plt.show()
```



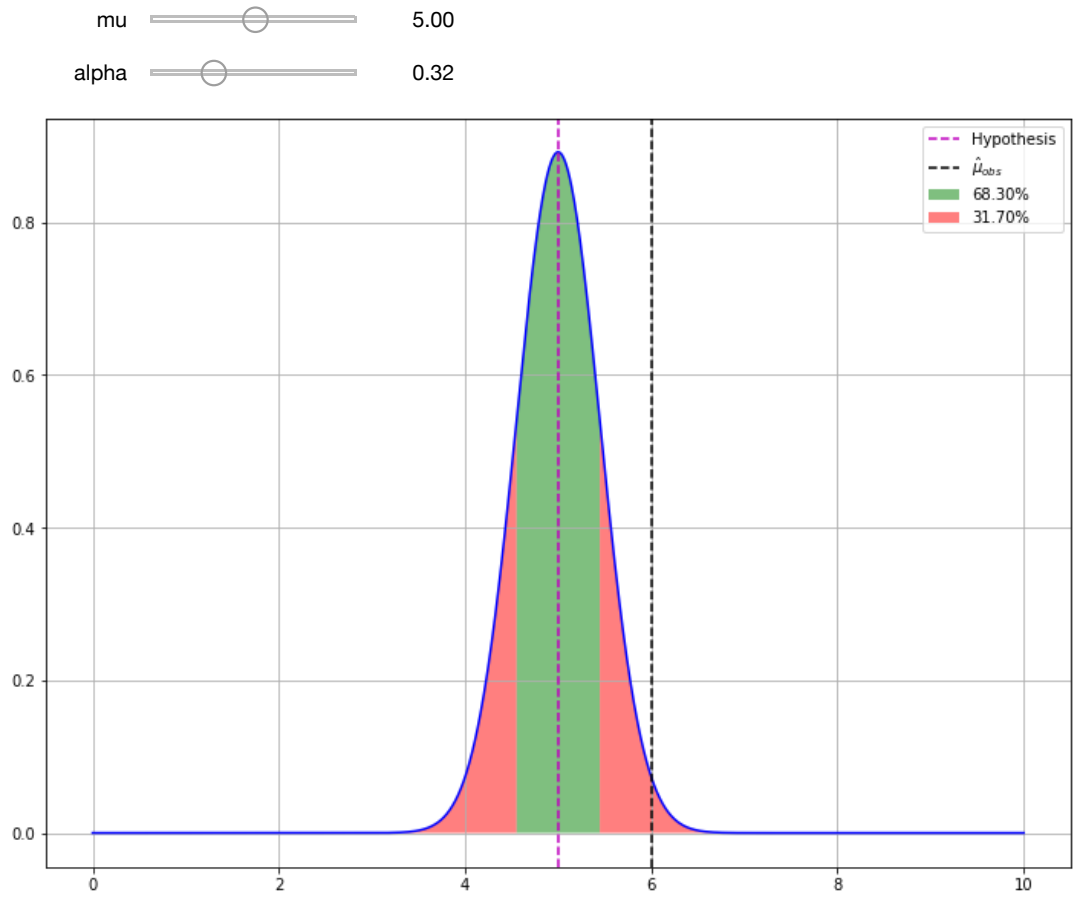
Красная область - это область, куда будет попадать $\hat{\mu}$ с вероятностью α. Если при этом мы будем отвергать нашу гипотезу для $\hat{\mu}$ в этой области и в случаях, когда гипотеза верна, то мы будем ошибаться α процентах случаях. Данную область мы будем называть критической.

Когда мы проводим эксперимент, мы получаем некую оценку параметра, в нашем примере - это оценка среднего $\hat{\mu}$. Данная величина, очевидно, является случайной величиной порожденной неким распределением, которое мы увидеть не можем. У этого распределения есть некие параметры. Теперь мы можем выдвигать различные статистические гипотезы касательно этих параметров. Для каждой гипотезы μ мы можем посмотреть, попадает ли наше вычисленного значение $\hat{\mu}$ в критическую область или нет. Это будет нам давать право отвергуть гипотезу или принять её.

При этом, множество всех принятых гипотез образует некую область. Вот эта область и будет доверительным интервалом (или областью в случае нескольких переменных) в частотном подходе.

Фактически, в данном подходе, доверительный интервал является случайной величиной, показывающей, что если проводить бесконечную серию экспериментов, то после построения доверительных интервалов для каждого эксперимента, мы получим, что $(1 - \alpha)$ процентов интервалов накроют истинное значение параметра.

In [13]:
```
from ipywidgets import interact, interactive, fixed, interact_manual
from scipy.stats import norm

x = np.linspace(0, 10, 1000)

def Figure(mu=5, alpha=0.317):
    f = norm(mu, np.sqrt(2/10))
    x1s = np.linspace(f.ppf(alpha/2), f.ppf(1-alpha/2), 100)
    x1l = np.linspace(x.min(), f.ppf(alpha/2), 100)
    x1r = np.linspace(f.ppf(1-alpha/2), x.max(), 100)

    plt.fill(np.concatenate([x1s, x1s[::-1]]),
             np.concatenate([f.pdf(x1s), np.zeros(x1s.shape)]),
             alpha=.5, fc='g', label="%.2f%%" % (100 - 100*alpha))

    plt.fill(np.concatenate([x1l, x1l[::-1]]),
             np.concatenate([f.pdf(x1l), np.zeros(x1l.shape)]),
             alpha=.5, fc='r', label="%.2f%%" % (100*alpha))

    plt.fill(np.concatenate([x1r, x1r[::-1]]),
             np.concatenate([f.pdf(x1r), np.zeros(x1r.shape)]),
             alpha=.5, fc='r')

    plt.plot(x, f.pdf(x), 'b-')
    #plt.axvline(5, c='r', ls='--', label="True Value")
    plt.axvline(mu, c='m', ls='--', label="Hypothesis")
    plt.axvline(6, c='k', ls='--', label="$\hat{\mu}_{obs}$")
    plt.legend()
    plt.grid()
    plt.show()
    return

interactive_plot = interactive(Figure, mu=(x.min(), x.max(), 0.05), alpha=(0.05
output = interactive_plot.children[-1]
output.layout.height = '600px'
interactive_plot
```



Наглядные примеры

https://seeing-theory.brown.edu/frequentist-inference/index.html (https://seeing-theory.brown.edu/frequentist-inference/index.html)

Вспомогательные параметры (Nuisance parameters)

В идеальном мире наше распределение выглядело бы как на бумаге. В реальном миру, мы должны встретиться с ограниченной возможностью нашего познания и ошибками измерительных приборов.

Для учета всех этих особенностей, мы переходим к более сложным вероятностным распределениям, что приводит к тому, что число параметров нашего распределения растет. При этом, может интересовать лишь только один параметр, остальные для нас интереса не представляют. Например, мы хотим измерить скорость света, при этом нас не сильно интересует точные оценки на все внутренние параметры прибора.

Все параметры распределения, которые нас не интересуют - называются вспомогательными (nuisance).

$$f(\mathbf{X}|\vec{\theta},\vec{\eta})$$

где $\vec{\theta}$ - параметры, которые нам бы хотелось найти, $\vec{\eta}$ - дополнительные параметры.

Есть разные подходы учета данных параметров, наиболее простой он в Байесовом подходе

$$f(\vec{\theta},\vec{\eta}|\mathbf{X}) \propto \pi(\vec{\theta},\vec{\eta}) f(\mathbf{X}|\vec{\theta},\vec{\eta})$$

Откуда просто маргинализацией мы получим распределение для интересующих нас параметров

$$f(\vec{\theta}|\mathbf{X}) \propto \int f(\vec{\theta},\vec{\eta}|\mathbf{X})\, d\vec{\eta}$$

В частотном подходе значительно тяжелее (хорошим примером является распределение Стьюдента). В самом лучшем случае, мы можем построить такую статистику, которая не будет зависеть от вспомогательных параметров и с помощью нее определить область покрытия значений для экспериментальных наблюдений. В случае чуть похуже, у нас может быть сделано дополнительное измерение, накладывающее ограничение на вспомогательные параметры (тут привет от Байесова подхода). В худшем случае придется использовать **profile likehood**.

Для уменьшения влияния вспомогательных параметров на ширину доверительного интервала можно использовать дополнительные измерения $\mathbf{Y}$, которые зависят только от вспомогательных параметров, что позволяет построить функцию правдоподобия в виде

$$L(\vec{\theta},\vec{\eta}|\mathbf{X},\mathbf{Y}) = L(\vec{\theta},\vec{\eta}|\mathbf{X}) L(\vec{\eta}|\mathbf{Y})$$

В противном случае, могут использовать **profile likehood**. В этом подходе фиксируются интересуемые параметры, а остальные параметры оцениваются исходя из этого методом максимального правдоподобия

$$L(\vec{\theta}) = \max_{\vec{\omega}} L(\vec{\theta},\vec{\omega}) = L(\vec{\theta},\hat{\vec{\omega}})$$

Здесь $\hat{\vec{\omega}}$ - это оценка по методу максимального правдоподобия при заданых каких-то значениях $\vec{\theta}$.

Статистические критерии

Мы уже знаем, что такое статистические гипотезы. Обсудим более подробно, как именно отдавать предпочтение той или иной гипотезе.

И так, у нас есть две гипотезы H_0 и H_1. Для проверки гипотезы вводят некий критерий λ, согласно которому H_0 принимается или отвергается в пользу H_1. Критерий определяет критическую область значений $\mathbf{X}$, при которых мы отвергаем гипотезу H_0 в пользу H_1. Обозначим эту область W, Пространство всех возможных значений $\mathbf{X}$ будем обозначать Ω. Следовательно область $\Omega - W$ - это допустимая область, где гипотеза H_0 не может быть отвергнута.

Размер критической области обычно подбирается таким образом, чтобы вероятность попадания критерия в эту область равнялась желаемому уровню значимости

$$P(\lambda \subset W|H_0) = \alpha$$

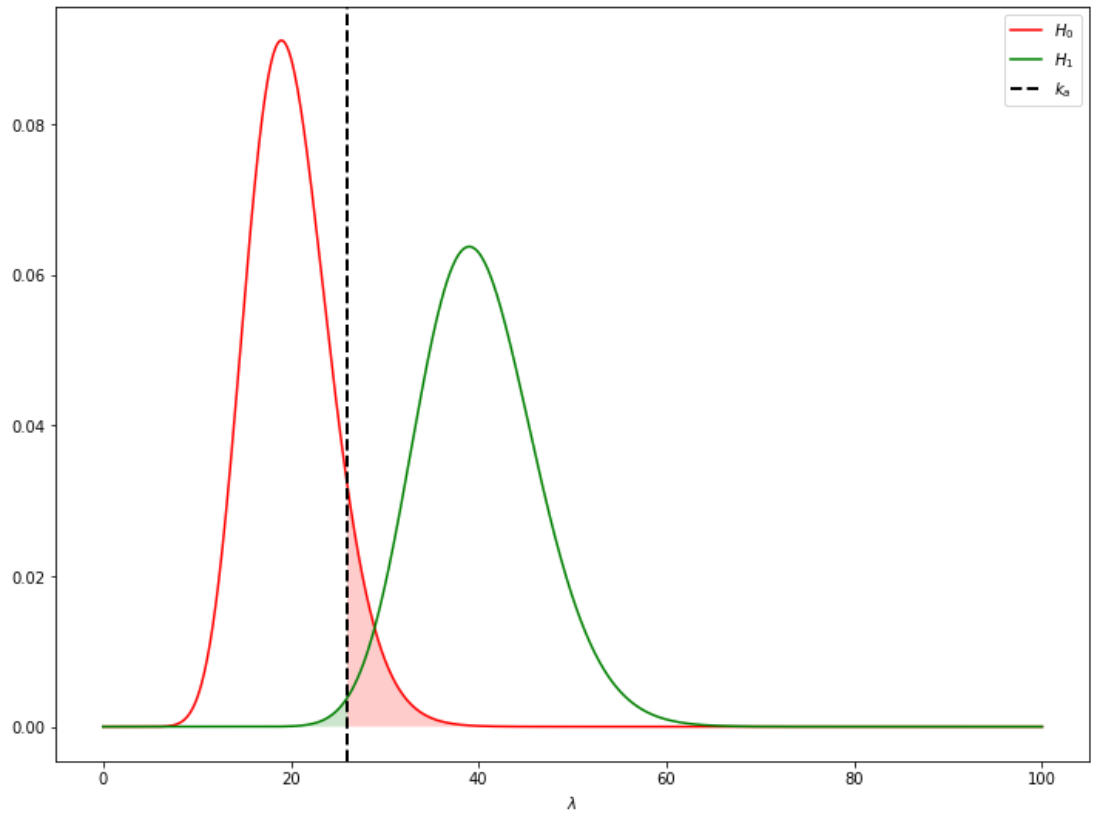
Фактически, α определяет вероятность того, что мы отвергаем гипотезу, когда она верна. Это называется ошибкой первого рода.

При этом вероятность попадания критерия в допустимую область в случае предположения гипотезы H_1, называются мощностью критерия

$$P(\Lambda \subset W|H_1) = 1 - \beta$$

Данная величина показывает вероятность ошибки, в случае принятия гипотезы H_0, в то время как верна гипотеза H_1. Это называется ошибкой второго рода. Данная величина показывает на сколько точно мы можем отделить одну гипотезу от другой.

In [48]:
```
from scipy.stats import gamma
import numpy as np

# Просто отрисовка распределения в зависимости от гипотезы
def Hypo(m, m0, label, xmin=0, xmax=100, fc='g'):
    x = np.linspace(xmin, xmax, 1000)
    f = gamma(m)
    plt.plot(x, f.pdf(x), c=fc, label=label)

    if m0 > 0:
        xs = np.linspace(m0, xmax, 1000)
    else:
        xs = np.linspace(xmin, -m0, 1000)

    #plt.axvline(m, c=fc, ls='--', label=label)
    plt.fill(np.concatenate([xs, xs[::-1]]),
         np.concatenate([f.pdf(xs), np.zeros(xs.shape)]),
         alpha=.2, fc=fc)

m0 = 26
Hypo(20, m0, "$H_0$", fc='r')
Hypo(40, -m0, "$H_1$", fc='g')
plt.axvline(m0, c='k', ls='--', lw=2, label="$k_{a}$");
plt.xlabel("$\lambda$")
plt.legend()
plt.show()
```



Здесь нарисовано распределение некого исскуственного критерия при условии, что верная гипотеза H_0 (красная линия), и при условии верности гипотезы H_1 (зеленая линия).

Здесь вертикальная черная линия определяет критическую область для гипотезы H_0. Красная область показывает область равную критерию значимости (ошибка первого рода). Если статистический критерий попадает в данную область, то мы отвергаем гипотезу H_0. Также данная область показывает вероятность допустить ошибку при отвергании гипотезы H_0.

Зеленая область нам дает β (ошибка второго рода) - вероятность того, что мы ошиблись приняв гипотезу H_0. Фактически, β отвечает за степень отличимости одной гипотезы от другой.

В настоящее время наряду с пороговым значением критерия вводят так называемое P-value, которое является просто вероятностью встретить значение критерия больше наблюдаемого (для случая одностороннего критерия)

$$P(k > k_{obs}) = \int_{k_{obs}}^{\infty} f(k)\, dk$$

Таким образом, если P-value меньше, чем α, то мы вынуждены отвергнуть нулевую гипотезу. Нередко P-value указывают в Z масштабе, т.е. в квантилях нормального распределения.

Критерий Неймана-Пирсона

В случае простых гипотез H_0 и H_1, как в примере выше, существует всюду оптимальный односторонний критерий

$$\lambda(\mathbf{X}) = \frac{L(H_1|\mathbf{X})}{L(H_0|\mathbf{X})} > k_\alpha$$

где L - это функция правдоподобия для соответсвующей гипотезы, k_α - специальная константа, которая определяется исходя из уровня значимости.

Если $\lambda(\mathbf{X}) > k_\alpha$, то принимают гипотезу H_1, если $\lambda(\mathbf{X}) \leq k_\alpha$ - H_0.

В самом общем случае не всегда возможно построить функцию правдоподобия, в этом случае могут помочь методы машинного обучения, которые позволяют аппроксимировать это отношение.

Также не всегда возможно найти распределение $\lambda(\mathbf{X})$ в явном виде. В этом случае, данное распределение можно получить с помощью Монте-Карло генераторов. То есть мы банально генерируем как можно больше вариантов исхода, для которых смотрим значение критерия.

Теорема Уилка (Wilk's theorem)

Пусть у нас есть некая функция распределения $f(\mathbf{X}|\vec{\theta})$ (функция правдоподобия), котороая зависит от $\vec{\theta}$ параметров.

Пусть у нас есть гипотеза H_0, такая что $\vec{\theta} \in \Theta_0$, где Θ_0 - это некая область в пространстве возможных значений набора параметров (в оригинальной статье, Вилк просто фиксировал некоторые параметры, задавая для них значения).

Пусть у нас есть альтернативная гипотеза, такая что $\vec{\theta} \in \Omega$, где Ω - это пространство всех возможных значений параметра (ни один параметр не зафиксирован).

Это нам дает критерий

$$\lambda = -2 \ln \frac{\max\limits_{\vec{\theta} \in \Theta_0} f(\mathbf{X}, \vec{\theta})}{\max\limits_{\vec{\theta} \in \Omega} f(\mathbf{X}, \vec{\theta})}$$

Здесь видно, что гипотеза H_0 включена в общее пространство значений Ω.

> Если ν - размерность пространства Ω, а r - размерность Θ_0, то величина λ в предположении правдивости гипотезы H_0 распределена асимптотически как $\chi^2_{\nu-r}$.

Это очень удобно использовать с профилем функции правдоподобия

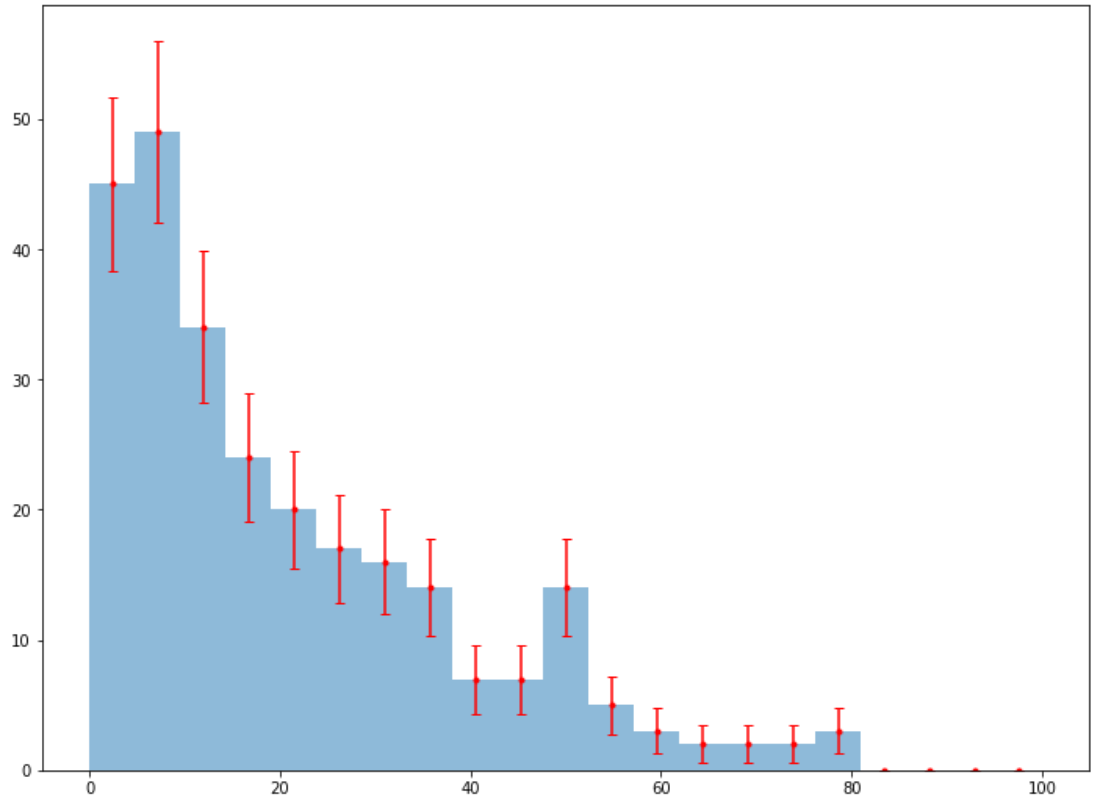
$$\lambda(\vec{\theta}) = -2 \ln \frac{L(\vec{\theta}, \hat{\vec{\omega}}|\mathbf{X})}{L(\hat{\vec{\theta}}, \hat{\hat{\vec{\omega}}}|\mathbf{X})}$$

Пример

Пусть мы провели некий эксперимент и получили следующий результат:

In [80]:
```
from scipy.stats import poisson

def gen(v, f_v):
    """
        Генерируем наши данные
    """
    N = poisson.rvs(v)
    return f_v.rvs(N)
```

In [168]:
```
signal = gen(s, f_s)
background = gen(b, f_b)
#np.savetxt("data.txt", np.hstack([background, signal]))

samples = np.loadtxt(open("data.txt"))
bins = np.linspace(0, 100, 22)
vv, bb, _ = plt.hist(samples, bins=bins, alpha=0.5)
plt.errorbar((bb[:-1] + bb[1:])/2, vv, yerr=np.sqrt(vv), fmt=".", c='r', capsiz
```



У нас теперь две простые гипотезы (параметры распределений будем считать известными для простоты):

- H_0 - Присутствует только фон, распределенный по экспоненциальному закону
$$f(x) = f_b(x) = \lambda e^{-\lambda x}$$
- H_1 - Присутствует фон и сигнал, распределенные по экспоненциальному и нормальному

закону, соответственно.

$$f(x) = c_b f_b(x) + c_s f_s(x) = c_b \lambda e^{-\lambda x} + c_s \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

Для того, чтобы выбрать одну из этих гипотез, нам нужно построить критерий, который позволит нам это сделать

In [79]:
```
from scipy.stats import poisson, expon, norm

# Ожидаемый уровень фона и сигнала
b = 270
s = 20

c_s = s / (b + s)
c_b = b / (b + s)

# истинные значения параметров для сигнала
mu = 50
sigma2 = 10
f_s = norm(mu, np.sqrt(sigma2))

# истинные значения параметров для фона
lamb = 0.05
f_b = expon(scale=1/lamb)
```

Unbinned Likehood

И так, в ходе эксперимента мы получили N независимых наблюдений: $x_1, \dots, x_N$. Таким образом, мы можем записать функцию правдоподобия

$$L(\vec{\theta}|\mathbf{X}) = \prod_{n=1}^{N} f(x_n|\vec{\theta})$$

Стоить отметить, что на практике, в наших экспериментах число N является также случайной величиной, а следовательно - нам нужно модифицировать функцию правдоподобия, чтобы учесть этот факт

$$L(\vec{\theta}|\mathbf{X}) = f(N|\vec{\theta}) \prod_{n=1}^{N} f(x_n|\vec{\theta})$$

Чаще всего $f(N|\vec{\theta})$ - это распределение Пуассона, что в итоге нам дает

$$L(\vec{\theta}|\mathbf{X}) = \frac{\nu^N}{N!} e^{-\nu} \prod_{n=1}^{N} f(x_n|\vec{\theta})$$

Обычно, мы будем выделять фоновые события (background) и сигнальные события (signal). Для каждого из них будет свое распределение f_b и f_s. Полное распределение f является смесью этих распределений

$$L(\vec{\theta}|\mathbf{X}) = \frac{\nu^N}{N!} e^{-\nu} \prod_{n=1}^{N} \left(c_s f_s(x_n|\vec{\theta}) + c_b f_b(x_n|\vec{\theta}) \right)$$

где $c_s + c_b = 1$, s и b - среднее ожидаемое количество сигнальных и фоновых событий. Значения c_s и c_s довольно легко найти - это просто доля соответствующих событий к полному числу предполагаемых событий (вероятность того, что это фон или сигнал по сути)

$$c_s = \frac{s}{s+b}$$
$$c_b = \frac{b}{s+b}$$
$$\nu = s + b$$

В итоге

$$L(\vec{\theta}|\mathbf{X}) = \frac{e^{-(s+b)}}{N!}\prod_{n=1}^{N}\left(sf_s(x_n|\vec{\theta}) + bf_b(x_n|\vec{\theta})\right)$$

или

$$\mathcal{L} = \ln L = -(s+b) - N! + \sum_n \ln\left(sf_s(x_n|\vec{\theta}) + bf_b(x_n|\vec{\theta})\right)$$

В самом общем случае, s и b - это тоже функции вектора параметров $\vec{\theta}$.

Часто вводят $s = \mu s_0$, где μ - это сила сигнала (0 - нет сигнала, 1 - есть), а s_0 - предсказываемое теорией значение

Критерий

У нас две простые гипотезы, лемма Пирсона-Неймана нам подсказывает, что самый лучший критерий для их разделения - это отношение правдоподобий этих гипотез. Также можно заметить, что одна из этих гипотез является вложенной в другую (H_1 превращается в H_0, если сила сигнала становится 0).

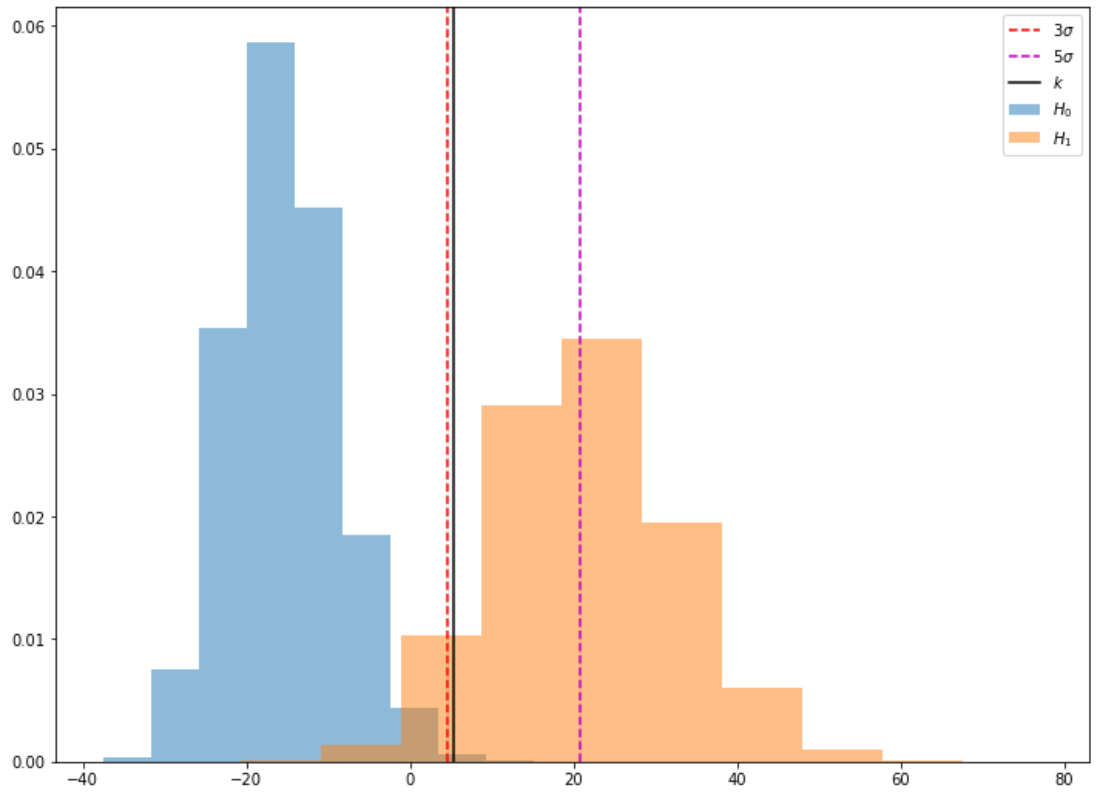
$$\frac{L_b}{L_{s+b}} = e^s \prod_{n=1}^{N} \frac{1}{1 + \frac{sf_s(x_n|\vec{\theta})}{bf_b(x_n|\vec{\theta})}}$$

$$-2\ln\frac{L_b}{L_{s+b}} = -2s + 2\sum_{n=1}^{N}\ln\left(1 + \frac{sf_s(x_n|\vec{\theta})}{bf_b(x_n|\vec{\theta})}\right)$$

Прежде, чем работать с данным критерием, рассмотрим его поведение в зависимости от гипотезы

In [104]:
```
# Проведем наш виртуальный эксперимент множество раз

h0 = []
h1 = []

def K(samples, b, f_b, s, f_s):
    return -2*s - 2*np.log(b*f_b.pdf(samples)).sum() + 2*np.log(b*f_b.pdf(sampl

for step in range(100000):
    background = gen(b, f_b)
    signal = gen(s, f_s)

    k_H0 = K(background, b, f_b, s, f_s)
    k_H1 = K(np.hstack([background, signal]), b, f_b, s, f_s)

    h0.append(k_H0)
    h1.append(k_H1)
```

In [105]:
```
samples = np.loadtxt(open("data.txt"))
k = K(samples, b, f_b, s, f_s)

plt.hist(h0, label="$H_0$", density=True, alpha=0.5)
plt.hist(h1, label="$H_1$", density=True, alpha=0.5)

plt.axvline(np.quantile(h0, 0.997), label="$3 \sigma$", color="r", ls="--")
plt.axvline(np.quantile(h0, 1 - 5.7 * 10**(-7)), label="$5 \sigma$", color="m",
plt.axvline(k, label="$k$", color="k")
plt.legend()
plt.show()
```



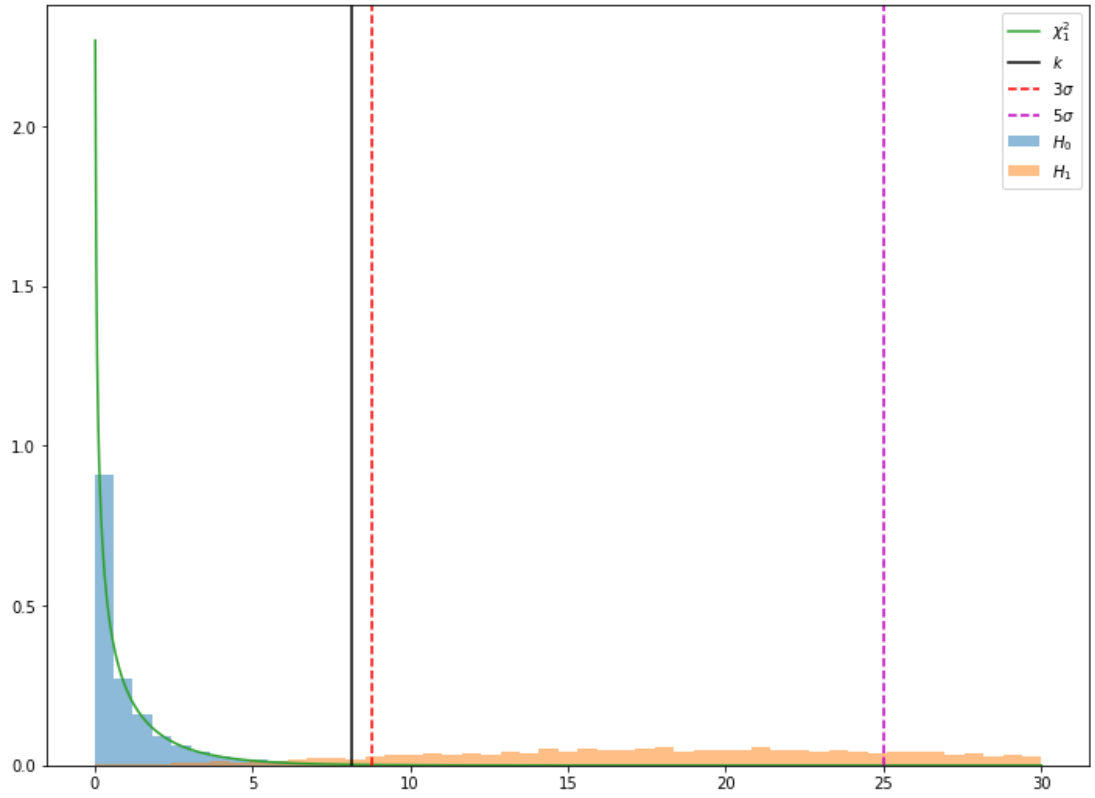
Мы можем пойти дальше и оценивать мощность сигнала исходя из данных (помним, что $s = \mu s_0$, где μ - это мощность сигнала, а не среднее значение нормального распределения), в этом случае

$$k = -2 \ln \frac{L(\mu = 0)}{L(\hat{\mu})}$$

In [106]:
```
import scipy

# Проведем наш виртуальный эксперимент множество раз

hm0 = []
hm1 = []

np.random.seed(5)
def Km(samples, b, f_b, s, f_s):

    lamb = 1/f_b.mean()
    mu, sigma2 = f_s.mean(), f_s.var()

    def f(m):
        return m * s - np.log(m*s*f_s.pdf(samples) + b*f_b.pdf(samples)).sum()

    res = scipy.optimize.minimize(f, 1, method='Nelder-Mead', tol=1e-6)
    m = res.x[0]

    top = np.log(b*f_b.pdf(samples)).sum()
```

```
    bottom = np.log(m * s * f_s.pdf(samples) + b * f_b.pdf(samples)).sum()
    return -2*m*s - 2*top + 2*bottom

for step in range(10000):
    background = gen(b, f_b)
    signal = gen(s, f_s)

    k_H0 = Km(background, b, f_b, s, f_s)
    k_H1 = Km(np.hstack([background, signal]), b, f_b, s, f_s)

    hm0.append(k_H0)
    hm1.append(k_H1)
```

In [107]:
```
plt.hist(hm0, bins=np.linspace(0, 30, 50), density=True, label="$H_0$", alpha=0
plt.hist(hm1, bins=np.linspace(0, 30, 50), density=True, label="$H_1$", alpha=0

samples = np.loadtxt(open("data.txt"))
k = Km(samples, b, f_b, s, f_s)

from scipy.stats import chi2
x = np.linspace(0, 30, 1000)
plt.plot(x, chi2(1).pdf(x), label="$\chi_{1}^2$")
plt.axvline(k, color="k", label="$k$")

plt.axvline(chi2(1).ppf(0.997), label="$3 \sigma$", color="r", ls="--")
plt.axvline(chi2(1).ppf(1 - 5.7 * 10**(-7)), label="$5 \sigma$", color="m", ls=
plt.legend()
plt.show()
```



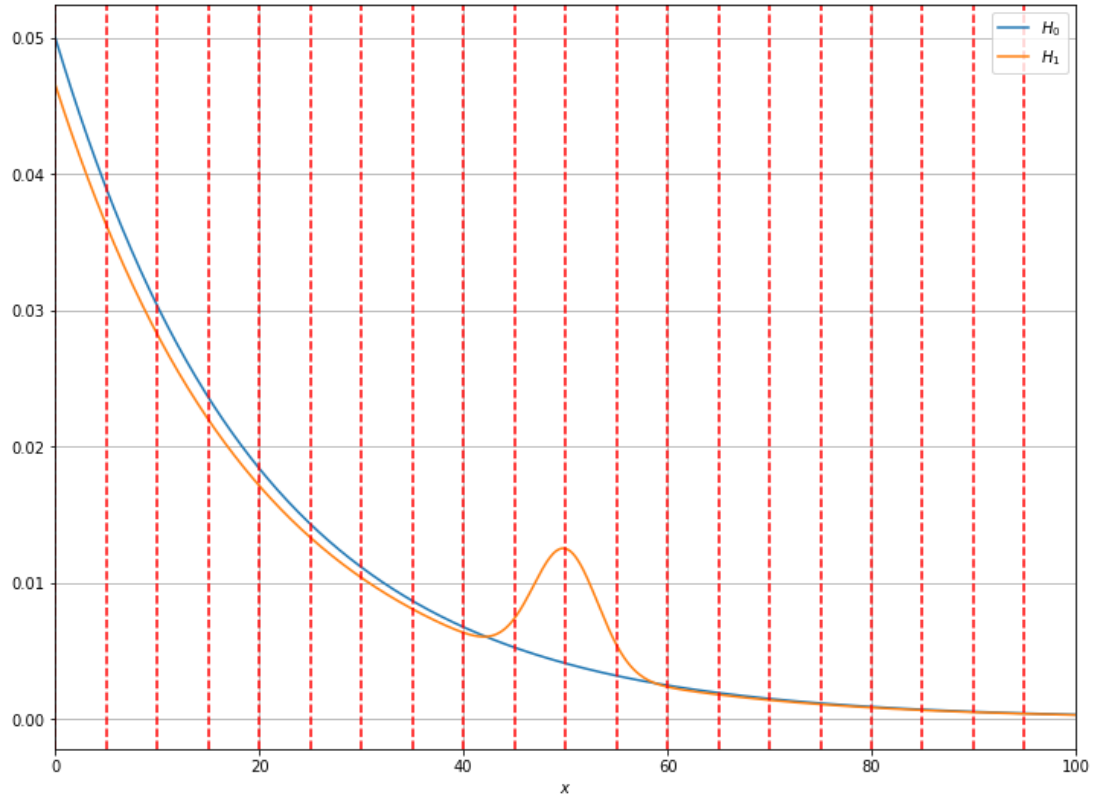
Как мы видим, картина согласуется с теоремой Вилка - мы получили χ^2-распределение с одной степенью свободы.

Binned Likehood

Нередко, данные будут доступны исключительно только в виде гистограммы. Как работать с такими данными мы рассмотрим ниже.

И так, мы знаем, что для каждой из гипотез у нас есть известный закон распределения. Для проверки принадлежности данных к выбранному закону распределния будем рассматривать дискретную величину x, которая может принимать K значений: x_1, ..., x_K. Так как законы распределний у нас непрерывные, нам нужно их привести к дискретному виду. Это легко сделать

In [259]:
```
from scipy.stats import expon, norm

x = np.linspace(0, 100, 1000)
f_b = expon(scale=1/lamb)
f_s = norm(mu, np.sqrt(sigma2))

plt.xlabel("$x$")
plt.xlim(0, 100)
plt.plot(x, f_b.pdf(x), label="$H_0$");
plt.plot(x, c_b * f_b.pdf(x) + c_s * f_s.pdf(x), label="$H_1$");
plt.legend()
for xv in np.arange(0, x.max(), 5):
    plt.axvline(xv, ls='--', c='r')
plt.grid()
```



Введем вектор $\vec{\theta}$, компоненты которого $\theta_k = 1$ если $x = x_k$ и $\theta_k = 0$ во всех остальных случаях.

Например, если x результат броска игральной кости, то если выпало 2

$$\vec{\theta}(x = 2) = (0, 1, 0, 0, 0, 0)^T$$

Таким образом $\vec{\theta}$ является многомерной случайной величиной (обобщение испытания Бернули), для которой имеет место

$$\sum_k \theta_k = 1$$

Все это мы записали для одного исхода. В общем случае множества исходов, мы можем ввести матрицу $\mathbf{\Theta}$, записывая $\vec{\theta}$ для исходов в столбцы этой матрицы.

$$\mathbf{\Theta} = (\vec{\theta}_1, \dots, \vec{\theta}_N)$$

Полную частоту появления какого-то значения случайной величины мы можем получить просуммировав по всем исходам

$$r_k = \sum_n \theta_{kn}$$

Мы только что изобрели гистограмму. Если внимательно смотреть на $\vec{r}$, то можно понять что мы фактически получили величину распределенную по полиномиальному закону

$$P(\vec{r}) = \frac{N!}{r_1!r_2!\dots r_K!} p_1^{r_1} p_2^{r_2} \dots p_K^{r_K}$$
$$\sum_k r_k = N$$
$$P(x = x_k) = p_k$$

Если исходов довольно много, то $\vec{r}$ является суммой довольно большого числа случайных величин. А как мы уже знаем, биномиальное распределение стремится к распределению Пуассона, то это значит, что распределение количества попаданий в каждом бине будет стремится к распределению Пуассона.

Это дает нам возможность записать вероятность получения гистограммы

$$L(\vec{\theta}|\mathbf{X}) = \prod_k \frac{\nu_k^{r_k}}{r_k!} e^{-\nu_k} = \prod_k \frac{(s_k + b_k)^{r_k}}{r_k!} e^{-s_k - b_k}$$

Далее в силу центральной предельной теоремы, если число событий в бине достаточно велико, получаем в каждом бине нормальное распределение

$$r_k \sim \mathcal{N}(r_k|\mu = Np_k, \sigma^2 = Np_k)$$

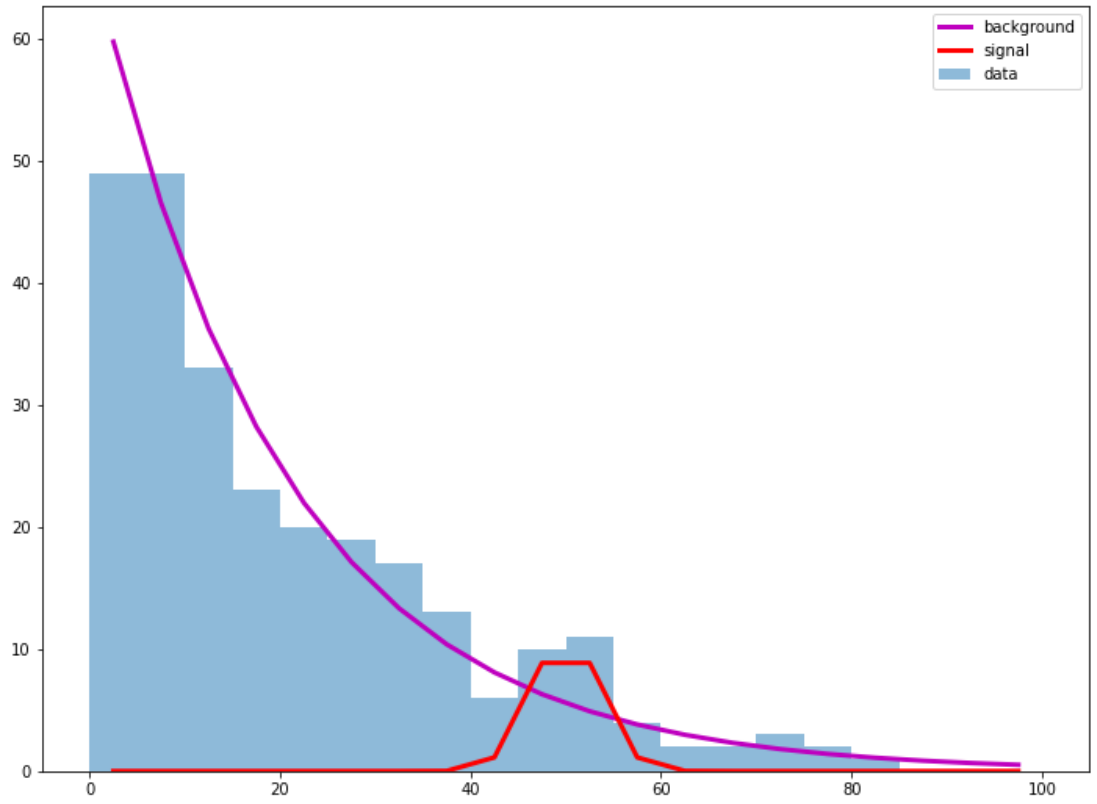
Пример

Построим такой же критерий, как и в Unbinned Likehood

$$-2\ln\frac{L_b}{L_{b+s}} = -2\ln\frac{\prod_k \frac{b_k^{r_k}}{r_k!} e^{-b_k}}{\prod_k \frac{(\mu s_k + b_k)^{r_k}}{r_k!} e^{-\mu s_k - b_k}} = -2\ln\prod_k \frac{e^{\mu s_k}}{\left(\mu\frac{s_k}{b_k} + 1\right)^{r_k}} = -2\sum_k \left[\mu s_k - r_k \ln\left(1 + \mu\right.\right.$$

здесь μ оцениваем из данных, что позволяет также воспользоваться теоремой Вилка

In [159]:
```
from scipy.stats import norm, expon, poisson

K = 20
bins = np.linspace(0, 100, K+1)
b_k = np.zeros(K)
s_k = np.zeros(K)
for i in range(K):
    b_k[i] = b * (f_b.cdf(bins[i+1]) - f_b.cdf(bins[i]))
    s_k[i] = s * (f_s.cdf(bins[i+1]) - f_s.cdf(bins[i]))

samples = np.loadtxt(open("data.txt"))

r_k, _, _ = plt.hist(samples, bins=bins, label="data", alpha=0.5);
plt.plot((bins[:-1] + bins[1:])*0.5, b_k, "m-", lw=3, label="background")
plt.plot((bins[:-1] + bins[1:])*0.5, s_k, "r-", lw=3, label="signal")
plt.legend()
plt.show()
```



$$\sum_k r_k \ln(\mu s_k + b_k) - \mu s_k$$

In [160]:
```
import scipy

# Проведем наш виртуальный эксперимент множество раз

hb0 = []
hb1 = []

np.random.seed(5)
def Kb(samples, bins, s_k, b_k):
    r_k, _ = np.histogram(samples, bins=bins)
    def f(m):
        return (m * s_k - r_k * np.log(m*s_k + b_k)).sum()

    res = scipy.optimize.minimize(f, 1, method='Nelder-Mead', tol=1e-6)
    m = res.x[0]

    return -2 * (m * s_k - r_k * np.log(m * s_k / b_k + 1)).sum()
```

```
for step in range(1000):
    background = gen(b, f_b)
    signal = gen(s, f_s)

    k_H0 = Kb(background, bins, s_k, b_k)
    k_H1 = Kb(np.hstack([background, signal]), bins, s_k, b_k)

    hb0.append(k_H0)
    hb1.append(k_H1)
```

In [164]:
```
samples = np.loadtxt(open("data.txt"))

k = Kb(samples, bins, s_k, b_k)

plt.hist(hb0, bins=50, label="$H_0$", density=True, alpha=0.5)
plt.hist(hb1, bins=50, label="$H_1$", density=True, alpha=0.5)

from scipy.stats import chi2
x = np.linspace(0, 30, 1000)
plt.plot(x, chi2(1).pdf(x), label="$\chi_{1}^2$")
plt.axvline(k, color="k", label="$k$")

plt.axvline(chi2(1).ppf(0.997), label="$3 \sigma$", color="r", ls="--")
plt.axvline(chi2(1).ppf(1 - 5.7 * 10**(-7)), label="$5 \sigma$", color="m", ls=
plt.legend()
plt.xlim(0, 40)
plt.show()
```