Université de Genève Autumn term 2022

Statistical Methods in Physics (14P058)

Prof. Federico Sánchez (federico.sancheznieto@unige.ch)
Dr. Hepeng Yao (hepeng.yao@unige.ch)
Dr. Knut Zoch (knut.zoch@unige.ch)

Exercise III – Monte Carlo integration I

27 October & 3 November 2022, 9:15, room: SCI–202

Today we look at Monte Carlo integration techniques and a small example from particle physics where these techniques could be used. Let's look at the top quark, the heaviest known elementary particle. The top quark is unstable and decays. Thus, according to the Heisenberg uncertainty principle, its mass distribution is not a delta peak, but follows a Cauchy distribution (in HEP often referred to as Breit-Wigner distribution).

The top quark mass is measured to be approximately $m_t = 173\,\text{GeV}$ with a decay width calculated to be $\Gamma_t \approx 1.33\,\text{GeV}$. The decay width corresponds to the "full width at half maximum" value of the Cauchy distribution (i.e. $\Gamma = 2\gamma$).

Task 1: *Accept–reject method*

a) Implement two functions in python that evaluate the Cauchy and Gauss distributions according to the equations below.

$$\text{Cauchy distribution:} \quad f_C(x; x_0, \gamma) = \frac{1}{\pi\gamma} \cdot \frac{\gamma^2}{(x - x_0)^2 + \gamma^2} \tag{1}$$

$$\text{Gauss distribution:} \quad f_G(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \cdot \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{2}$$

Hint: Choose the definitions of the python functions to have the following signatures:

```
def Cauchy(x, x_0, gamma):
    """Implementation here."""
def Gauss(x, mu, sigma):
    """Implementation here."""
```

b) Using your definition, plot the mass distribution of the top quark in the interval $[170, 176]$. Remember that Γ_t is the *full* width at half maximum, but γ is the half width.

c) Now use the accept–reject method to sample from a Cauchy distribution. Initialise a random number generator (`np.random.default_rng()`) and generate a set of 2000 pairs of random x, y values. Choose $x \in [170, 176]$. For y, use a uniform distribution $h(x)$ such that $\forall x \in [170, 176] : h(x) \geq f_C(x)$. Remember: the closer $h(x)$ is to the Cauchy distribution, the higher your acceptance rate! Plot the generated pairs of x, y values using the `plt.scatter(x, y)` function, together with the Cauchy distribution.

d) Implement the acceptance criterion: accept all x, y pairs where $y <= f_C(x)$. Then make the same plot as before, but with accepted and rejected points in different colours.

Hint: comparison operators in python can also work with lists. For example, in the expression `a = (b <= c)`, `b` and `c` can be lists. `a` will then be a list of booleans.

e) Calculate the acceptance rate and comment.

f) Repeat the accept–reject method with a Gaussian distribution as an envelope. The Gauss distribution $f_G(x)$ should serve as an *envelope* to the Cauchy distribution, so again $\forall x \in [170, 176] : f_G(x) \geq f_C(x)$. Choose $\mu = m_{top}$, then pick an appropriate value for σ and a scaling value for the entire function. Remember: the closer you are to the Cauchy distribution, the higher the acceptance rate. Plot them together.

g) Now, for the already generated $x \in [170, 176]$, sample 2000 new y values based on the Gaussian envelope that you picked. Create a plot that shows the Cauchy distribution of the top quark mass, the Gaussian envelope, as well as the accepted and rejected sample points (in different colours).

h) Calculate the acceptance rate again and compare it to that of the uniform envelope. Is it what you expected? Comment.

Task 2: *Bonus: equal-density sampling*

When trying to fill the sampled values into a histogram with `plt.hist()`, you will notice that their density is not flat in x (because we did not use a uniform envelope).

a) What event weights do you need to add such that the filled histogram follows the shape of the top quark mass distribution? Plot the final histogram and overlay it with the Cauchy distribution.

   Hint: remember which function you used to sample the y values. You need to use that function and scale with the number of bins / number of samples.

b) Once you have added the event weights correctly, get the return values of `plt.hist()` like in the following code:

```
contents, bin_edges, _ = plt.hist(x, weights=weights, bins=bins)
```

   Calculate the sum of all bin contents times the bin width. This should be an estimator of the integral of the Cauchy distribution for $x \in [170, 176]$!

c) Use the following snippet to calculate the actual integral of the Cauchy distribution and compare with the estimate from the previous task. Play with the number of drawn sample points. Can you get a better estimate?

```python
import scipy.integrate as integrate
integrate.quad(lambda x: Cauchy(x, m_top, gamma_top), 170, 176)[0]
```