Designing a particle physics experiment

PROJECT III

Urs Boison

Sandro Brunner

DATA ANALYSIS II (PHY241)

Spring 2022

Contents

1 Introduction

The goal of this project is to optimize the layout of a particle physics experiment by performing simulation studies. The experiment should measure decays of charged kaons (K^+) into a charged pion (π^+) and a neutral pion (π^0). These particles are all unstable, but we are only interested in this particular decay of the kaons, which happens with a probability of 21%.

The experiment has the following setup: There is an incoming beam of K^+ along the z-axis, which can be detected by a first detector, placed at z=0. A second detector, which has a circular cross section with a diameter of 4m, is set up at z=d and used to detect the π^+ and the π^0 that are emitted in the decay of the kaons.

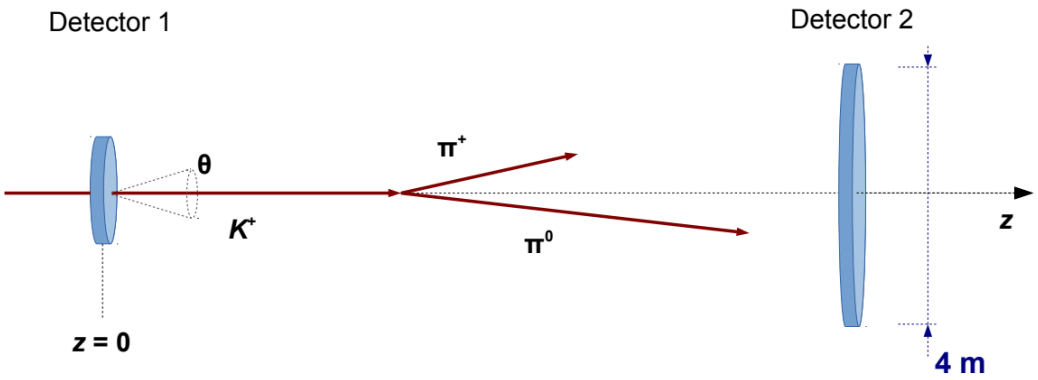


Figure 1: Sketch of the setup

Designing this experiment means concretely, that one needs to find the optimal distance, d, between the two detectors, such that the number of K^+ decays with both pions detected (by the second detector) is as large as possible. There are two antagonizing effects: on the one hand, one loses events by decreasing d, because less and less kaons decay before passing the second detector, but on the other hand one also loses events by increasing d, since then more and more pions miss the acceptance of the second detector; so there is an optimal distance, d.

To find it, we will first determine the average decay length of the kaons with the help of provided data from a previous experiment. Using this result, we will simulate the complete decay according to the arising exponential probability distribution and random decay angles in the K^+ rest frame. In a next step, we are boosting back into the laboratory frame, taking relativistic effects into account, and finally we optimize the z-position of the second detector numerically by repeating this procedure for several positions. In a last step, we will then improve the simulation by making it more realistic and taking the dispersion of the K^+ beam into account.

2 Determination of the average decay length of the K^+

The Average decay length of the Kaon can be determined using an existing data set in which the decay length of Kaons and pi mesons were measured. The experiment used to collect this data was not able to differentiate between the two particle types however, so it is a combination of the decay lengths of Kaons and pi mesons. It is known that the data consists of 84% pi meson decay lengths and of 16% of Kaon decay lengths. It is also known that the average decay length of a plus pion is 4.188 km. Since the mean of the entire dataset is easily calculated, the decay length of the Kaon can be isolated using weighted means.

$$\mu_{total} = 0.84\mu_{\pi+} + 0.16\mu_K \quad \Longrightarrow \quad \mu_k = \frac{\mu_{total} - 0.84\mu_{\pi+}}{0.16}$$

Inserting the mean of the entire dataset, which is 3604 m and the decay length of the pion which is 4188 m into this equation yields a kaon decay length of **537 m**.

To optimize the position of the detector plate, it will be necessary to loop over every possible decay length. To do this using Monte Carlo, it is necessary to produce a new dataset of exponentially distributed decay lengths, with an average of 537 m. A histogram of this dataset is depicted in Figure 2 below:

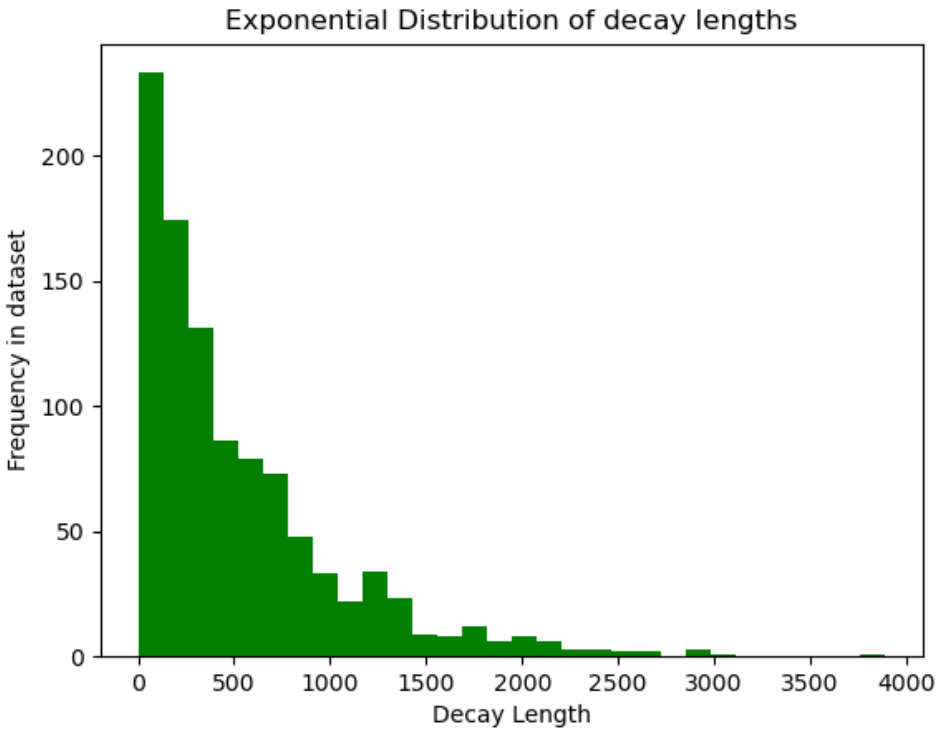


Figure 2: Exponential distribution of the Kaon decay lengths

3 Infinitely narrow beam along the z-axis

3.1 Methods

Now, as we know the mean lifetime of the kaons, we can start to simulate the experiment. For the moment, we will assume, that the beam is infinitely narrow, and ignore dispersion effects, i.e., the kaons travel exactly along the z-axis.

We generated the positions of the K^+ decay vertices according to an exponential distribution with the mean decay length that we have determined before. The decay angles of the π^+ and π^0 have also to be distributed. We will do this in the K^+ rest frame, such that the new direction of the π^+ is opposite to the one of the π^0, due to momentum conservation. In this part, we restricted ourselves to two dimensions and distributed the decay angles of the pions isotropically in the yz-plane. This should, however, not change the result significantly (or not at all), due to the symmetry of the circularly shaped detector. To receive the actual moving directions of the pions in the lab frame, we need to perform a Lorentz boost as a result of relativistic effects that arise when we deal with very big velocities (see 3.2.2).

Depending on all the different resulting moving directions of the pions combined with the corresponding decay vertex positions, one can now tell, how many of those decaying events get detected by the second detector, if it is located at a specific distance, d, from the first one. Varying that distance, we will finally find the maximally possible detections, i.e., the optimal distance for the setup of the experiment.

3.2 Results

3.2.1 Finding the speed of the Kaon

The speed of the Kaon can be determined using its decay length, l. The relation that can be used is

$$\beta\gamma c\tau = l$$

where c is the speed of light in a vacuum, β is the relativistic factor v/c, γ is the Lorentz factor $\frac{1}{\sqrt{1-\beta^2}}$ and τ is the lifetime of the Kaon (in its own inertial system).

Then $\frac{v}{c}\frac{1}{\sqrt{1-\frac{v^2}{c^2}}}c\tau = l \quad \Longrightarrow \quad v\tau = l\sqrt{1-\frac{v^2}{c^2}} \quad \Longrightarrow \quad v^2\tau^2 = l^2 - \frac{l^2v^2}{c^2}$

And finally $v = \sqrt{\frac{l^2}{\tau^2+\frac{l^2}{c^2}}}$

Inserting numbers, we obtain $\mathbf{v_{K^+}}$ **= 299785292 m/s** which is **0.99997609c.**

This is a highly relativistic speed, so it will be necessary to boost the four vectors of the pi mesons from the reference frame of the Kaon to the reference frame of the detector (lab frame).

3.2.2 Four-vectors in the Kaon Reference Frame and Lorentz boost

In the kaon reference frame, during the decay, a π^+ and a π^0 meson is produced such that they travel in opposite directions (momentum conservation). The four-vector of the π^+ meson is defined as $(E,\ p_x,\ p_y,\ p_z)$, where the energy, E, is given by:

$$E = \sqrt{m^2 + |p|^2}$$

The four-vector of the π^0 meson will then be in the opposite direction with the corresponding energy and momenta (its mass is slightly different to the one of the π^+).

In general, the directions of these pi mesons are randomly distributed in space. To find the trajectory of the pi mesons (still in the rest frame of the Kaon), it is necessary to know their momentum. The speed of the π^+-mesons can be determined from the equation derived earlier, using the given decay length of l = 4.188 km:

$v_{\pi^+} = \sqrt{\frac{l^2}{\tau^2 + \frac{l^2}{c^2}}}$ = 299791937 m/s = 0.99999826 c

This speed is likewise highly relativistic. Hence, the momentum must be calculated using the relativistic momentum equation:

$$p = \gamma m v$$

From momentum conservation, we find thus v_{π^0} = 299791970 m/s = 0.99999837 c

Now, the trajectories of the pions after the decay can be shown as the four-vectors $(E,\ p_x,\ p_y,\ p_z)$, where

$$p_x = 0\,, \quad p_y = p\,sin(\theta)\,, \quad and \quad p_z = p\,cos(\theta)$$

where θ are the uniformly randomly distributed angles. Recall that we reduced the problem to 2 dimensions and generated all decay angles in the yz-plane (=> p_x = 0) without loss of generality as already mentioned in 3.1. In this way the pions will only scatter along a line in the y axis.

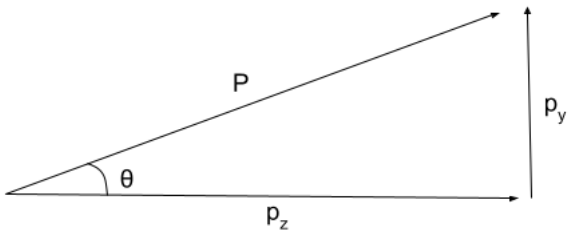


Figure 3: Momentum after decay

The four vectors can now be boosted into the lab reference frame by multiplying them with the Lorentz matrix, L, for a boost along the z-axis. Such a Lorentz matrix is defined as

$$L = \begin{bmatrix} \gamma & 0 & 0 & \beta\gamma \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \beta\gamma & 0 & 0 & \gamma \end{bmatrix}$$, where γ and β are defined using the speed of the kaon determined above.

3.2.3 Detections and optimal distance

Since we are now back in the lab frame, it remains to check for each pion, whether it gets detected or not. This depends of course on the decay vertex position, on the decay angle and on the distance between the two detectors, d, in which we are actually interested. If d < decay_vertex_position, we can preclude a detection directly, because the Kaon didn't even decay before passing the detector. Otherwise, the problem reduces to the following condition: there is a detection if and only if the decay angle (which changed during the Lorentz-boost!) of both pions in the lab frame (with respect to the z-axis) is smaller than the solid angle

$$\theta_{max} = \arctan\left(\frac{2}{d - decay\_vertex\_position}\right)$$

which depends on d and on where the Kaon decayed and can be found by simple trigonometric considerations. (The radius of the detector is 2m.)

We are now able to plot the number of detections for several different values of d by using this condition and looping over all the generated decay vertex positions, each paired with a generated angle:

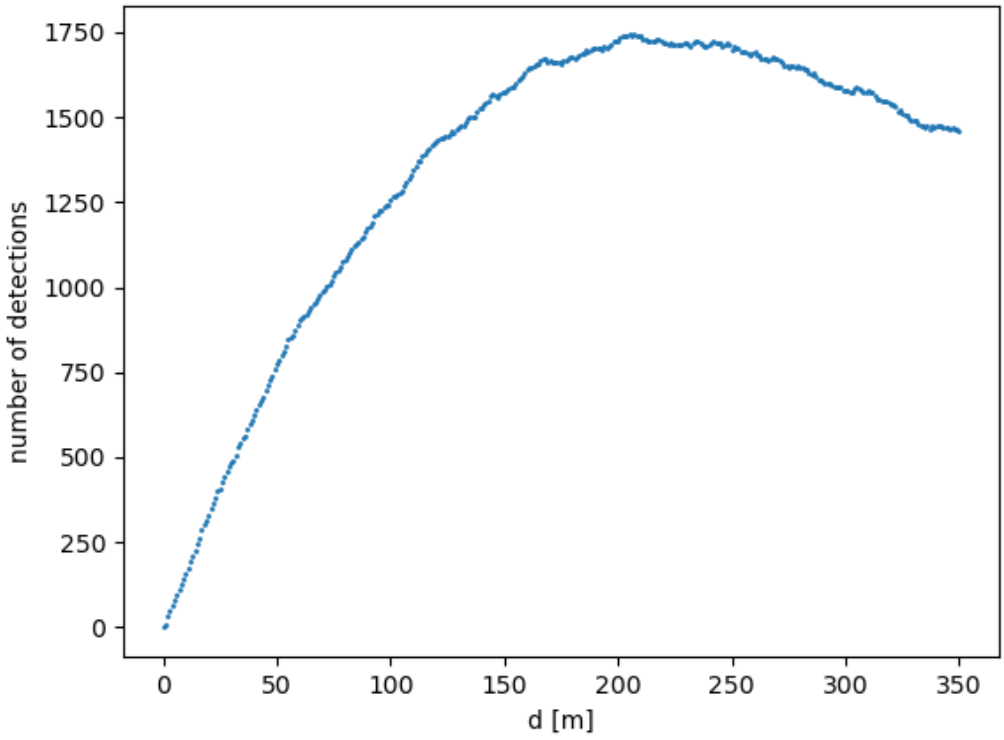


Figure 4: Number of detections for different distances between the detectors

For bigger distances, the development of the curve would look like this:

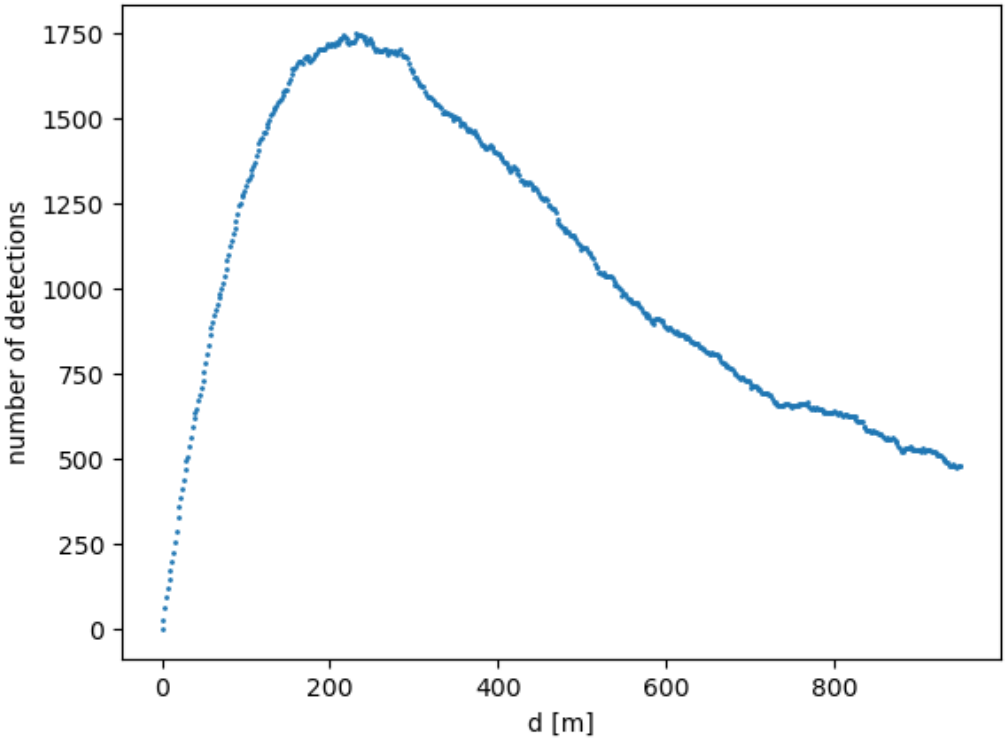


Figure 5: Long-distance development

However, we are interested in the maximum of the curve, so let's zoom in:

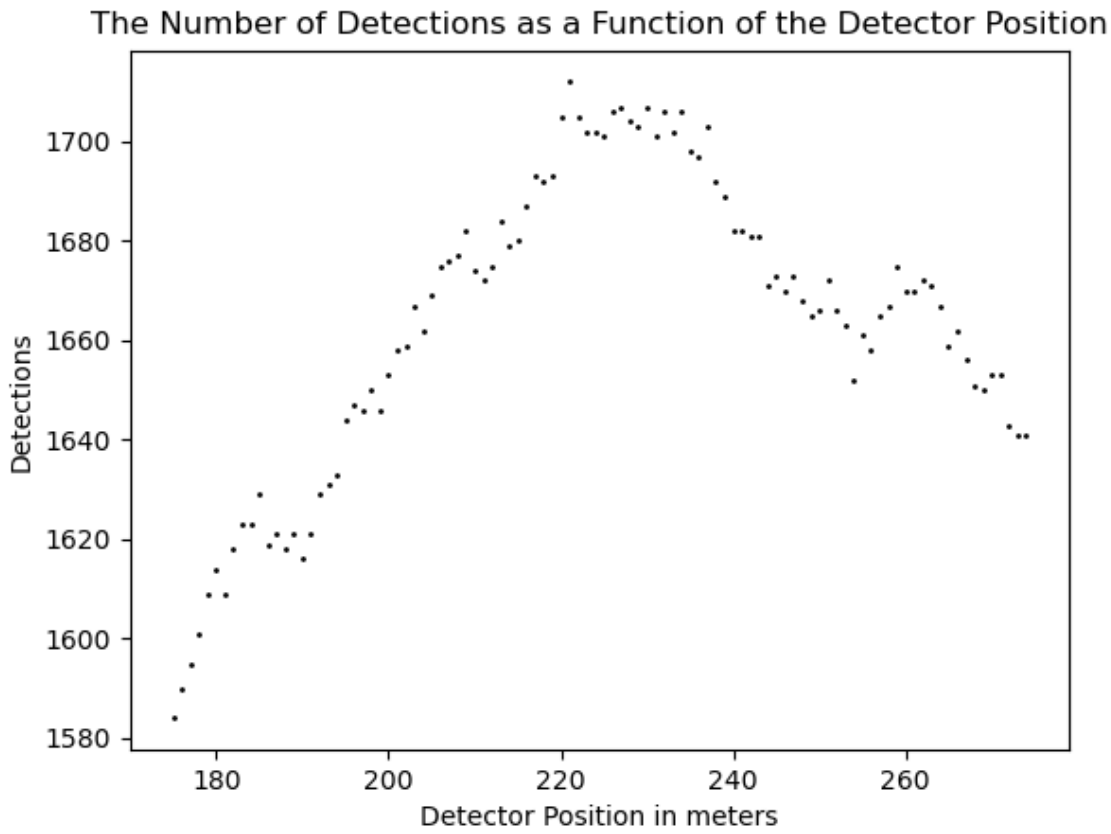


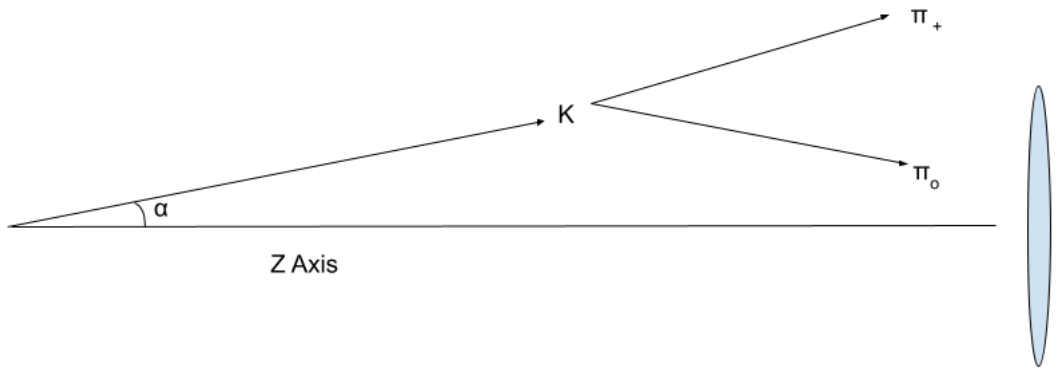
Figure 6: Zoom-in on peak

Since this maximum arises from many random variables, it will be slightly different every time one runs the program. Therefore, we can simply calculate the mean and standard deviation of the optimal distance, d, by running the program several times and reading off the maximum each time (or of course by writing a corresponding code). Finally, we end up with the following optimal distance between the detectors:

d = 223 ± 13 m

4 Divergent beam

In this part, it is no longer assumed that the kaon travels in a straight path along the z axis. Instead, it is assumed that the kaon leaves the source at an angle α.



To optimize the number of detections, it is necessary to additionally loop over the possible angles that the kaon could take. To do this, a further dataset of normally (Gaussian) distributed angles was created. A histogram of this data is shown in Figure 7 below:

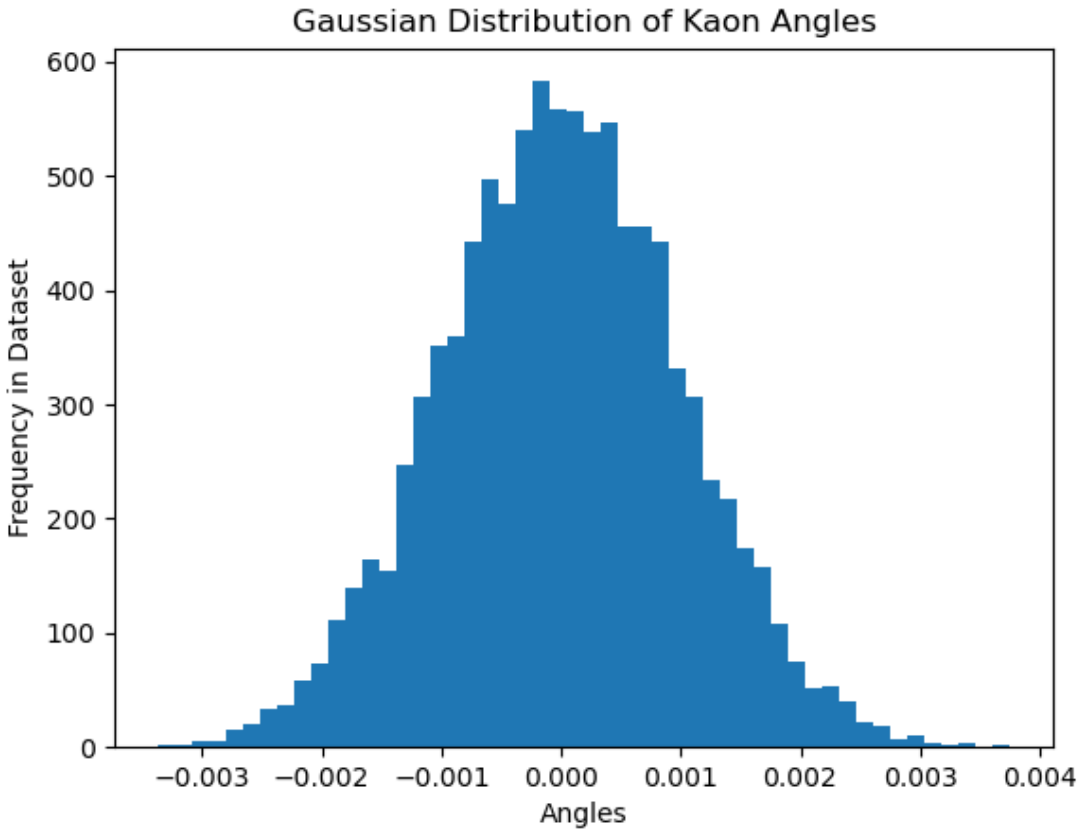


Figure 7

The four-vectors of the pions in the reference frame of the Kaons are still determined using the uniformly distributed angles (θ) as before, and the two pions are still produced moving in opposite directions. However, now that the kaons are not moving in a straight line along the z-axis it is necessary to use a more general form of the Lorentz matrix:

$$L = \begin{bmatrix} \gamma & -\gamma v_x/c & -\gamma v_y/c & -\gamma v_z/c \\ -\gamma v_x/c & 1+(\gamma-1)\frac{v_x^2}{v^2} & (\gamma-1)\frac{v_x v_y}{v^2} & (\gamma-1)\frac{v_x v_z}{v^2} \\ -\gamma v_y/c & (\gamma-1)\frac{v_y v_x}{v^2} & 1+(\gamma-1)\frac{v_y^2}{v^2} & (\gamma-1)\frac{v_y v_z}{v^2} \\ -\gamma v_z/c & (\gamma-1)\frac{v_z v_x}{v^2} & (\gamma-1)\frac{v_z v_y}{v^2} & 1+(\gamma-1)\frac{v_z^2}{v^2} \end{bmatrix}$$

Here, γ is the Lorentz factor for the kaon, while $v_y = vsin(\alpha),\ and\ v_z = vcos(\alpha)$

Here too, the problem is restricted to two dimensions so $v_x = 0$.

In this case the Lorentz matrix becomes:

$$L = \begin{bmatrix} \gamma & 0 & -\gamma v_y/c & -\gamma v_z/c \\ 0 & 1 & 0 & 0 \\ -\gamma v_y/c & 0 & 1+(\gamma-1)\frac{v_y^2}{v^2} & (\gamma-1)\frac{v_y v_z}{v^2} \\ -\gamma v_z/c & 0 & (\gamma-1)\frac{v_z v_y}{v^2} & 1+(\gamma-1)\frac{v_z^2}{v^2} \end{bmatrix}$$

The four vectors of the pions were then multiplied by this matrix to boost into the lab reference frame. Using these boosted four vectors it was possible to calculate the number of detections as a function of the decay length as in the previous section. When plotted, the result is similar to the previous case where the kaons were restricted to the z axis, as shown in figure 8 below:

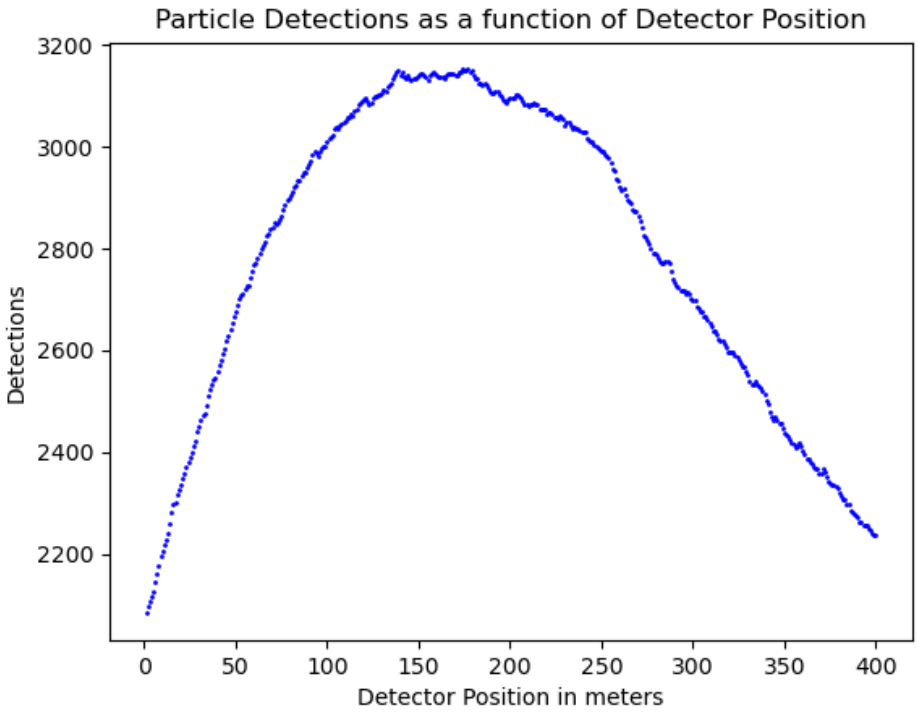


Figure 8

As In the previous section, we ran the simulation 25 times to obtain an average optimal detector position of **169 meters** with a standard deviation of **13 meters**

Comparing this value with the optimal detector position for kaons constrained to the z axis (223m), we see that it is necessary to move the detector closer to the source when the kaons have an angular divergence, which is what we expected.

5 Appendix

5.1 Code to part 2

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

decay_length_data = np.loadtxt('dec_lengths.txt')
total_mean = np.mean(decay_length_data)

'''
plt.hist(decay_length_data, bins=1000)
plt.show()
'''


def kaon_mean_decay_length(u, u_pi):
    return (u - 0.84*u_pi)/0.16


print(kaon_mean_decay_length(total_mean, 4188))
print(total_mean)
print(4188)
print('')


# to check
a = list(np.random.exponential(scale=4188, size=84000))
b = list(np.random.exponential(scale=537, size=16000))
c = a + b

print(np.mean(b))
print(np.mean(c))
print(np.mean(a))

'''
plt.hist(c, bins=1000)
plt.show()
'''
```

5.2 Code to part 3

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

data_size = 10000

pi = np.pi
mean_decay_length = 537
kaon_lifetime = 1.2385*(10**-8)
pion_lifetime = 2.6033*(10**-8)
kaon_mass = 8.837*(10**-28)
pi_plus_mass = 2.498*(10**-28)
pi_zero_mass = 2.416*(10**-28)
```

```
c = 299792458

decay_vertex_positions = np.random.exponential(scale=mean_decay_length,
size=data_size)
decay_angles = np.random.uniform(low=0, high=2*pi, size=data_size)


#plt.hist(decay_vertex_positions, bins=1000)
#plt.show()

#plt.hist(decay_angles, bins=1000, color = 'green')
#plt.show()



def velocity(d, T):
    numerator = d
    denominator = np.sqrt((T**2) + ((d**2)/c**2))
    return numerator/denominator


kaon_velocity = velocity(mean_decay_length, kaon_lifetime)
pi_velocity = velocity(4188, pion_lifetime)
print(pi_velocity)
print(pi_velocity/c)


β = kaon_velocity/c
γ = 1/np.sqrt(1-(β**2))

pi_β = pi_velocity/c
pi_γ = 1/np.sqrt(1-(pi_β**2))

# print(γ)
# print(β)

lorentz_matrix = np.array([[γ, 0, 0, γ*β], [0, 1, 0, 0], [0, 0, 1, 0],
[γ*β, 0, 0, γ]])


def detections(detector_position):
    sumk = 0
    for i in range(len(decay_vertex_positions)):
        p_plus = pi_γ * pi_plus_mass * pi_velocity
        p_zero = pi_γ * pi_zero_mass * pi_velocity
        if detector_position < decay_vertex_positions[i]:
            continue
        else:
            max_angle = np.arctan(2/(detector_position -
decay_vertex_positions[i]))

            pi_plus_four_vector = [np.sqrt((pi_plus_mass**2)+(p_plus**2)),
0, p_plus * np.sin(decay_angles[i]), p_plus * np.cos(decay_angles[i])]
            pi_zero_four_vector = [np.sqrt((pi_zero_mass**2)+(p_zero**2)),
0, -p_zero * np.sin(decay_angles[i]), -p_zero * np.cos(decay_angles[i])]

            boosted_pi_plus_four_vector =
lorentz_matrix.dot(pi_plus_four_vector)
            boosted_pi_zero_four_vector =
lorentz_matrix.dot(pi_zero_four_vector)

            pi_plus_angle = np.arccos(boosted_pi_plus_four_vector[-1] /
```

```
np.sqrt(boosted_pi_plus_four_vector[-1]**2 +

boosted_pi_plus_four_vector[-2]**2 +

boosted_pi_plus_four_vector[-3]**2))
            pi_zero_angle = np.arccos(boosted_pi_zero_four_vector[-1] /
np.sqrt(boosted_pi_zero_four_vector[-1]**2 +

boosted_pi_zero_four_vector[-2]**2 +

boosted_pi_zero_four_vector[-3]**2))

            if pi_plus_angle < max_angle and pi_zero_angle < max_angle:
                sumk += 1
    return sumk


# print(detections(100))


detector_positions = np.linspace(175, 274, 100)
number_of_successes = []

for detector_position in detector_positions:
    number_of_successes.append(detections(detector_position))

plt.plot(detector_positions, number_of_successes, 'o', markersize=1, color
= 'black')
plt.title('The Number of Detections as a Function of the Detector
Position')
plt.xlabel('Detector Position in meters')
plt.ylabel('Detections')
plt.savefig('results')
plt.show()
```

5.3 Code to part 4

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

data_size = 10000

pi = np.pi
mean_decay_length = 537
kaon_lifetime = 1.2385*(10**-8)
kaon_mass = 8.837*(10**-28)
pion_lifetime = 2.6033*(10**-8)
pi_plus_mass = 2.498*(10**-28)
pi_zero_mass = 2.416*(10**-28)
c = 299792458

decay_vertex_positions = np.random.exponential(scale=mean_decay_length,
size = data_size)
decay_angles = np.random.uniform(low=0, high=2*pi, size = data_size)
kaon_angles = np.random.normal(0, 1/1000, 10000)
```

```
def velocity(d,T,):
    numerator = d
    denominator = np.sqrt((T**2) + ((d**2)/c**2))
    return numerator/denominator

kaon_velocity = velocity(537, kaon_lifetime)
pion_velocity = velocity(4188, pion_lifetime)

β = kaon_velocity/c
γ = 1/np.sqrt(1-(β**2))

print(γ)
print(β)

B = pion_velocity/c
G = 1/np.sqrt(1-(B**2))

boosted_plus_four_vectors = []
boosted_zero_four_vectors = []
for i in range(data_size):
    p_x = 0
    p_y = G * pi_plus_mass * pion_velocity * np.sin(decay_angles[i])
    p_z = G * pi_plus_mass * pion_velocity * np.cos(decay_angles[i])
    p = np.sqrt((p_x**2) + (p_y**2) + (p_z**2))
    E = np.sqrt((pi_plus_mass**2)+(p**2))

    pi_plus_four_vector = np.array(list([E, p_x, p_y, p_z]))
    pi_zero_four_vector = np.array(list([E, p_x, -p_y, -p_z]))

    v_y = kaon_velocity * np.sin(kaon_angles[i])
    v_z = kaon_velocity * np.cos(kaon_angles[i])
    v = kaon_velocity

    lorentz_matrix = np.array([[γ, 0, -(γ * v_y) / c, -(γ * v_z) / c],
                               [0, 1, 0, 0],
                               [-(γ * v_y) / c, 0,
                                1 + ((γ - 1) * ((v_y ** 2) / (v ** 2))),
                                (γ - 1) * ((v_y * v_z) / (v ** 2))],
                               [-(γ * v_z) / c, 0,
                                (γ - 1) * ((v_z * v_y) / (v ** 2)),
                                1 + (γ - 1) * ((v_z ** 2) / (v ** 2))]])


    boosted_pi_plus_four_vector = lorentz_matrix.dot(pi_plus_four_vector)
    boosted_pi_zero_four_vector = lorentz_matrix.dot(pi_zero_four_vector)
    boosted_plus_four_vectors.append(boosted_pi_plus_four_vector)
    boosted_zero_four_vectors.append(boosted_pi_zero_four_vector)


detector_positions = np.linspace(0,400, 400)
number_of_successes = []


def detections(detector_position):
    successes = 0
    for A in range(data_size):
        m = pi_plus_mass
        p_z = (boosted_plus_four_vectors[A][3])
        p_y = (boosted_plus_four_vectors[A][2])
        v_z = p_z/m
        time = abs(detector_position - decay_vertex_positions[A])/v_z
```

```
        v_y = p_y/m
        d_y = v_y * time
        if abs(d_y) < 2:
            mm = pi_zero_mass
            pz = (boosted_zero_four_vectors[A][3])
            py = (boosted_zero_four_vectors[A][2])
            vz = pz/mm
            t = abs(detector_position - decay_vertex_positions[A])/vz
            vy = py/mm
            dy = vy * t
            if abs(dy) < 2:
                successes = successes + 1
    return successes

for detector_position in detector_positions:
    number_of_successes.append(detections(detector_position))

plt.plot(detector_positions, number_of_successes, 'o', markersize=1,
color='blue')
plt.title('Particle Detections as a function of Detector Position')
plt.xlabel('Detector Position in meters')
plt.ylabel('Detections')
plt.savefig('divergent kaons')
plt.show()
```