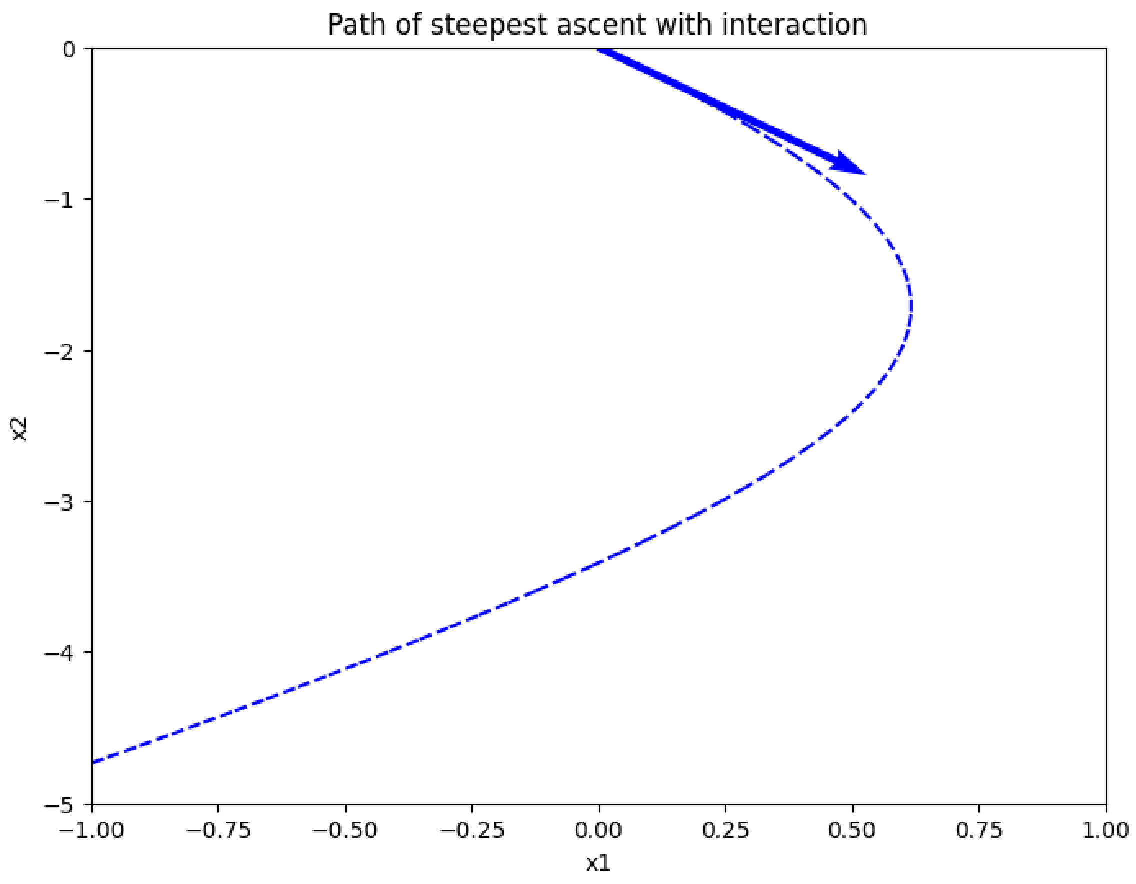
QUESTION 3 :

- The path of steepest ascent is usually computed assuming that the model is truly first order; that is, there is no interaction. However, even if there is interaction, steepest ascent ignoring the interaction still usually produces good results. To illustrate, suppose that we have fit the model
  ##### $y' = 20 + 5x1 - 8x2 + 3x1\ x2$
- Draw the path of steepest ascent that you would obtain with the interaction included in the model. (Hint: x1 on x-axis and x2 on Y-axis)

In [18]:

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

# define the response function
def response(x1, x2):
    return 20 + 5*x1 - 8*x2 + 3*x1*x2

# define the partial derivatives
def partial_derivatives(x1, x2):
    dy_dx1 = 5 + 3*x2
    dy_dx2 = -8 + 3*x1
    return dy_dx1, dy_dx2

# define the initial point and step size
x1_start, x2_start = 0, 0
step_size = 0.1

# calculate the unit vector in the direction of steepest ascent
dy_dx1, dy_dx2 = partial_derivatives(x1_start, x2_start)
grad_vector = np.array([dy_dx1, dy_dx2])
unit_vector = grad_vector / np.linalg.norm(grad_vector)

# plot the path of steepest ascent
plt.figure(figsize=(8, 6))
plt.xlabel('x1')
plt.ylabel('x2')
plt.title('Path of steepest ascent with interaction')
plt.xlim(-1, 1)
plt.ylim(-5,0)
plt.quiver(x1_start, x2_start, unit_vector[0], unit_vector[1], angles='xy', scale_units
='xy', scale=1, color='blue')

# take steps in the direction of steepest ascent and plot them
for i in range(100):
    x1_next, x2_next = x1_start + step_size*unit_vector[0], x2_start + step_size*unit_v
ector[1]
    plt.plot([x1_start, x1_next], [x2_start, x2_next], color='blue', linestyle='dashe
d')
    x1_start, x2_start = x1_next, x2_next
    dy_dx1, dy_dx2 = partial_derivatives(x1_start, x2_start)
    grad_vector = np.array([dy_dx1, dy_dx2])
    unit_vector = grad_vector / np.linalg.norm(grad_vector)

plt.show()
```


Path of steepest ascent with interaction
0
−1
−2
−3
−4
−5
x2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x1