Chapter 1

Numerical Double Integration

In an earlier chapter, we gave an example of the multivariate normal distribution: the mass and shell diameter of a population of snails.

Starting with a table of measurements, we estimated that the mean vector μ was $\begin{bmatrix}4.25\text{g} & 2.35\text{cm}\end{bmatrix}$.

We then computed the covariance matrix:

$$\Sigma = \begin{bmatrix}1.33 & 0.443\\ 0.443 & 0.416\end{bmatrix}$$

Plotting the data points, the mean, and the equal-density contours looked like this:

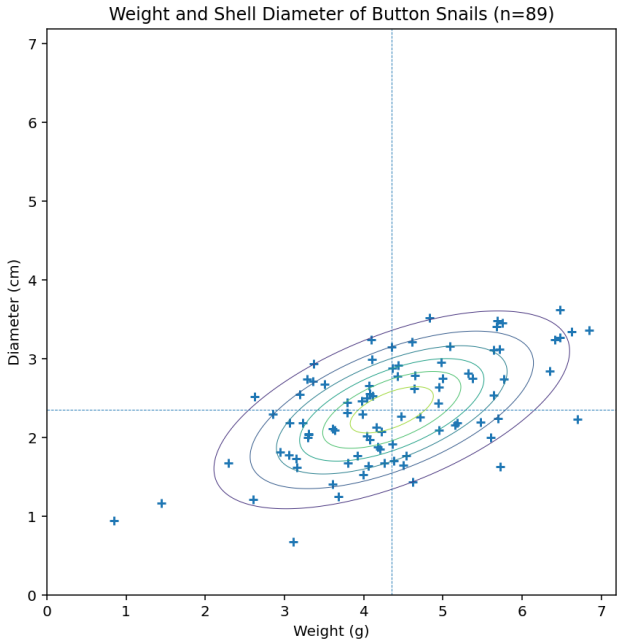

Figure 1.1: Equal density contours plotted with the snail data.

We have a great formula for computing the probability density for any mass/diameter combination:

$$p(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}-\boldsymbol{\mu})\right)$$

Can you answer the following question? "If I pick a random snail off the floor of the ocean, what is the probability that its mass is between 3 and 4 grams and its diameter is between 1.5 and 2.0 centimeters?"

If we think of the probability density as a surface, this question is really "What is the volume under the surface in the rectangular patch $3 <= x_1 <= 4$ and $1.5 <= x_2 <= 2.0$?" Which you should recognize as a double integration problem:

$$P = \int_{1.5}^{2.0} \int_{3}^{4} \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}-\boldsymbol{\mu})\right) dx_1 dx_2$$

Sadly, however, no one has ever been able to find the antiderivative of the multivariable probability density function, so no one can solve this problem.

Instead, we use Reimmann sums to find an approximate solution. This is known as *numerical integration*.

(After spending so much time learning the techniques for integration, it may be disappointing to hear this: For a lot of real-world problems, there is no way to find an antiderivative, so we end up doing numerical integration much more often than most people realize.)

1.1 Reimann Sums on 2-Dimensional Domains

When doing Reimann sums on function that takes a single real number, you summed the area of the rectangles under the function to approximate the integral:

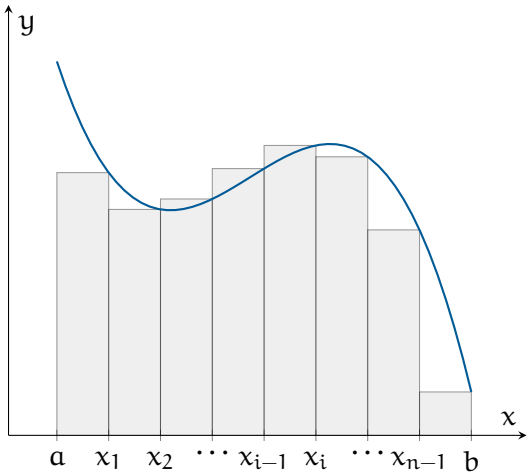

Figure 1.2: A representative function divided into n rectangles of equal width, with rectangle height determined by the right endpoint of the subinterval

Here you are finding the volume under a function that takes two variables (the probability density is based on the mass and diameter of the shell).

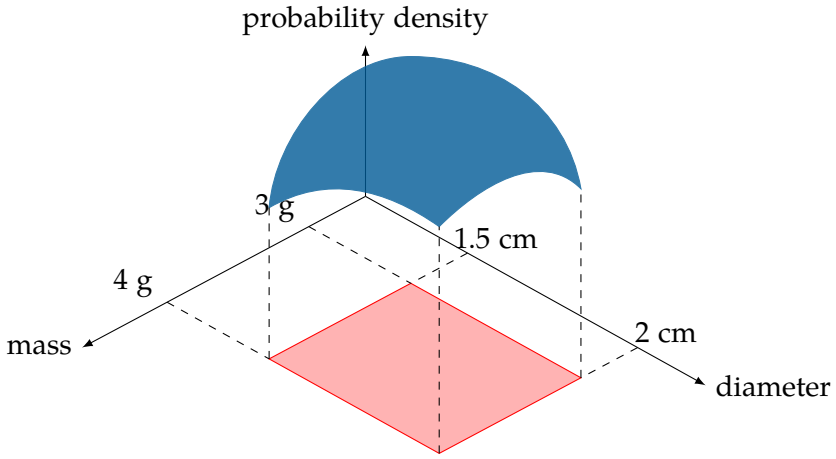

Figure 1.3: The probability is the volume under a surface for a region

To do the Reimann sum, we can break the range of the mass into n_1 equally sized intervals and the range of the diameter into n_2 equally sized intervals. For the diagram, we will just break each range into three, but you will get more accurate estimate as the intervals get smaller.

Now you will be calculating the volume of rectangular solids and summing up those volumes. Here are a few of the rectangular volumes:

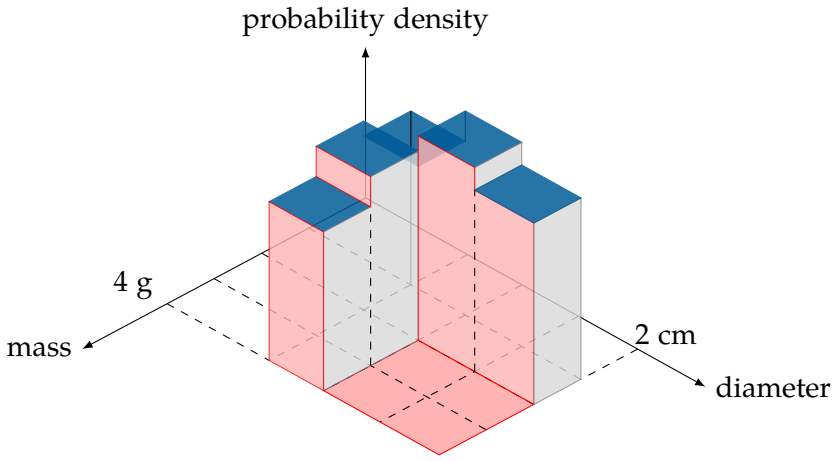

Figure 1.4: Reimann Sum

1.2 Numerical Integration in Python

Here's the code:

```
import numpy as np
from scipy.stats import multivariate_normal

# pip3 install scipy
weight_lower_limit = 3.0
weight_upper_limit = 4.0
weight_slices = 100

diameter_lower_limit = 1.5
diameter_upper_limit = 2.0
diameter_slices = 100

# What's the average weight and diameter
mean_vector = np.array([4.35559489,2.3526593 ])
print(f"Mean [weight, diameter] = {mean_vector}")

# Do heavier snails tend to have bigger shells?
covariance_matrix = np.array([[1.33099714,0.44309754],
 [0.44309754,0.41603925]])
print(f"Covariance = \n{covariance_matrix}")

# Create a multivariate normal distribution
rv = multivariate_normal(mean_vector, covariance_matrix)

delta_weight = (weight_upper_limit - weight_lower_limit) / weight_slices
delta_diameter = (diameter_upper_limit - diameter_lower_limit) / diameter_slices

sum = 0.0
```

```
# Step though each different weight
for i in range(weight_slices):
    # What is the weight in the middle of this slice?
    current_weight = weight_lower_limit + (i + 0.5) * delta_weight

    for j in range(diameter_slices):
        # What is the diameter in the middle of this slice?
        current_diameter = diameter_lower_limit + (j + 0.5) * delta_diameter

        # What is the probability density there?
        prob_density = rv.pdf((current_weight, current_diameter))

        # What is the volume under that for this tiny square
        sum += prob_density * delta_weight * delta_diameter

print(
    f"\nThe probability that the weight is between {weight_lower_limit} and {weight_upper_limit}"
)
print(
    f"and that the diameter is between {diameter_lower_limit} and {diameter_upper_limit}"
)
print(f"is about {sum * 100.0:.4f}%")
```

This should get you the following output:

```
> python3 num_integration.py
Mean [weight, diameter] = [4.35559489 2.3526593 ]
Covariance =
[[1.33099714 0.44309754]
 [0.44309754 0.41603925]]

The probability that the weight is between 3.0 and 4.0
and that the diameter is between 1.5 and 2.0
is about 7.8316%
```

Appendix A

Answers to Exercises