

Lab 1

Gaussian Quadrature

Lab Objective: *Numerical quadrature is an important numerical integration technique. The popular Newton-Cotes quadrature uses uniformly spaced points to approximate the integral, but Gibbs phenomenon prevents Newton-Cotes from being effective for many functions. The Gaussian Quadrature method uses carefully chosen points and weights to mitigate this problem.*

Shifting the Interval of Integration

As with all quadrature methods, we begin by choosing a set of points x_i and weights w_i to approximate an integral.

$$\int_a^b f(x)dx \approx \sum_{i=1}^n w_i f(x_i).$$

When we do gaussian quadrature, we are required to choose a weight function $W(x)$. This function determines both the x'_i s and the w'_i s. Theoretically, the weight function determines a set of orthogonal polynomials to approximate the function f .

The weight function also determines the interval over which the integration will occur. For example, we choose the weight function as $W(x) = 1$ over $[-1, 1]$ to integrate functions on $[-1, 1]$. To calculate the definite integrate over any interval, we perform a u-substitution. This results in the following formula.

$$\int_a^b f(x)dx = \frac{b-a}{2} \int_{-1}^1 f\left(\frac{b-a}{2}z + \frac{a+b}{2}\right)dz.$$

Once we have changed the interval, we may apply quadrature to the integral from -1 to 1 and then scale it appropriately to get the answer we want.

$$\int_a^b f(x)dx \approx \frac{b-a}{2} \sum_i w_i f\left(\frac{(b-a)}{2}x_i + \frac{(b+a)}{2}\right)$$

Problem 1. Let $f(x) = x^2$ on $[1, 4]$. Then $g(x)$ will be the interval-adjusted version of f on $[-1, 1]$, with $W(x) = 1$, $a = 1$, and $b = 4$. So,

$$\begin{aligned} g(x) &= f\left(\frac{b-a}{2}x + \frac{b+a}{2}\right) \\ &= \frac{9}{4}x^2 + \frac{15x}{2} + \frac{25}{4} \end{aligned}$$

and the interval-adjusted integral of $f(x)$ will be

$$\begin{aligned} G(x) &= \frac{b-a}{2} \int f\left(\frac{b-a}{2}x + \frac{b+a}{2}\right) dx \\ &= \frac{9}{8}x^3 + \frac{45}{8}x^2 + \frac{75}{8}x \end{aligned}$$

Verify that evaluating $G(1) - G(-1) = \int_1^4 f(x) dx$.

Problem 2. Write a function that will accept a function f and an interval $[a, b]$ and return a function g on $[-1, 1]$ that has the same integral (scaled by a constant) as f .

Use your function to plot $f(x) = x^2$ on $[1, 4]$ and the corresponding function $\frac{(b-a)}{2}g$ on $[-1, 1]$. Note that the functions will not look the same plotted, since they are defined over intervals with different lengths, but they integrate to the same value.

Integrating with Given Weights and Points

We now give an example of quadrature with known weights and points. We use the constant weight function $W(x) = 1$ from -1 to 1 (this weight function corresponds to the Legendre polynomials) to calculate the integral of $f(x) = \sin(x)$ from $-\pi$ to π , with 5 interpolation points.

First, we change the interval from $[-\pi, \pi]$ to $[-1, 1]$.

```
>>> import numpy as np
>>> a, b = - np.pi, np.pi

# f is the function to integrate.
>>> f = np.sin

# g is the function with the interval changed.
>>> g = lambda x: f((b - a) / 2 * x + (a + b) / 2)
```

The weights(w_i) and points at which f is evaluated (x_i) are given in order in Table 1.1. We put them into an array here.

point x_i	weight w_i
$-\frac{1}{3}\sqrt{5 + 2\sqrt{\frac{10}{7}}}$	$\frac{322-13\sqrt{70}}{900}$
$-\frac{1}{3}\sqrt{5 - 2\sqrt{\frac{10}{7}}}$	$\frac{322+13\sqrt{70}}{900}$
0	$\frac{128}{225}$
$\frac{1}{3}\sqrt{5 - 2\sqrt{\frac{10}{7}}}$	$\frac{322+13\sqrt{70}}{900}$
$\frac{1}{3}\sqrt{5 + 2\sqrt{\frac{10}{7}}}$	$\frac{322-13\sqrt{70}}{900}$

Table 1.1: Quadrature points and weights on $[-1, 1]$.

```
>>> from math import sqrt
>>> points = np.array([- sqrt(5 + 2 * sqrt(10. / 7)) / 3,
- sqrt(5 - 2 * sqrt(10. / 7)) / 3,
0,
sqrt(5 - 2 * sqrt(10. / 7)) / 3,
sqrt(5 + 2 * sqrt(10. / 7)) / 3])
>>> weights = np.array([(322 - 13 * sqrt(70)) / 900,
(322 + 13 * sqrt(70)) / 900,
128. / 225,
(322 + 13 * sqrt(70)) / 900,
(322 - 13 * sqrt(70)) / 900])
```

We now calculate the integral

```
>>> integral = (b - a)/2 * np.inner(weights, g(points))
```

Problem 3. Write a function that accepts a function f , an array of points, an array of weights, and limits of integration and returns the integral. Don't forget to adjust the interval as in the above example.

Calculating Weights and Points

Calculating an integral when the weights and points are given is straightforward. But, how are these weights and points found? There are many publications that will give tables of points for various weight functions. We will demonstrate how to find such a list using the Golub-Welsch algorithm.

The Golub-Welsch Algorithm

This Golub-Welsch algorithm builds a tri-diagonal matrix and finds its eigenvalues. These eigenvalues are the points at which a function is evaluated for Gaussian quadrature. The weights are the length of $[a, b]$ times the first coordinate of each

eigenvector squared. We note that finding eigenvalues for a tridiagonal matrix is a well conditioned, relatively painless problem. Using a good eigenvalue solver gives the Golub-Welsch algorithm a complexity of $O(n^2)$. A full treatment of the Golub-Welsch algorithm may be found at <http://gubner.ece.wisc.edu/gaussquad.pdf>.

We mentioned that the choice of weight function corresponds to a class of orthogonal polynomials. An important fact about orthogonal polynomials is that any set of orthogonal polynomials $\{u_i\}_{i=1}^N$ satisfies a three term recurrence relation

$$u_i(x) = (\gamma_{i-1}x - \alpha_i)u_{i-1}(x) - \beta_i u_{i-2}(x)$$

where $u_{-1}(x) = 0$ and $u_0(x) = 1$. The coefficients $\{\gamma_k, \alpha_i, \beta_i\}$ have been calculated for several classes of orthogonal polynomials, and may be determined for an arbitrary class using the procedure found in “Calculation of Gauss Quadrature Rules” by Golub and Welsch. Using these coefficients we may create a tri-diagonal matrix

$$J = \begin{bmatrix} a_1 & b_1 & 0 & 0 & \dots & 0 \\ b_1 & a_2 & b_2 & 0 & \dots & 0 \\ 0 & b_2 & a_3 & b_3 & \dots & 0 \\ \vdots & & & & & \vdots \\ \vdots & & & & & \vdots \\ 0 & \dots & & & b_{N-1} & \\ 0 & \dots & & b_{N-1} & a_N & \end{bmatrix}$$

Where $a_i = \frac{-\beta_i}{\alpha_i}$ and $b_i = (\frac{\gamma_{i+1}}{\alpha_i \alpha_{i+1}})^{\frac{1}{2}}$. This matrix is called the Jacobi matrix. The eigenvalues of this matrix give us the points x_i and the length of $[a, b]$ times the squares of the first entries of the corresponding eigenvectors gives the weights.

Problem 4. Write a function that will accept three arrays representing the coefficients $\{\gamma_i, \alpha_i, \beta_i\}$ from the recurrence relation above and return the Jacobi matrix.

Problem 5. The coefficients of the Legendre polynomials (which correspond to the weight function $W(x) = 1$ on $[-1, 1]$) are given by

$$\alpha_i = \frac{2i-1}{i} \quad \beta_i = 0 \quad \gamma_i = \frac{i-1}{i}$$

Write a function that accepts an integer n representing the number of points to use in the quadrature. Calculate α , β , and γ as above, calculate the Jacobi matrix, then use it to find the points x_i and weights w_i that correspond to this weight function. When $n = 5$, do they match the ones given in the first part of this lab?

Problem 6. Write a new function that accepts a function f , bounds a and b , and n for the number of points to use. Use the previously defined functions to estimate $\int_a^b f(x)dx$ using the coefficients of the Legendre polynomials.

This completes our implementation of the Gaussian Quadrature for a particular set orthogonal polynomials.

scipy.integrate

There are other techniques for finding the weights and points for a given weighting function. This is, in fact, not even the fastest method. In general practice, we use `scipy.integrate` to calculate integrals. `scipy.integrate.quadrature` offers a reasonably fast Gaussian quadrature implementation.

Another common hallmark of quadrature is that it can be used adaptively. It is common in practice to refine the points of a quadrature estimate on an interval where a function is observed to be changing rapidly. This allows for more accurate computation at a relatively low computational cost. This is the approach used by the function `scipy.integrate.quad`.

Problem 7. The standard normal distribution is an important object of study in probability and statistic. It is defined by the probability density function $p(x) = \frac{1}{\sqrt{2\pi}}e^{-x^2/2}$ (here we are assuming a mean of 0 and a variance of 1). This is a function that cannot be integrated symbolically.

The probability that a normally distributed random variable X will take on a value less than (or equal to) a given value x is

$$P(X \leq x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}}e^{-t^2/2}dt$$

This function is essentially zero for values of x that lie reasonably far from the mean, so we can estimate this probability by integrating from -5 to x instead of from $-\infty$ to x .

Write a function that uses `scipy.integrate.quad` to estimate the probability that this normally distributed random variable will take a value less than a given number x that lies relatively close to the mean. You can test your result at $x = 1$ by comparing it with the following code:

```
from scipy.stats import norm
N = norm()
N.cdf(1)
```