RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

Seminar on Numerical Libraries

# QUADRATURE

Tobias Schürg

July 9, 2015

**Abstract**

This paper gives an introduction to numerical quadrature. Besides common approaches such as the Midpoint rule, Simpson's rules and Gaussian quadrature an overview of the routines, included in numerical libraries, is given. The GNU Scientific Library and the NAG Numerical Library are considered as representative for this purpose. The aim of this work is to give an overall introduction and thus to be a central entry point when considering numerical integration.

**Keywords.** Composite Rule, Integration, Gaussian Quadrature, GSL, Midpoint Rule, Monte-Carlo Quadrature, NAGlib, Newton-Cotes Quadrature, Numerical Integration, Quadrature, Trapezoid Rule,

# Contents

# 1 Introduction

Quadrature itself is a historical term and originally meant determining the area of a plane figure by constructing a square that has the same area as the given figure. *Squaring the circle*, also known as quadrature of the circle, is beside *doubling the cube* and *trisecting the angle* the most famous of the three classical problems in Greek mathematics which was extremely influential in the development of geometry and calculus.

Quadrature is now more or less a synonym for integration, especially as applied to one-dimensional integrals and is generally used to mean numerical integration [37].

If $f(x)$ is a continuous real-valued function defined on a closed interval $[a, b]$, the area below the graph can be exactly computed with the antiderivative $F(x)$ of $f(x)$, for which $f = F'$ applies. The value of the integral on the closed interval is then determined by formula (1). [7]

$$\int_a^b f(x)\,dx = F(b) - F(a) \tag{1}$$

## 1.1 Problem Description

There are cases when an exact mathematical integration is not available (or needed) and a numerical approach becomes useful. Then, approximating a definite integral from few values of the integrand is of central interest. There are several cases when this problem occurs, the most important are described next.

### 1.1.1 The integrand is partially unknown

In many situations, such when points are obtained by sampling, the integrand $f(x)$ may be known only at a finite set of points. Of course, the data points can be interpolated by a polynomial of which an antiderivative is known and is thereby integratable but without background knowledge about the origin, this interpolated polynomial might increase the error.

### 1.1.2 The antiderivative is unknown

There are cases when the antiderivative is unknown or can only be estimated symbolically but with much more effort. An example of a function which can not be calculated easily because $f(x)$ has no elementary antiderivative is the Gaussian function does:

$$f(x) = exp(-x^2) \tag{2}$$

## 1.2 Solution

The basic idea to determine the value of an integral is to evaluate it at a finite set of points, called integration points. Thereby the number of integration points depends on the degree of the function. If a function has singularities or oscillates, its integration points need to be chosen more carefully.

Section 2 explains the mathematical background and basic techniques for numerical integration. Specialized routines from numerical libraries for one- and multi-dimensional integrals are then presented in section 3.

# 2 Numerical Integration

The basic problem in numerical integration is to compute an approximate solution to a definite integral of the form of formula 1, to a given degree of accuracy which depends on the application [37]. This section presents the main approaches to numerical integration. The choice depends mainly on the present data or function and on the results required [2, 3].

## 2.1 Manual Method - Counting Squares

One of the simpler but inexact methods for determining the area below a figure is *integration by hand*. The basic idea of this approach is to superimpose a grid with a known size on the graph of the function to be integrated and count the squares which are covered by more than 50% of the function. This approach is visualized in figure 1. The finer the grid is chosen, the more accurate is the result. Another way to improve accuracy is to use triangles or other shapes instead of squares in the border area. [2]

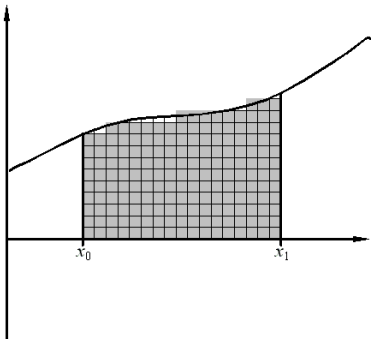


Figure 1: Manual method for determining the area below a graph. The grey squares are the ones which are by more than 50% inside and thus are counted.

## 2.2 Midpoint Rule

The idea of the midpoint rule, which is also called *rectangle method*, is to approximates the integral over an interval $[a, b]$ by calculating the area of a corresponding rectangle (see figure 2). This rectangle has width $b - a$ and height $f(c)$ with $c = \frac{a+b}{2}$ being the midpoint. Thus the integral can be approximated with equation 3. This rule is exact for polynomials with one degree, namely constant and linear functions. [38]

$$\int_a^b f(x)\,dx \approx (b-a)\cdot f(c) = (b-a)\cdot f\left(\frac{a+b}{2}\right) \tag{3}$$

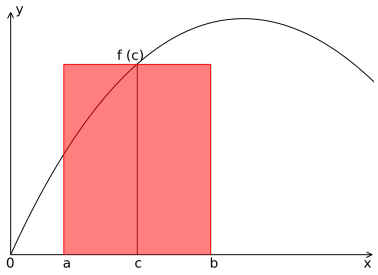


Figure 2: Midpoint Rule. Approximation of the integral by constructing a rectangle with height $f(c)$.

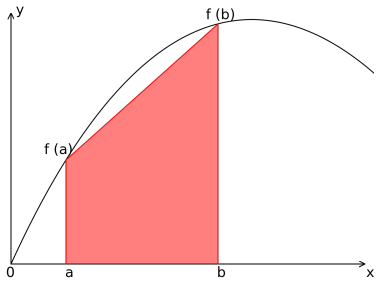


Figure 3: Trapezoid Rule. Approximation of the integral by constructing a trapezoid from $f(a)$ to $f(b)$.

## 2.3 Trapezoid Rule

The trapezoid rule uses a similar concept as the Midpoint rule. It approximates the integral of a function by the area of a trapezoid as shown in figure 3. The trapezoid itself has base $h = b-a$ and sides equal to the values of the integrand at the two endpoints $a$ and $b$. Thus, the integral of a function can be approximated by equation 4. Again, this rule is exact for constant and linear functions but provides better approximations for functions of a higher degree, compared to the midpoint rule.

$$\int_a^b f(x)\,dx \approx (b-a)\left[\frac{f(a)+f(b)}{2}\right] \tag{4}$$

## 2.4 Composite Rules

Although the previous rules are exact for constant and linear functions, one cannot expect neither the midpoint nor the trapezoid rule to give accurate results for polynomials of a higher degree over a large interval. Therefore the idea is to split the main interval into $n$ short sub-intervals and compute an approximation for each sub-interval as shown in figure 4. Summing up those partial results gives a better approximation of the integral over the whole interval. This is called the composite rule. [33]

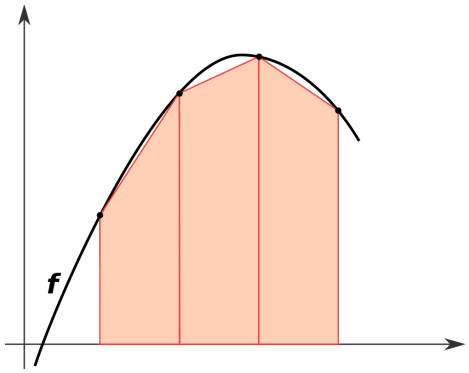


Figure 4: Trapezoid composite rule

As $n$ gets larger, the intervals become smaller and the computation of the integral gets more accurate. This idea is the spirit of the definition of the Riemann integral and the limit of this approximation as n $\to \infty$ is defined and equal to the integral $\int_a^b f$, if this Riemann integral is defined. [38]

Comparing the midpoint and trapezoid rule, there are two advantages of the former one. First, there are less function evaluations needed to approximate the integral which can matter. Second it can be used more effectively for determining the integral near an integratable singularity since endpoints are not used, only a point inside the interval. [2]

However, an analysis of Weidemann [35] shows that the trapezoid rule becomes extremely accurate when integrating periodic functions. He concludes that 'Using more sophisticated rules may not be worth the effort'.

## 2.5 Simpsons's Rule

The idea of the Simpson's rule is to use a parabola to approximate the integrand. This increases the accuracy of the approximation without a need to decrease the step size as much as it would be necessary with some lower order polynomial. The Simpson's rule provides exact results for any polynomial of degree three or less. The name of this rule comes from the English mathematician Thomas Simpson but since Kepler used similar formulas more than a century before, it sometimes is also called the Kepler's rule or Keplersche Fassregel. [39]

$$\int_a^b f(x)\,dx \approx \tfrac{b-a}{6}\left[f(a) + 4f\left(\tfrac{a+b}{2}\right) + f(b)\right] \tag{5}$$

Although this rule provides better results, approximating a function which

is highly oscillatory or which lacks derivatives at certain points, the results may be still very poor. To increase the accuracy this function can also be used on sub-intervals and is then called the *Composite Simpson's Rule*.

Formula 5 is derived by evaluating the function at the two boundary points $a$ and $b$ of the interval, plus the midpoint $m = \frac{a+b}{2}$, see figure 5.

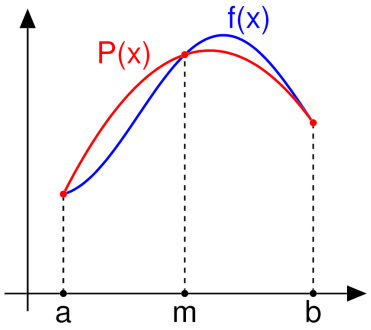


Figure 5: Simpsons Rule. Using three points to interpolate a polynomial $p(x)$ in order to approximate $f(x)$.

## 2.6 Newton-Cotes formulas

When looking at the rules which are presented so far, it can be observed, that all rules have a general form, which is shown in equation 6. The points $x_i$ are always equally spaced and the rules only differ in the number of function points which are used for the approximation. Their corresponding weights $w_i$ are derived from Lagrange basis polynomials. An advantage of equal spacing is, that values can be reused which can minimizes the computational overhead.

$$\int_a^b f(x)\,dx \approx w_0 f(x_0) + w_1 f(x_1) + \cdots + w_n f(x_n) = \sum_{i=0}^{n} w_i\, f(x_i) \tag{6}$$

Equations 7, 8 and 9 exemplary show the corresponding points $x_i$ and their weights.

$$\text{Midpoint:} \quad x = \frac{a+b}{2}, \quad w = 1 \tag{7}$$

$$\text{Trapezoid:} \quad x_0 = a, x_1 = b, \quad w_0 = w_1 = \frac{b-a}{2} \tag{8}$$

$$\text{Simpson's:} \quad x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b, \quad w_0 = w_2 = \frac{b-a}{6}, w_1 = 4*\frac{b-a}{6} \tag{9}$$

There are two types of Newton-Cotes formulas, open and closed ones. While the former only uses the points in between the interval (for example the midpoint rule), the closed formulas (such as the Simpson's rule) also use the function values at the endpoints. Newton-Cotes formulas of any higher degree $n$ can be constructed as well. [36]

## 2.7 Gaussian Quadrature

The main idea of the Gaussian quadrature is to not use equally spaced intervals at which the function is evaluated but to carefully choose the points $x_i$ and their weights to maximize accuracy. Thereby, the general form of the rule stays the same as for the Newton-Cotes rule (equation 6) but the $x_i$ and weights differ. There exist multiple forms of the Gaussian quadrature while the Gauss-Legende quadrature is the most known one. In general, the result of a rule which uses $n$ points is exact for polynomials of degree $2n-1$ or less. The Gauss-Legendre quadrature is by default defined over the interval $[-1, 1]$ so that an integral over an interval $[a, b]$ must be changed before the Gaussian quadrature rule can be applied. A change of interval can be done as shown in equation 10. However, there are other forms which are defined over other intervals. Although the computational overhead is slightly higher compared to the Newton-Cotes rules, the Gaussian quadrature rules are typically more accurate. [40]

Table 1 shows some low order Gaussian-Legendre rules up to a degree of 5.

$$\int_a^b f(x)\,dx = \frac{b-a}{2}\int_{-1}^{1} f\left(\frac{b-a}{2}x + \frac{a+b}{2}\right)\,dx \tag{10}$$

## 2.8 Monte Carlo Method

A method which does not use an approximation formula is the Monte Carlo method. Instead the integral is determined by randomly choosing $n$ points over the integration interval $[a, b]$. The integral is then approximated by the average over the function values at those random points (equation 11).

$$\int_a^b f(x)\,dx \approx \frac{b-a}{n}\sum_{i=1}^{n} f(x_i) \tag{11}$$

Table 1: Parameters for low order Gaussian-Legendre quadrature

| Number of points, $N$ | Points, $X_i$ | Weights, $w_i$ |
|---|---|---|
| 1 | $0$ | $2$ |
| 2 | $\pm\sqrt{\frac{1}{3}}$ | $1$ |
| 3 | $0$<br>$\pm\sqrt{\frac{3}{5}}$ | $\frac{8}{9}$<br>$\frac{5}{9}$ |
| 4 | $\pm\sqrt{\frac{3}{7}-\frac{2}{7}\sqrt{\frac{6}{5}}}$<br>$\pm\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}$ | $\frac{18+\sqrt{30}}{36}$<br>$\frac{18-\sqrt{30}}{36}$ |
| 5 | $0$<br>$\pm\frac{1}{3}\sqrt{5-2\sqrt{\frac{10}{7}}}$<br>$\pm\frac{1}{3}\sqrt{5+2\sqrt{\frac{10}{7}}}$ | $\frac{128}{225}$<br>$\frac{322+13\sqrt{70}}{900}$<br>$\frac{322-13\sqrt{70}}{900}$ |

The advantage of this approach is its easy implementation and extensibility, especially on multidimensional integrals. However, the computation time is higher compared to other approximation formulas. Moreover, Monte Carlo does not always provide 100% correctness but in general the results will be correct. [32, 9]

## 2.9 Multidimensional Integrals

When trying to estimate the value of multidimensional integrals, a distinction between cases of low and high dimensionality has to be made [34]. Where the number of dimensionalities is lower than 4 or 5, one- dimensional methods, such as the ones mentioned above, may be applied to each dimension, according to some suitable strategy. However, special techniques are needed for integrals of higher dimensionality since the number of evaluation rises very rapidly with the number of dimensions. Therefore non deterministic approaches such as the *Monte-Carlo* method, which randomly chose the points at which the integrand is evaluated, can be used.

# 3 Quadrature Methods

This section will give an overview of routines available in numerical libraries to perform quadrature. For reasons of simplicity only the GNU Scientific Library (GSL)[4] and the NAG Numerical Library (NAGLib)[34] are considered. These routines are classified into three categories: one-dimensional, two-dimensional and multidimensional quadrature. Furthermore, another distinction between adaptive and non-adaptive routines can be made.

There exist general purpose routines for people who do not care about computation time or are not willing to do any analysis of the integral. However, if efficiency is needed, there exist a lot of methods which accept additional parameters for specifying peaks, singularities or other problematic points. Also desired relative and absolute error bounds for the approximation can be specified.

The final choice of a routine should be based amongst others on the type of interval, number of integrals, accepted error and available computation time.

## 3.1 One-Dimensional Quadrature

This section covers routines for numerical integration in one dimension. Both considered libraries are largely based on *QUADPACK*, a Fortran library for numerical integration of one-dimensional functions by Piessens et al.[31]. There are nine routines for automatic quadrature included in QUADPACK which are described in the scope of this section.

The naming conventions for the algorithms in QUADPACK use the following naming scheme:

Q - Quadrature routine

N - Non-adaptive integrator
A - Adaptive integrator

G - General integrand (user-defined)
W - Weight function with integrand

S - Singularities can be more readily integrated
P - Points of special difficulty can be supplied
I - Infinite range of integration
O - Oscillatory weight function, cos or sin
F - Fourier integral
C - Cauchy principal value

Since the GSL uses a similar naming scheme (*gsl_integration_* + *quadpack_algorithm_name*) the function names will not be stated separately for the GSL.

### 3.1.1 QNG

QNG calculates an approximation to the integral of a function over a finite interval. It is a simple and non-adaptive routine and thus only recommended for smooth functions which neither have singularities, nor height peaks nor oscillate strongly. Depending on the desired accuracy the routine uses a 10-point, 21-point, 43-point or 87-point Gaussian rule. This is the only non-adaptive automatic integrator of QUADPACK. [31, 4]

NAGLib name: nag_quad_1d_fin_smooth (d01bdc) [23]

### 3.1.2 QAG

QAG is a simple globally adaptive integrator which uses 30-point and 61-point Gauss-Kronrod rules. The integration region is divided into subintervals, and on each iteration the subinterval with the largest estimated error is bisected. As this function is based on integration rules of high order, it is especially suitable for non-singular oscillating integrands. [15, 31, 4]

NAGlib name: nag_1d_quad_osc_1 (d01skc) [15]

### 3.1.3 QAGS

The QAGS routine is one of two general purpose routines, which are suitable for use without further analysis of the integrand. While this one is for integration over a finite interval, QAGI is the corresponding one for integration over an infinite interval. QAGS is a globally adaptive routine based on 21-point Gauss-Kronrod quadrature in connection with extrapolation. This routine is suitable as a general purpose integrator, and can be used when the integrand has singularities, especially when these are of algebraic or logarithmic type. [31, 4, 12]

NAGlib name: nag_1d_quad_gen_1 (d01sjc) [12]

Another general purpose implementation, which is included in the NAGlib (nag_quad_1d_fin_gonnet_vec (d01rgc)) and is similar but differs in how the evaluation error is estimated and in how singularities are treated. [22]

### 3.1.4 QAGP

QAGP is a modified version of the QAGS routine which allows the user to supply additional *breakpoints*, points at which the function is difficult, such as internal singularities, discontinuities and other difficulties of the integrand function. By providing these breakpoints, the routine is less computational expensive than the original QAGS routine. [31, 4, 10]

NAGlib name: nag_1d_quad_brkpts_1 (d01slc) [10]

### 3.1.5 QAGI

QAGI is the second general purpose routine which is the only routine for infinite intervals. Internally the infinite interval is mapped onto the semi-open interval $(0, 1]$ using a transformation similar to equation 10. After that a similar approach as in QAGS is used, except with 15-point rather than 21-point Gauss-Kronrod quadrature. [31, 4, 13]

NAGlib name: nag_1d_quad_inf_1 (d01smc) [13]

### 3.1.6 QAWO

QAWO is designed for integrands with an oscillatory factor, $\sin(\omega x)$ or $\cos(\omega x)$. Internally an adaptive subdivision scheme is used, which is a modification of the one in QAGS. Thus this routine can also deal with singularities. [4, 31, 19]

NAGlib name: nag_1d_quad_wt_trig_1 (d01snc) [19]

### 3.1.7 QAWF

QAWF calculates an approximation to the sine or the cosine transform of a function over a semi-infinite interval using a Fourier transform. The integral is computed using the QAWO algorithm over each of the subintervals.[4, 31, 14]

NAGlib name: nag_1d_quad_inf_wt_trig_1 (d01ssc) [14]

### 3.1.8 QAWS

QAWS is an adaptive integrator which integrates a function of the form $\int_a^b g(x)w(x)\mathrm{d}x$, where the weight function $w$ may have algebraic-logarithmic singularities at

the end-points $a$ and/or $b$[31]. In order to work efficiently the algorithm requires a precomputed table of Chebyshev moments which are used on all sub-intervals which have a or b as one of their end-points. On the other sub-intervals Gauss–Kronrod (7–15 point) integration is carried out.[4, 17].

NAGlib name: nag_1d_quad_wt_alglog_1 (d01spc) [17]

### 3.1.9 QAWC

QAWC computes the Cauchy principal value of the integral of $f(x)$ over $(a, b)$, with a singularity at c. [31]

$$I = \int_a^b dx f(x)/(x-c)\mathrm{d}x \tag{12}$$

A globally adaptive bisection algorithm ensures that subdivisions do not occur at the singular point x = c. If a sub-interval contains the point x = c or is close to it then a special 25-point modified Clenshaw-Curtis rule is used to control the singularity. Further away from the singularity the algorithm uses an ordinary 15-point Gauss-Kronrod integration rule. [4]

NAGlib name: nag_1d_quad_wt_cauchy_1 (d01sqc) [18]

### 3.1.10 Gauss-Quadrature

The fixed-order Gauss-Legendre integration routines, whose rules are presented in more detail in section 2.7, provide fast integration of smooth functions with known polynomial order over a finite interval. Both libraries provide a method for approximating an integral by a n-point Gauss-Legendre rule which is exact for polynomials up to an order of $2*n-1$. Unlike the other numerical integration routines which are presented before, these routines do not accept absolute or relative error bounds. [11, 4]
The NAGlib also provides formulae for semi-infinite intervals (Gauss–Laguerre, rational Gauss) and infinite interval (Gauss–Hermite). [11]

### 3.1.11 Data Values

The function nag_1d_quad_vals (d01gac)[16] is part of the NAGlib and integrates functions which are defined by data values only. The function to integrate needs to be specified at four or more points over the whole range. The points can be unequally spaced but should be in either ascending or descending order.

## 3.2 Two-Dimensional Quadrature

The value of a two-dimensional integral can be determined by first integrating along one axis and then a along another. Additionally, NAGlib provides one specialized function, nag_quad_2d_fin (d01dac) [24], to evaluate a definite double integral, such as the one in equation 13 where a and b are constants and $\phi_1 y$ and $\phi_2 y$ are functions of the variable y. Both integrals, the inner and the outer one, are then evaluated with the method described by Patterson [30] of 'the optimum additions of points to quadrature formulae'.

$$\int_a^b \int_{\phi_1 y}^{\phi_2(y)} f(x,y)\,\mathrm{d}xy = \int_a^b F(y)\,\mathrm{d}y \quad \text{where} \quad F(y) = \int_{\phi_1 y}^{\phi_2(y)} f(x,y)\,\mathrm{d}x \quad (13)$$

## 3.3 Multidimensional Quadrature

This sections gives a short overview of the routines for multidimensional quadrature which are provided in the two considered numerical libraries. The routines and their general idea will only be mentioned briefly since the interested reader needs to become familiar with more specific or complex concepts such as number theoretic transformations, hypercubes, sparse grids, simplexes, etc. which are beyond the scope of this work.

### 3.3.1 Multidimensional Quadrature in GSL

There are no native routines for multidimensional quadrature included in the GSL. However, there exists a third-party package by Steven G. Johnson which provides adaptive multidimensional integration as extension. Two algorithms are provided.

The first recursively partitions the integration domain into smaller subdomains and applies the same integration rule to each, until convergence is achieved. 'This algorithm is best suited for a moderate number of dimensions (say, $< 7$), and is superseded for high-dimensional integrals by other methods (e.g. Monte Carlo variants or sparse grids).'

The second routine repeatedly doubles the degree of the quadrature rules until convergence is achieved. 'This algorithm is often superior to h-adaptive integration for smooth integrands in a few (<= 3) dimensions, but is a poor choice in higher dimensions or for non-smooth integrands.' [8]

### 3.3.2 Multidimensional Quadrature in NAGlib

The NAGlib provides seven methods for determining the value of a multidimensional integral which are outlined below:

- nag_quad_md_sgq_multi_vec (d01esc) [27]
  Uses sparse grids to estimate a vector $F$ of integrals over the unit hypercupe, given the vector of $f(x)$. This routine is feasible for estimating integrals of high dimensions $d \sim O(100)$

- nag_quad_md_gauss (d01fbc) [25]
  This function uses Gaussian weights and abscissas to compute the value of a multidimensional integral over a hyper-rectangle. The number of dimensions may be anything between 1 and 20.

- nag_quad_md_sphere (d01fdc) [29]
  An approximation to a definite integral in up to 30 dimensions is computed by using the method of Sag and Szekers. The integration region is an n-sphere which can also be transformed to an n-cube.

- nag_quad_md_numth_vec (d01gdc) [26]
  Uses a number theoretic approach to approximate an multidimensional definite integral in up to 20 dimensions. The result depends on the sequence of random numbers which is generated.

- nag_quad_md_simplex (d01pac) [28]
  This function integrates a multidimensional integral by computing a sequence of approximations over a n-dimensional simplex.

- nag_multid_quad_adapt_1 (d01wcc) [20]
  Estimates the integral over a hyper-rectangle with constant limits. It repeatedly subdivides each hyper-rectangle into smaller ones and applies a seventh-degree rule to it.

- nag_multid_quad_monte_carlo_1 (d01xbc) [21]
  Uses a Monte-Carlo method to approximate the integral of a function over a hyper rectangle. In the beginning the function subdivides the integration region into smaller sub-regions and estimates the integral and variance of each sub-region. The routine stops when desired accuracy has been reached or too many integral evaluations are required in the next cycle. Since this function relies on random numbers, each run may return different results.

# 4 Summary

This work shows several ways how to estimate an integral numerically. This is desired if either the integrand is partially unknown, there is no antiderivative or its computation is too expensive. At this point numerical algorithms provide an easier and approximated solution of the integral.

In the first part, the most important concepts are presented. The basic idea of these approaches is to evaluate the integrand at a finite set of points and then to sum these values with addition of a weighting function. If the function to be integrated is of a low polynomial degree, an exact solution can be obtained by the Gaussian quadrature which only requires n points to approximate a polynomial of degree $2n - 1$. However, if the function has a high degree, oscillates or contains singularities, other concepts are needed. This often results in more function evaluations and can become expensive again.

A variant to approximating the integral of a function in one step is to subdivide the interval into smaller intervals and determine the integral as a sum of all sub intervals. Especially on periodic functions, the composite midpoint rule becomes extremely accurate. The values of multidimensional intervals can also be determined by using a one-dimensional method in each dimension. This approach is only suitable for low dimensions since the number of evaluations rises exponentially with increasing degree. A technique which evaluates an integral at a random number of points are Monte-Carlo methods.

The second part gives an overview of the routines implemented in numerical libraries. As example libraries the Gnu Scientific Library and the library of the Numerical Algorithms Group are considered. In the one-dimensional case they both mostly rely on QUADPACK, a Fortran library for numerical integration of one-dimensional integrals. QAGS and QAGI are two general purpose routine which can be used without further analysis of the integrand while the former is for the use over finite intervals and the latter for infinite intervals. However, there are routines specialized in integrals of functions which have singularities or are strongly oscillating. Using such a specialized function will be less expensive in most cases since fewer conditions need to be checked.

Although the GSL does not natively include methods for multidimensional integration, there exists an extension for that purpose. NAGlib provides routines for multidimensional integration over hyper rectangles and simplexes, n-cubes and n-spheres innately. Moreover, a routine which uses a Monte-Carlo method is included as well. With the returned absolute or relative error assumptions of the accuracy can be made.

# 5 Appendix

Tutorial exercises.

$$\int_0^{\pi} \sin(x)\,dx \tag{14}$$

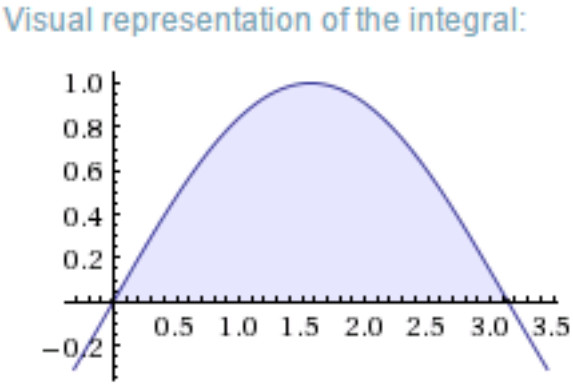


Figure 6: $sin(x)$

## 5.1 Integration with the 'simple rules'

1. Estimate the exact solution to the integral
2. Use the Midpoint, Trapezoid and Simpson's rule to approximate the integral.
3. Use the composite trapezoid rule for two and four intervals to approximate the integral.

### 5.1.1 Solution

$a = 0, b = \pi$

1. Solving the integral:

$$\int_0^{\pi} \sin(x)\,dx = -\cos(x)\Big|_{x=0}^{x=\pi} = 2 \tag{15}$$

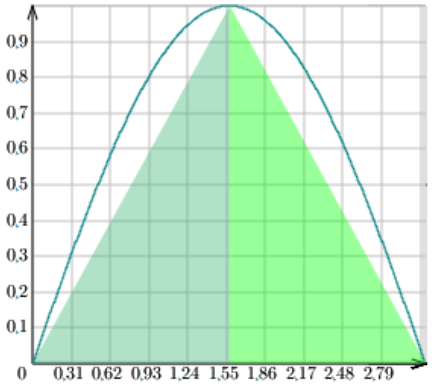


Figure 7: Trapezoid rule for n = 2

2. Solution for midpoint (16), trapezoid (17) and simpson's (18) rule to approximate the integral.

$$\int_a^b f(x)\,dx \approx (b-a)\cdot f\left(\frac{a+b}{2}\right) = (\pi)\cdot \sin\left(\frac{\pi}{2}\right) = \pi \tag{16}$$

$$\int_a^b f(x)\,dx \approx (b-a)\left[\frac{f(a)+f(b)}{2}\right] = \pi * \frac{0}{2} = 0 \tag{17}$$

$$\int_a^b f(x)\,dx \approx \tfrac{b-a}{6}\left[f(a)+4f\left(\tfrac{a+b}{2}\right)+f(b)\right] = \tfrac{\pi}{6}\left[4\sin\left(\tfrac{\pi}{2}\right)\right] = \frac{2\pi}{3} \approx 2.094 \tag{18}$$

3. Application of the composite trapezoid rule on two intervals:
$a_1 = 0, b_1 = a_2 = \pi/2, b_2 = \pi$

$$\int_0^{\frac{\pi}{2}} \sin(x)\,dx \approx \frac{\pi}{2}\left[\frac{\sin(0)+\sin(\frac{\pi}{2})}{2}\right] = \frac{\pi}{4} \tag{19}$$

$$\int_{\frac{\pi}{2}}^{\pi} \sin(x)\,dx \approx \frac{\pi}{2}\left[\frac{\sin(\frac{\pi}{2})+\sin(\pi)}{2}\right] = \frac{\pi}{4} \tag{20}$$

$$\int_0^{\frac{\pi}{2}} \sin(x)\,dx + \int_{\frac{\pi}{2}}^{\pi} \sin(x)\,dx \approx \frac{\pi}{2} = 1.571 \tag{21}$$

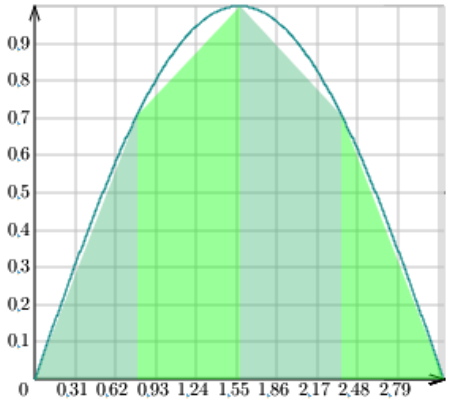


Figure 8: Trapezoid rule for n=4

4. Application of the composite trapezoid rule on four intervals:

$$\int_0^{\pi} \sin(x)\,dx = \int_0^{\frac{\pi}{4}} \sin(x)\,dx + \int_{\frac{\pi}{4}}^{\frac{\pi}{2}} \sin(x)\,dx + \int_{\frac{\pi}{2}}^{\frac{3\pi}{4}} \sin(x)\,dx + \int_{\frac{3\pi}{4}}^{\frac{\pi}{2}} \sin(x)\,dx = \cdots \approx 1.8961 \tag{22}$$

## 5.2 Gaussian Quadrature

Transform the interval from $[0, \pi]$ to $[-1, 1]$ and apply the Gaussian quadrature rule for $n = 2$

### 5.2.1 Interval Transformation

The formula from section 2.7

$$\int_a^b f(x)\,dx = \frac{b-a}{2} \int_{-1}^{1} f\left(\frac{b-a}{2}x + \frac{a+b}{2}\right) dx \tag{23}$$

Applied to our integral

$$\int_0^{\pi} \sin(x)\,dx = \frac{\pi}{2} \int_{-1}^{1} \sin\left(\frac{\pi}{2}x + \frac{\pi}{2}\right) dx \tag{24}$$

### 5.2.2 Application of the Gaussian Quadrature Rule

The tranforme interval

$$\frac{\pi}{2}\int_{-1}^{1}\sin\left(\frac{\pi}{2}x+\frac{\pi}{2}\right)\,dx \tag{25}$$

The Gaussian two-point rule:

$$\int_{-1}^{1} f(x)\,dx \approx f(-\frac{1}{\sqrt{3}}) + f(\frac{1}{\sqrt{3}})$$

Deployed:

$$\frac{\pi}{2}\int_{-1}^{1}\sin\left(\frac{\pi}{2}x+\frac{\pi}{2}\right)\,dx = \frac{\pi}{2}\left(\int_{-1}^{1} f'(x)\,dx\right) \approx \frac{\pi}{2}\left(f'(-\frac{1}{\sqrt{3}}) + f'(\frac{1}{\sqrt{3}})\right) \tag{26}$$

$$f'(x) = \sin\left(\frac{\pi}{2}x+\frac{\pi}{2}\right) \tag{27}$$

$$f'(\pm\frac{1}{\sqrt{3}}) = 0.6162 \tag{28}$$

$$=> \frac{\pi}{2}\int_{-1}^{1}\sin\left(\frac{\pi}{2}x+\frac{\pi}{2}\right)\,dx \approx \frac{\pi}{2} * 0.6162 = 1,936 \tag{29}$$

## 5.3 Comparison

Compare the results.

### 5.3.1 Solution

| | |
|---|---|
| Exact: | 2 |
| Midpoint: | $\pi$ |
| Trapezoid n = 1: | 0 |
| Trapezoid n = 2: | 1.571 |
| Trapezoid n = 4: | 1.896 |
| Simpson's: | 2.094 |
| Gaussian n = 1: | 1.936 |

The trapezoid rule for n = 1 has the worst result. However, one can see how the results get better when the interval is subdivided and more points are considered. This fact is also clearly visible when viewing the corresponding figures. The Simpson's rule is close but the Gaussian rule provides the best result while only using two points. The midpoint rule uses only one point and thus provides also a bad approximation but one can image that its performance increases when the interval is subdivided as with the trapezoid rule.

# References


[1] Thomas G. Coleman, Hillary C. Mesick, and Ronnie L. Darby. "Numerical integration". In: *Annals of Biomedical Engineering* 5.4 (1977), pp. 322–328. ISSN: 0090-6964. DOI: 10.1007/BF02367312.

[2] Stuart Dalziel. *Numerical Methods Natural Sciences Tripos 1B Lecture Notes - Numerical Integration*. 1998. URL: http://www.damtp.cam.ac.uk/lab/people/sd/lectures/nummeth98/integration.htm.

[3] Terje O. Espelid. *A test of QUADPACK and Four Doubly Adaptive Quadrature Rules*. Tech. rep. Department of Informatics University of Bergen, Norway, 2004.

[4] Mark Galassi et al. *GNU Scientific Library Reference Manual*. URL: http://www.gnu.org/software/gsl/manual/gsl-ref_17.html\#SEC292.

[5] Walter Gautschi. "The work of Philip Rabinowitz on numerical integration". In: *Numerical Algorithms* 9.2 (1995), pp. 199–222. ISSN: 10171398. DOI: 10.1007/BF02141588.

[6] Åke Björck Germund Dahlquist. "Numerical integration". In: *Numerical Methods in Scientific Computing*. Vol. 1. SIAM, 2009, pp. 521–607. ISBN: 978-0-898716-44-3. URL: http://www.siam.org/books/ot103/.

[7] Harro Heuser. *Lehrbuch der Analysis*. 16. Teubner, 2006, p. 648.

[8] Steven G. Johnson. *Cubature (Multi-dimensional integration)*. 2014. URL: http://ab-initio.mit.edu/wiki/index.php/Cubature.

[9] Kenneth L Judd and Benjamin S Skrainka. "High performance quadrature rules: how numerical integration affects a popular model of product differentiation". In: *Working Papers* (2011). ISSN: 1556-5068. DOI: 10.2139/ssrn.1870703.

[10] Library Manual. *NAG Library Function Document: nag_1d_quad_brkpts_1 (d01slc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01slc.html.

[11] Library Manual. *NAG Library Function Document: nag_1d_quad_gauss_1 (d01tac)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01tac.html.

[12] Library Manual. *NAG Library Function Document: nag_1d_quad_gen_1 (d01sjc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01sjc.html.

[13] Library Manual. *NAG Library Function Document: nag_1d_quad_inf_1 (d01smc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01smc.html.

[14] Library Manual. *NAG Library Function Document: nag_1d_quad_inf_wt_trig_1 (d01ssc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01ssc.html.

[15] Library Manual. *NAG Library Function Document: nag_1d_quad_osc_1 (d01skc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01skc.html.

[16] Library Manual. *NAG Library Function Document: nag_1d_quad_vals (d01gac)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01gac.html.

[17] Library Manual. *NAG Library Function Document: nag_1d_quad_wt_alglog_1 (d01spc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01spc.html.

[18] Library Manual. *NAG Library Function Document: nag_1d_quad_wt_cauchy_1 (d01sqc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01sqc.html.

[19] Library Manual. *NAG Library Function Document: nag_1d_quad_wt_trig_1 (d01snc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01snc.html.

[20] Library Manual. *NAG Library Function Document: nag_multid_quad_adapt_1 (d01wcc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01wcc.html.

[21] Library Manual. *NAG Library Function Document: nag_multid_quad_monte_carlo_1 (d01xbc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01xbc.html.

[22] Library Manual. *NAG Library Function Document: nag_quad_1d_fin_gonnet_vec (d01rgc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01rgc.html.

[23] Library Manual. *NAG Library Function Document: nag_quad_1d_fin_smooth (d01bdc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01bdc.html.

[24] Library Manual. *NAG Library Function Document: nag_quad_2d_fin (d01dac)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01dac.html\#ref270.

[25] Library Manual. *NAG Library Function Document: nag_quad_md_gauss (d01fbc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01fbc.html.

[26] Library Manual. *NAG Library Function Document: nag_quad_md_numth_vec (d01gdc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01gdc.html.

[27] Library Manual. *NAG Library Function Document: nag_quad_md_sgq_multi_vec (d01esc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01esc.html.

[28] Library Manual. *NAG Library Function Document: nag_quad_md_simplex (d01pac)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01pac.html.

[29] Library Manual. *NAG Library Function Document: nag_quad_md_sphere (d01fdc)*. 2015. URL: http://www.nag.co.uk/numeric/cl/nagdoc_cl25/html/d01/d01fdc.html.

[30] T. N. L. Patterson. "The optimum addition of points to quadrature formulae." In: (1967). ISSN: 0025-5718. DOI: 10.1090/S0025-5718-68-99866-9.

[31] Robert Piessens. *Quadpack : a subroutine package for automatic integration*. Springer series in computational mathematics. Catalogin based on CIP information. Berlin, New York: Springer-Verlag, 1983. ISBN: 0-387-12553-1. URL: http://opac.inria.fr/record=b1091802.

[32] C.P. Robert and G. Casella. "Monte Carlo Integration". In: *Introducing Monte Carlo Methods with R* 4.1 (2010), pp. 1–14. DOI: 10.1007/978-1-4419-1576-4.

[33] Ch Skokos and E. Gerlach. "Numerical integration of variational equations". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 82.3 (2010). ISSN: 15393755. DOI: 10.1103/PhysRevE.82.036704. arXiv: 1006.0154.

[34] Oxford The Numerical Algorithms Group Ltd. "NAG Library Chapter Introduction - D01 - Quadrature". In: *Library Manual* (2012), pp. 1–7. URL: http://www.nag.com/numeric/fl/nagdoc_fl24/pdf/D01/d01intro.pdf.

[35] J. André C Weideman. "Numerical integration of periodic functions: A few examples". In: *American Mathematical Monthly* 109.1 (2002), pp. 21–36. ISSN: 00029890. DOI: 10.2307/2695765.

[36] Wikimedia Foundation. *Wikipedia, the free encyclopedia: Newton-Cotes Formulas*. 2015. URL: https://en.wikipedia.org/wiki/NewtonCotes_formulas.

[37] Wikimedia Foundation. *Wikipedia, the free encyclopedia: Numerical Integration*. 2015. URL: http://en.wikipedia.org/wiki/Numerical_integration (visited on 04/07/2015).

[38] Wikimedia Foundation. *Wikipedia, the free encyclopedia: Rectangle method*. 2015. URL: https://en.wikipedia.org/wiki/Rectangle_method.

[39] Wikimedia Foundation. *Wikipedia, the free encyclopedia: Simpson's rule*. 2015. URL: https://en.wikipedia.org/wiki/Simpson\%27s_rule.

[40] Wikipedia. *Gaussian quadrature — Wikipedia, The Free Encyclopedia*. [Online; accessed 29-June-2015]. 2015. URL: https://en.wikipedia.org/w/index.php?title=Gaussian_quadrature&oldid=662972730.