Computational Geometry Lab: QUADRATURE

John Burkardt
Information Technology Department
Virginia Tech
http://people.sc.fsu.edu/∼jburkardt/presentations/cg_lab_quadrature.pdf

August 28, 2018

1 Introduction

This lab continues the study of Computational Geometry. We are now concerned with the idea of *quadrature*, that is, the approximation of the integral of some function $f(x, y)$ over a finite two dimensional integration domain $\mathcal{R}$.

We assume that $\mathcal{R}$ has been triangulated with a triangulation τ. If $\mathcal{R}$ is curved, then we already may expect some problems caused by the fact that the triangulated region can only approximate such curved geometry. We will for now ignore this problem.

Once we can regard the integration domain as a collection of disjoint triangles, we can reduce the original integral to the sum of integrals over triangles. If we can do a good job of integration over one triangle, then we have the key to approximating integrals over regions that can be triangulated.

2 Quadrature Rules for an Interval

The second semester of a calculus course is devoted to developing rules to determine the integral of a function $f(x)$ over a region, especially a line segment $[a, b]$. Unfortunately, there is no general procedure that can compute the integral for any function. If a value for the integral is desired, then the only option left is to try some kind of approximation.

A *quadrature rule* is a rule for approximating an integral over a specific integration region. A typical quadrature rule for the 1 dimensional case will approximate an integral using a weighted sum of function values:

$$\int_{-1}^{1} f(x)\,dx \approx \sum_{i=1}^{n} w_i\, f(x_i)$$

Here, the number n is called the *order* or the rule, the values x_i, where the function is evaluated, are called the *abscissas*, and the values w_i are known as the *weights*. Generally, the abscissas will be expected to be elements of the interval $[-1, 1]$ (that is, we don't expect an abscissa at 1.5, for instance!) and the weights will usually all be positive values.

Another feature of most quadrature rules is that they have some level of *polynomial precision* or *exactness*. A quadrature rule has precision equal to p if its approximation to the integral of $f(x)$ is exact whenever $f(x)$ is a polynomial of degree p of less. In particular, you should see that if a rule defined on $[0, 1]$ has polynomial precision 0, the weights must sum to 1!

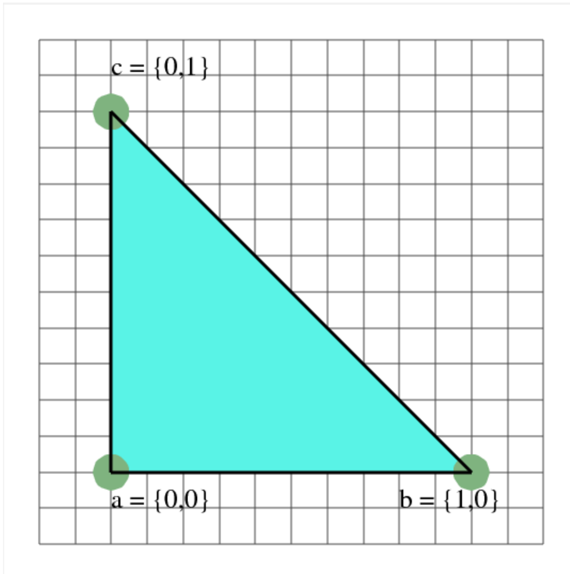

Figure 1: The unit triangle or reference triangle **Tref**.

If a quadrature problem is defined on the interval [a,b], then the quadrature rule defined on $[-1, 1]$ can be transformed to a corresponding rule on $[a, b]$ using the transformation

$$x_i \to X_i = \frac{(x_i - 1)a + (x_i + 1)b}{2}$$
$$w_i \to W_i = w_i \frac{\text{Area([a, b])}}{\text{Area}([-1, +1])} = w_i \frac{(b - a)}{2}$$

When we turn to the problem of approximating the integral of a function over a triangle, the same issues will arise, with some added complexity due to the geometry.

3 Quadrature over the Unit Triangle

A *quadrature rule* for the triangle is a rule for approximating an integral over triangle. A typical quadrature rule is specified for **Tref**, *the unit triangle*, also known as the *reference triangle*, whose vertices are {{0,0}, {1,0},{0,1}}. The rule has the form:

$$\int_{\text{Tref}} f(x, y)\, dx \approx \sum_{i=1}^{n} w_i\, f(x_i, y_i)$$

A quadrature rule for the unit triangle has precision p if its approximation to the integral of $f(x, y)$ is exact whenever $f(x, y)$ is a polynomial of total degree p of less. The total degree is the maximum exponent sum over all the monomial terms in a polynomial.

There is a formula for the exact integral of any monomial $x^q y^r$ over the unit triangle:

$$\int_{\text{Tref}} x^q y^r\, dx = \frac{q!r!}{(q + r + 2)!}$$

4 Program #1: Quadrature over the Unit Triangle

Here are examples of quadrature rules for the unit triangle, with the order N, precision P, weights W, and abscissas (X, Y):

Table 1: Quadrature Rules for the Unit Triangle.

N	P	W	X	Y
1	1	1.000000	0.333333	0.333333
3	2	0.333333	0.500000	0.000000
		0.333333	0.500000	0.500000
		0.333333	0.000000	0.500000
4	3	-0.562500	0.333333	0.333333
		0.520833	0.600000	0.200000
		0.520833	0.200000	0.600000
		0.520833	0.200000	0.200000
6	4	0.109951	0.816847	0.091576
		0.109951	0.091576	0.816847
		0.109951	0.091576	0.091576
		0.223381	0.108103	0.445948
		0.223381	0.445948	0.108103
		0.223381	0.445948	0.445948
7	5	0.225000	0.333333	0.333333
		0.125939	0.797427	0.101287
		0.125939	0.101287	0.797427
		0.125939	0.101287	0.101287
		0.132394	0.059716	0.470142
		0.132394	0.470142	0.059716
		0.132394	0.470142	0.470142

Write a program which applies these quadrature rules to integrate the function $f(x, y) = x^q\, y^r$ on the unit triangle **Tref**.

Your program should:

- read the order of the quadrature rule **N**;
- read the abscissas (x_i, y_i) and weights w_i of the quadrature rule;
- read the powers q and r of the integrand;
- compute the exact integral $I = \int_{\text{Tref}} x^q\, y^r\, dx\, dy$;
- compute Q, integral estimate from the quadrature rule.
- compute the error $E = \|Q - I\|$;
- print q, r, $q + r$, P, I, E.

If the precision of a rule is **P**, then the rule should be able to integrate exactly any monomial $x^q\, y^r$ for which $q + r \leq P$. Verify the precision claims for the quadrature rules.

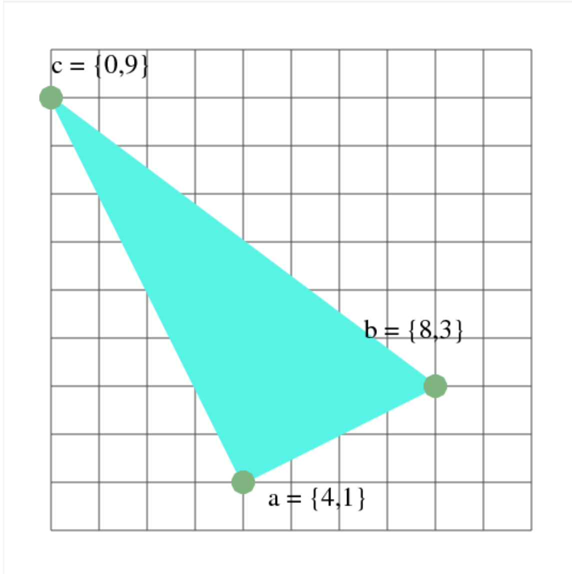

Figure 2: Example triangle #1, "Tex1".

5 Quadrature Over a General Triangle

If we are integrating over a general triangle $T = \{a, b, c\}$, then it is possible to apply a quadrature rule defined on the reference triangle, but only after we have transformed that rule to the new region:

$$(x_i, y_i) \rightarrow (X_i, Y_i) = (1 - x_i - y_i)a + x_i b + y_i c$$
$$w_i \rightarrow W_i = w_i \frac{\text{Area(T)}}{\text{Area(Tref)}} = 2\, w_i\, \text{Area(T)}$$

Consider our example triangle #1 or **"Tex1"**, whose definition is

```
{ { 4, 1 },
  { 8, 3 },
  { 0, 9 } }
```

The **Tref** point (x_2, y_2)=(0.6, 0.2) in rule #2 would map to the point (X_2, Y_2) in **Tex1** as follows:

$$X_2 = (1.0 - 0.6 - 0.2) * (4) + 0.6 * (8) + 0.2 * (0) = 5.6$$
$$Y_2 = (1.0 - 0.6 - 0.2) * (1) + 0.6 * (3) + 0, 2 * (9) = 3.8$$

Notice that this transformation is a linear mapping between the two triangles, is invertible (as long as T is not a degenerate triangle) and that the mapping takes (0,0) to a, (1,0) to b and (0,1) to c.

Because this is a linear mapping, any function $f(x, y)$ defined on **Tref** of total degree p will be transformed by the mapping to a function $F(X, Y)$ on T which is also of total degree p. This means that if a quadrature rule has precision p on the unit triangle, it will have the same precision on a general triangle under the linear mapping.

There is an inverse map from T to **Tref**:

$$X_i \rightarrow x_i = \frac{(Y_3 - Y_1)(X_i - X_1) - (X_3 - X_1)(Y_i - Y_1)}{(Y_3 - Y_1)(X_2 - X_1) - (X_3 - X_1)(Y_2 - Y_1)}$$
$$Y_i \rightarrow y_i = \frac{(X_2 - X_1)(Y_i - Y_1) - (Y_2 - Y_1)(X_i - X_1)}{(Y_3 - Y_1)(X_2 - X_1) - (X_3 - X_1)(Y_2 - Y_1)}$$

Table 2: Values of some test integrals.

(p,q)	$\int x^p y^q dxdy$
(0,0)	20.00
(1,0)	80.00
(0,1)	86.66
(2,0)	373.33
(1.1)	306.66
(0,2)	433.33
(2,1)	1333.33
(2,2)	5294.22

For instance, the point (X_2, Y_2)=(5.6, 3.8) in **Tex1** will be mapped to the point (x_2, y_2) in **Tref** as follows:

$$x_2 = \frac{(9-1)(5.6-4)-(0-4)(3.8-1)}{(9-1)(8-4)-(0-4)(3-1)} = 0.6$$
$$y_2 = \frac{(8-4)(3.8-1)-(3-1)(5.6-4)}{(9-1)(8-4)-(0-4)(3-1)} = 0.2$$

6 Program #2: Quadrature on the General Triangle

Write a program which applies the quadrature rules to integrate the function $f(x, y) = x^q\, y^r$ on a general triangle.

Your program should:

- read a triangle **T**;
- read the order of the quadrature rule **N**;
- read the abscissas (x_i, y_i) and weights w_i of the quadrature rule;
- read the powers **q** and **r**;
- compute transformed abscissas (X_i, Y_i) and weights W_i;
- apply the quadrature rule, and print the integral estimate;

Try the program on the **Tex1** triangle.

Since we are working in a general triangle, we don't have a formula for the exact integrals. Here are several values:

Try to approximate some of these values with your program.

7 Quadrature Over a Triangulation

Now suppose that we have a region $\mathcal{R}$ for which we have a triangulation $\mathcal{T} = \{T_i : 1 \leq i \leq N\}$, with the triangles T_i having disjoint interiors and whose union is $\mathcal{R}$. Suppose that we wish to estimate the integral

$$I(\mathcal{R}, f) = \int_{\mathcal{R}} f(x, y)\, dx\, dy$$

Since $\mathcal{R}$ is identical to the extent of $\mathcal{T}$, and since $\mathcal{T}$ is the disjoint sum of the triangles T_i, an integral over $\mathcal{R}$ is the sum of the integrals over the triangles:

$$I(\mathcal{R}, f) = \int_{\mathcal{T}} f(x,y)\,dx\,dy$$

$$= \sum_{i=1}^{N} \int_{T_i} f(x,y)\,dx\,dy = \sum_{i=1}^{N} I(T_i, f)$$

and, if we now apply a quadrature rule Q to approximate the integral over each triangle, we have:

$$I(\mathcal{R}, f) = \sum_{i=1}^{N} I(T_i, f) \approx \sum_{i=1}^{N} Q(T_i, f)$$

In other words, to approximate an integral over a triangulated region, we may use a quadrature rule to approximate the integral of the function over each triangle in the triangulation and sum the result.

8 Program #3: Quadrature Over a Triangulation

Write a program which estimates the integral of a function over a triangulated region by applying a quadrature rule to each triangle in the triangulation.

Your program should:

- read the number of triangles **T_Num**;
- read the triangles;
- read the order of the quadrature rule **N**;
- read the weights and abscissas of the quadrature rule;
- apply the quadrature rule to each triangle
- print the estimated value of the integral.

Use the following simple triangulation:

```
{ { {2,0}, {2,2}, {0,2} },
  { {1,0}, {2,0}, {1,1} },
  { {0,1}, {1,1}, {0,2} } }
```

This triangulation has "hanging nodes" but that won't be a problem for our calculation.

The function $f(x,y)$ to integrate is

$$f(x,y) = \sqrt{x^2 + y^2}$$

The value of this integral is 5.35637...(Thanks, Mathematica!) Run your program with quadrature rule #3 from the table.

9 Improving a Quadrature Estimate

The value returned by a quadrature rule is an estimate of an integral. Unless the integrand is a polynomial for which the rule is precise, the estimate will have a certain amount of error.

If our quadrature rule has precision p, and our integrand $f(x,y)$ is smooth enough, we would expect that the error made over triangle Δ_i is of order $C * h_i^{p+1} * \text{Area}(\Delta_i)$, where C is a bound on the integrand

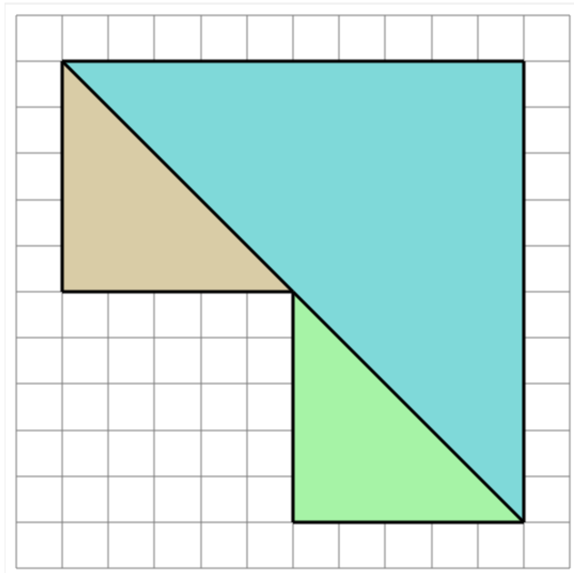

Figure 3: The triangulation to be used for the quadrature calculation.

derivatives of order $p+1$, and h_i is the length of the longest side or "characteristic length" of Δ_i. Our total error is the sum of all these errors, so it can then be estimated by

$$|\text{Error}| \leq \sum_{i=1}^{N} C * h_i^{p+1} * \text{Area}(\Delta_i) \leq C * h_{max}^{p+1} * \text{Area}(\mathcal{T}),$$

where h_{max} is the maximum value of h_i and Area($\mathcal{T}$) is the total area of the triangulated region.

By looking at the formula for the error, it seems that one way to reduce the error for an integral over a triangulation is to keep the triangulation fixed, but to use a quadrature rule of higher precision $p2 > p$. If our integrand has bounded derivatives of order $p2+1$, then our error estimate will go down because the exponent of h_{max} has increased.

A second approach would be to refine the triangulation; that is, to reduce the value of h_{max} by replace some or all of the triangles by smaller ones. A simple procedure can be used to replace any triangle of characteristic size h by 4 triangles of characteristic size $h/2$. If we refine every triangle in this way, but use the same quadrature rule as before, then p stays the same, but h_{max} has been reduced by a factor of 2 so the new error estimate is divided by 2^p. This procedure may be beneficial if the integrand has limited differentiability, or if we simply don't have access to a quadrature rule of higher precision.

If accuracy is important, it may be be desirable to estimate the size of the error, so that corrective action can be taken, if necessary. A simple way to estimate the error is to carry out the approximation process at least twice, using for the second estimate a rule with better accuracy, either by increasing the exponent p or reducing the characteristic length h_{max}. If we have two such estimates, the difference between them suggests the amount of error in our estimate. If the estimated error seems large, we may need to reduce p or h_{max} yet again, and compare our second and third results.

10 Program #4: Repeated Quadrature Over a Fixed Triangulation

Modify your program from the previous exercise. Approximate an integral using one rule, and then estimate the error by carrying out a second approximation with a better rule and taking the difference.

Your program should:

- read the number of triangles **T_Num**;
- read the triangles;
- read the order of the quadrature rule # 1: **N1**;
- read the weights and abscissas of the quadrature rule # 1;
- compute **Q1**, the first estimate;
- read the order of the quadrature rule # 2: **N2**;
- read the weights and abscissas of the quadrature rule # 2;
- compute **Q2**, the second estimate;
- print **Q1**, **Q2**, and the error estimate | **Q1-Q2** |.

Run your program on the same problem as before, but now compare quadrature rules #1 and #2, then #2 and #3, and so on up to rules #4 and #5.