Introduction to Numerical Integration

James R. Nagel (nageljr@ieee.org)

1. INTRODUCTION

Although exact analytical solutions to integrals are always desirable, such luxuries are rarely available when dealing with the real-world systems. Consequently, numerical integration has become an indispensable tool when evaluating complex designs, and it is far more common for engineers to obtain solutions numerically rather than through exact mathematical expressions. This can be especially common when processing random physical data that does not follow any sort of continuous, deterministic function. It is therefore important to gain an appreciation for the scope of numerical integration and its power to solve problems in engineering.

A typical text on numerical integration will easily contain hundreds of pages of information on various integration algorithms, but there are generally three major trade-offs to consider when choosing a particular one. The first, and most important, is obviously the accuracy of the numerical approximation. However, no integration scheme is so inaccurate that it cannot be compensated for by dividing the integration into smaller and smaller segments. Thus, the second metric to consider is the computational time required to achieve a given accuracy. This is usually measured in terms of the required function evaluations, or *fevals*, that the computer needs to perform while implementing the algorithm. Many advanced schemes exist to provide very high degrees of accuracy with few fevals, but can often require far more time just to write the code than to simply grind out on more primitive version. The complexity of the algorithm is therefore the third major trade-off to consider, and the "correct" choice will always depend on the nature of the application.

2. RIEMANN SUMS

By definition, the integral of some function $f(x)$ between the limits a and b may be thought of as the area A between the curve and the x-axis. This configuration is shown in Figure 1, and is written mathematically as

$$A = \int_a^b f(x)\,dx \ . \tag{1}$$

By the fundamental theorem of calculus, we know that continuous functions have an exact analytical solution. Using $F(x)$ to denote the antiderivative of $f(x)$, this is given by

$$\int_a^b f(x)\,dx = F(b) - F(a) \ . \tag{2}$$

In practice, it is not uncommon for $F(x)$ to be very difficult (if not impossible) to obtain, and we are frequently forced to settle for a numerical approximation instead. The simplest way to obtain such an approximation is through the use of a *Riemann Sum*, which may be defined in several different ways. For example, the simplest Riemann Sum is the *right-point* rule, as depicted in Figure 2(a), in which the area is approximated by dividing the region into n little rectangles and then adding up their combined areas. The width Δx of each rectangle is then simply given by

$$\Delta x = \frac{b-a}{n} \ . \tag{3}$$

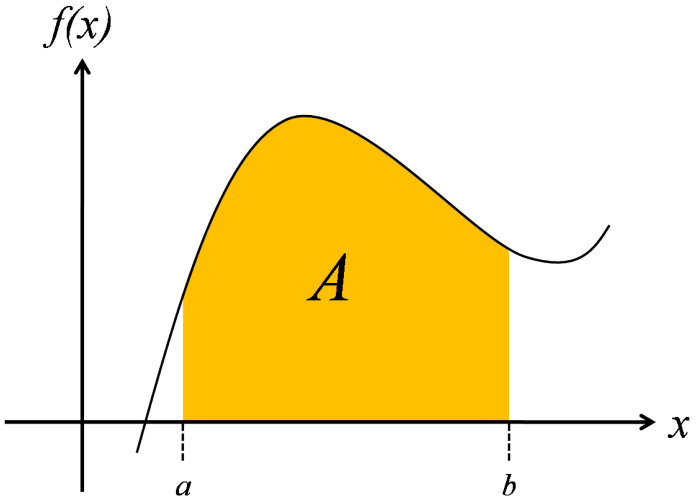


Figure 1. The integral of $f(x)$ from a to b, represented as the area under the curve.

The height of each rectangle is found by evaluating the function at the points x_i along the domain between a and b, where x_i is defined as

$$x_i = a + i\Delta x \ . \tag{4}$$

The total area A is therefore approximated by

$$A \approx \sum_{i=1}^{n} f(x_i)\,\Delta x \ . \tag{5}$$

It is important to note that part of the definition of a definite integral is the condition that a Riemann sum must approach the true value of A as $n \to \infty$:

$$\lim_{n\to\infty} \sum_{i=1}^{n} f(x_i)\,\Delta x = \int_a^b f(x)\,dx = A \ . \tag{6}$$

The importance of this condition is that it assures us any arbitrary degree of accuracy by simply adding up the appropriate number of rectangles. Note, however, that we could have just as easily satisfied this requirement by using the *left-point* rule, as depicted in Figure 2(b). In this case, the x_i points are simply evaluated as

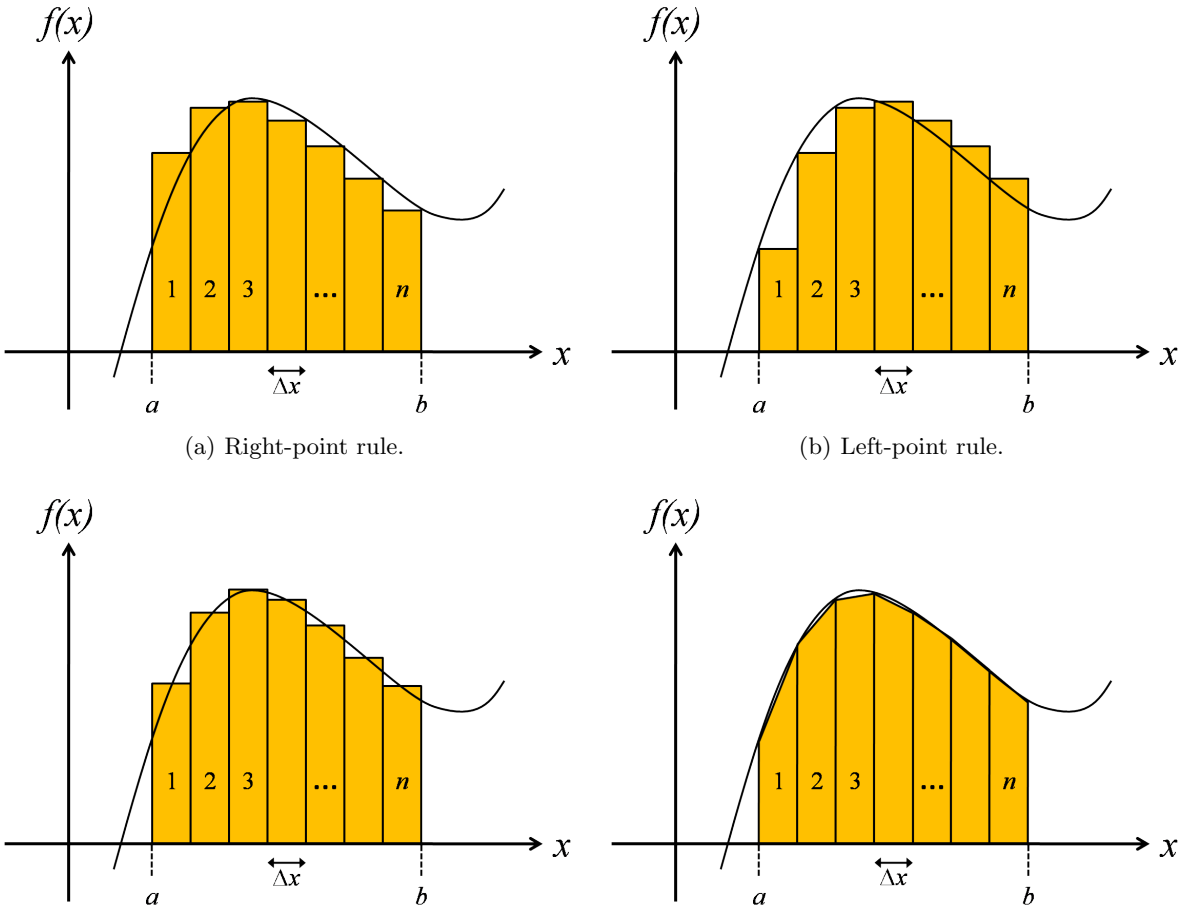
$$x_i = a + (i-1)\Delta x \ , \tag{7}$$

and A is again approximated by adding up the appropriate number of rectangles.

3. ERROR BOUNDS

When considering a potential rule for numerical integration, it is helpful to establish some sort of metric to represent the effectiveness of the rule. This is usually accomplished by taking the difference between the integration rule and the true, analytical result. Letting A' denote the numerical approximation to the true area A, the numerical error of integration is given by

$$\epsilon = A' - A \ . \tag{8}$$



(a) Right-point rule.

(b) Left-point rule.

(c) Midpoint rule.

(d) Trapezoidal rule.

Figure 2. Various methods for calculating a numerical integral.

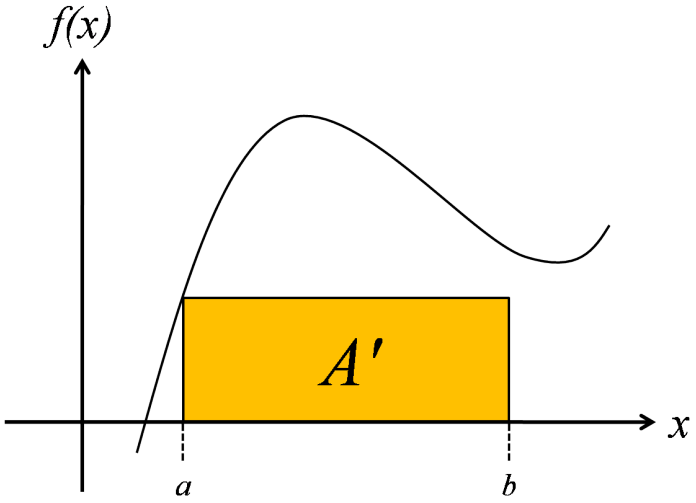


Figure 3. Left-point approximation using $n = 1$ rectangle.

Naturally, it is not possible to evaluate ϵ without knowledge of A itself (and if we knew A, there would be no need to perform the numerical integral). We can, however, establish a sort of upper limit that determines the effectiveness of the numerical integration technique. To see how, remember that the exact area A is given from Equation (2) as

$$A = F(b) - F(a) \ . \tag{9}$$

If we now take the number $F(b)$ and express it as a Taylor series expansion around a, we find

$$F(b) = F(a) + F'(a)(b-a) + \frac{1}{2!}F''(a)(b-a)^2 + \frac{1}{3!}F^{(3)}(a)(b-a)^3 + \cdots \ . \tag{10}$$

Next, we note that the function $F'(a)$ is simply $f(a)$, and that $\Delta x = (b-a)$. Subtracting $F(a)$ therefore gives

$$A = F(b) - F(a) = f(a)\Delta x + \frac{1}{2!}f'(a)\Delta x^2 + \frac{1}{3!}f^{(2)}(a)\Delta x^3 + \cdots \ . \tag{11}$$

So as this expression shows, the area A under a curve can be represented as its own Taylor series expansion.

Now consider the left-point rule evaluated with $n = 1$ rectangles, as shown in Figure 3. Clearly, this scenario represents the absolute worst-case accuracy for approximating the area of the curve under f. The error for $n = 1$ therefore represents the *error bound* of the left-point rule, because the error can only improve as more rectangles are added. Thus, if we take $\Delta x = (b-a)$ and $x_i = a$, the approximate area is found to be

$$A' = f(a)\Delta x \ . \tag{12}$$

Comparison with Equation (11) shows that this is simply the first term in the Taylor series expansion for A. For this reason, the left-point rule is referred to as a *zeroth order* approximation to the area, because it represents the zeroth-order derivative term.

If we now compute the error term, the result is

$$\epsilon = -\left(\frac{1}{2!}f'(a)\Delta x^2 + \frac{1}{3!}f^{(2)}(a)\Delta x^3 + \cdots\right) \ . \tag{13}$$

For very small values of Δx, the f' term will tend to dominate the error. It is therefore common to drop all but the lowest order error term and express the error bound as simply

$$\epsilon \approx -\frac{1}{2} f'(a)\Delta x^2 \ . \tag{14}$$

The importance of Equation (14) is that it tells us the relative severity of error we can expect by approximating the area of a curve with a single rectangle over a very small value of Δx. In particular, it shows how smaller values of Δx will have less error, and how functions with a small derivative are better approximated. For this reason, constant-valued functions such as $f(x) = c$ are perfectly evaluated by the left-point rule because their derivatives are identically zero.

To calculate the error-bound of the right-point rule, we begin with the Taylor series expansion of $F(a)$ around the point b:

$$F(a) = F(b) - F'(b)(a-b) + \frac{1}{2!}F''(b)(a-b)^2 - \frac{1}{3!}F^{(3)}(b)(a-b)^3 + \cdots \ . \tag{15}$$

Solving for A in the same manner as before then leads to

$$A = f(b)\Delta x - \frac{1}{2!}f'(b)\Delta x^2 + \frac{1}{3!}f^{(2)}(b)\Delta x^3 - \cdots \ . \tag{16}$$

Again, we immediately recognize the first term in the series as the right-point rule, thus proving it is likewise a zeroth-order approximation to the area. It is also important to note the alternating signs in the series expansion, which will be important later. The total error is then found to be

$$\epsilon = \frac{1}{2!}f'(a)\Delta x^2 - \frac{1}{3!}f^{(2)}(a)\Delta x^3 + \cdots \ , \tag{17}$$

with alternating signs out to infinity. For small values of Δx, this approximates into

$$\epsilon \approx \frac{1}{2}f'(b)\Delta x^2 \ , \tag{18}$$

which is the same the left-point rule, but with a positive sign in front. The significance of the sign is that functions with positive slope will give positive error (ie, overestimated) when using the right-point rule, and negative error (underestimated) when using the left-point rule.

4. MIDPOINT RULE

In practice, a far more useful integration technique is the *midpoint rule*, as shown in Figure 2(c). The rule works the same as before, but simply replaces the x_i terms with

$$x_i = a + (2i-1)\frac{\Delta x}{2} \ . \tag{19}$$

The benefit to this method becomes apparent after computing the error bound, which is readily found by considering the $n = 1$ case in Figure 4. The approximate area A' is first divided into two regions around the point $c = \frac{b+a}{2}$ such that

$$A' = A_1 + A_2 \ . \tag{20}$$

The area defined by A_1 represents a right-point rule at the point c, while the area A_2 represents a left-point rule defined at the same point. Both of these give the same value of area, which is simply

$$A_1 = A_2 = f(c)\frac{\Delta x}{2} \ . \tag{21}$$

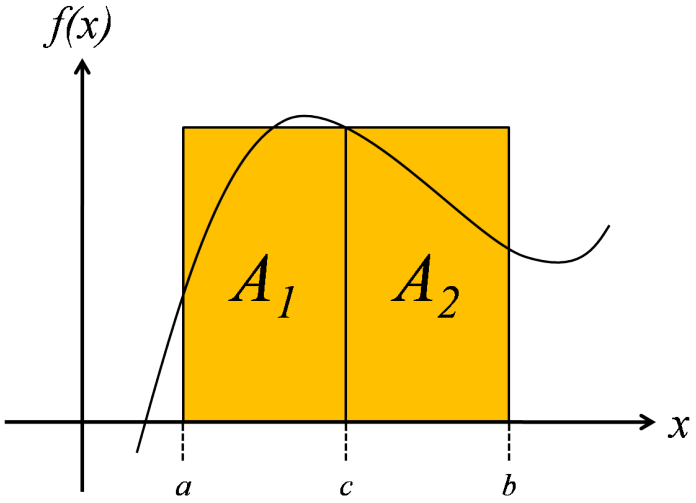


Figure 4. Midpoint approximation using $n = 1$ rectangle. The point c is defined by $\frac{b+a}{2}$.

The total error in A' is then found by simply adding the individual errors from A_1 and A_2. Adding Equations (13) and (17) with a width of $\Delta x/2$ therefore gives

$$\epsilon = -2\left(\frac{1}{3!}f^{(2)}(c)\frac{\Delta x^3}{8} + \frac{1}{5!}f^{(4)}(c)\frac{\Delta x^5}{32} + ...\right) \ . \tag{22}$$

For small values of Δx, this simplifies into

$$\epsilon \approx -\frac{1}{24}f^{(2)}(c)\Delta x^3 \ . \tag{23}$$

As this expression shows, the midpoint rule is a *first order* approximation for A because the error bound is dependent on the second derivative of f. This means linear functions such as $f(x) = mx + b$ are perfectly evaluated by the midpoint rule. As a result, the midpoint rule provides an extra order of accuracy by simply redefining the position of each x_i. For this reason, the midpoint rule is nearly always preferable to either the left- and right-point rules.

5. TRAPEZOIDAL RULE

Another useful rule is the *trapezoidal rule*, which is depicted in Figure 2(d). Under this rule, A' is evaluated by dividing the area into little trapezoids and then adding up their corresponding areas. From this definition, it is straightforward to show that

$$A' = \Delta x\left[\frac{f(a)}{2} + \frac{f(b)}{2} + \sum_{i=1}^{n-1} f(x_i)\right] \ , \tag{24}$$

where $x_i = a + i\Delta x$. The approximate error bound for the trapezoidal rule is derived in a simlar manner as the others, but we shall skip ahead to the end result:

$$\epsilon \approx -\frac{1}{12}f^{(2)}(c)\Delta x^3 \ . \tag{25}$$

Interestingly, the trapezoidal rule is a first-order approximation just like the midpoint rule. In fact, based on the scale factor in front, the midpoint rule is even slightly more accurate. This might make one wonder why we should even bother with the added complexity of the trapezoidal rule when the midpoint rule is just as

good. The answer to this lies beyond the mere error bounds of the approximation. For example, one common routine for numerically evaluating an integral is to iteratively double the precision until a specified error bound is reached. The trapezoidal rule is ideal for this because half the points are already evaluated, and the computer need only perform a new set of function evaluations in between the original x_i's. Such an algorithm would not be as efficient with the midpoint rule because it would have to reevaluate the function at every new x_i along the domain. So depending on the application, the trapezoidal rule can still hold merit over the midpoint rule.

6. HIGHER ORDER RULES

The field of numerical integration is very rich, and great efforts have been made to achieve the highest possible accuracy with each numerical computation. For example, the simplest second-order approximation is called *Simpson's Rule* and works by connecting the x_i points with second-order polynomials. However, there is no need to stop here, and integration rules have been developed to order four, five, and beyond. The trade-off with such rules is usually with the complexity of the algorithm, and eventually one must decide where it is worthwhile to simply perform more fevals rather than try to squeeze the most out of each computation.

Singularities present an interesting problem when using Riemann sums, and a special form of numerical integration called *Gaussian quadrature* must be employed in order to reasonably approximate an area. Discontinuities are also a major concern, and must be carefully considered as well. However, such topics are best left for a course in numerical methods.

7. 2D INTEGRATION

Integration in two dimensions works very much the same as in one dimension, only now it represents the volume V between some function $f(x, y)$ and a bounded region R in the x-y plane:

$$V = \iint_R f(x, y) dy dx \ . \tag{26}$$

For introductory purposes, it helps to limit R to the simple rectangular area depicted in Figure 5. We shall therefore define R as the region bounded by $a \leq x \leq b$ and $c \leq y \leq d$, leaving the generalized case for an advanced course. The volume is then expressed in the double integral

$$V = \int_a^b \int_c^d f(x, y) \, dy dx \ . \tag{27}$$

Just as there exists a wide variety of techniques for evaluating integrals in one dimension, there is an equally wide variety for two dimensions and beyond. We shall therefore limit the discussion to an introduction of the two simplest methods, which are the 2D midpoint rule and the 2D trapezoidal rule.

7.1 2D Midpoint Rule

The 2D Midpoint rule begins by dividing the region R into a set of little rectangular subdomains, each with length Δx and width Δy. As illustrated in Figure 6, the x-dimension is divided into n subdomains while the y-dimension is divided into m, with the center point of each rectangle given by (x_i, y_j). The volume of the box at this subdomain is then simply given by $f(x_i, y_j)\Delta x \Delta y$, and the total volume V is approximated by adding up all the little box elements:

$$V' = \sum_{i=1}^{n} \sum_{j=1}^{m} f(x_i, y_j) \Delta x \Delta y \ , \tag{28}$$

where

$$x_i = a + \frac{i\Delta x}{2} \quad \text{and} \quad y_j = c + \frac{j\Delta y}{2} \ . \tag{29}$$

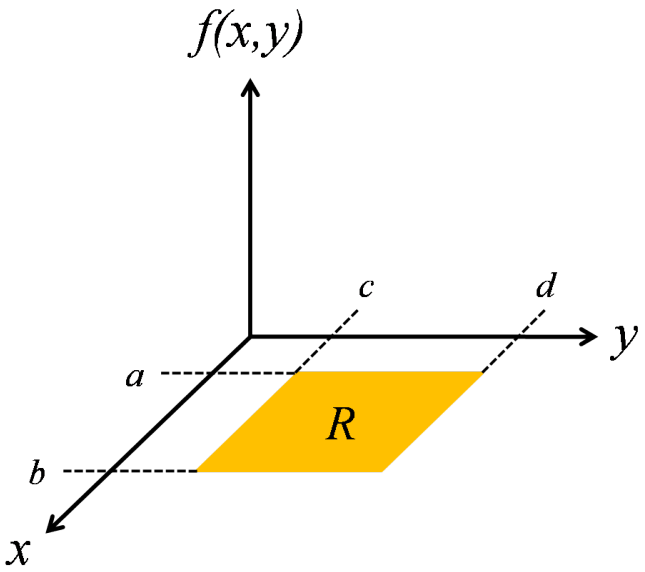


Figure 5. 2D region R defined by $a \leq x \leq b$ and $c \leq y \leq d$.

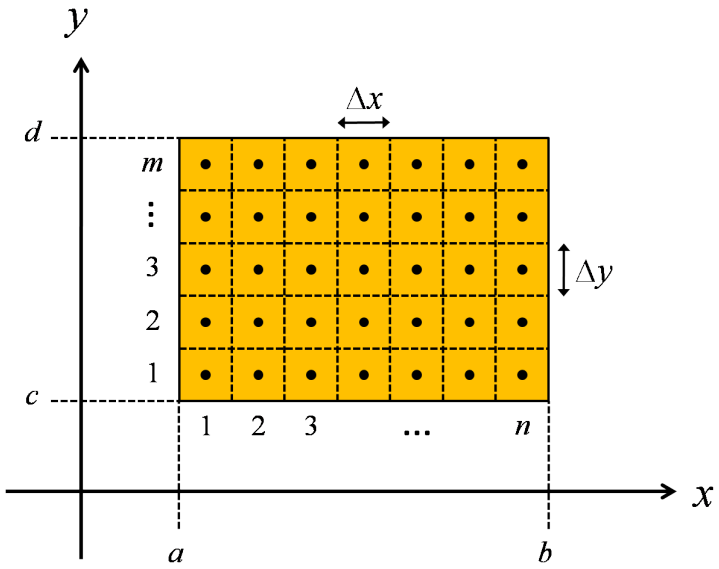


Figure 6. Subdomains for the 2D midpoint rule and their corresponding feval points.

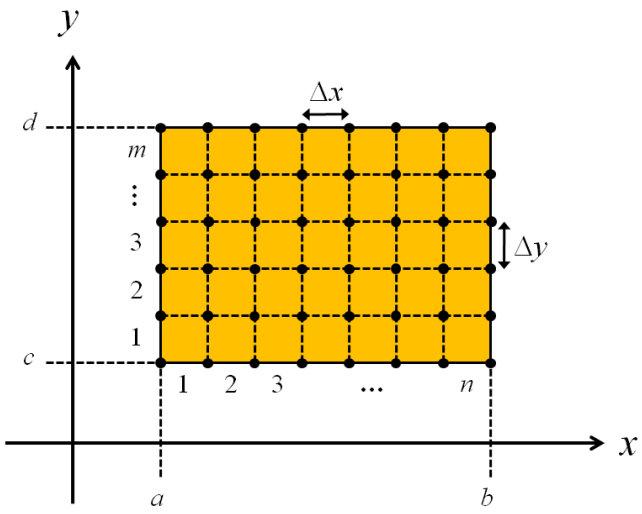


Figure 7. Subdomains for the 2D trapezoidal rule and their corresponding feval points.

7.2 2D Trapezoidal Rule

The division of R by the 2D trapezoidal rule is shown in Figure 7, where the feval points are given by

$$x_i = a + i\Delta x \quad \text{and} \quad y_j = c + j\Delta y \ , \tag{30}$$

with $i = 0, 1, \ \ldots \ , n$ and $j = 0, 1, \ \ldots \ , m$. In other words, the vertices represent the four corner of each subdomain. The volume of each 3D trapezoidal element is found by simply taking the average of the height at all four vertices and treating this as the equivalent "height" of a box. Each sub-volume may then be written as

$$V_{ij} = \frac{1}{4}\Delta x \Delta y \left[f(x_{i-1}, y_{j-1}) + f(x_i, y_{j-1}) + f(x_{i-1}, y_j) + f(x_i, y_j) \right] \ . \tag{31}$$

Extending this single volume out to the entire region R therefore gives

$$\begin{aligned} V' = {} & \frac{\Delta x \Delta y}{4} \left[f(a,c) + f(a,d) + f(b,c) + f(b,d) \right] \\ & + \frac{1}{2}\Delta x \Delta y \left[\sum_{i=1}^{n-1} f(x_i, c) + \sum_{i=1}^{n-1} f(x_i, d) + \sum_{j=1}^{m-1} f(a, y_j) + \sum_{j=1}^{m-1} f(b, y_j) \right] \\ & + \Delta x \Delta y \sum_{i=1}^{n-1} \sum_{j=1}^{m-1} f(x_i, y_j) \ . \end{aligned} \tag{32}$$

Although this algorithm may appear complicated, it is really nothing more than an expression of how many averages each vertex contributes to. Namely, the four corners only contribute to one subdomain each, while the outer edges contribute to two subdomains, and the inner points all contribute to four subdomains. This expression also demonstrates the difference in complexity between various integration methods. From a purely computational perspective, both the 2D midpoint rule and trapezoidal rule require the same number of fevals to complete. Yet from an implementation perspective, the 2D trapezoidal rule requires much more thought and attention to physically program and debug the code. But just as the 1D case, scaling is much easier to do with the trapezoidal rule than it is with the midpoint rule. Which method one chooses to employ will therefore depend on the nature of the application.