Acta Scientiarum

http://periodicos.uem.br/ojs
ISSN on-line: 1807-8664
Doi: 10.4025/actascitechnol.v45i1.63310

Numerical Integration of locally Peaked Bivariate Functions

Abdelhamid Taieb Zaidi

Department of Mathematics, College of Science, Qassim University, 51452, Buraydah, Saudi Arabia. E-mail: a.zaidi@qu.edu.sa

ABSTRACT. The aim of this paper is to compare the relative accuracies between deterministic and stochastic methods for solving bounded integrals numerically to observe which methods tend to function well and converge to a small amount of error based on computational resources. For the deterministic method, the Gauss-Legendre quadrature method has been selected and for the stochastic method, the Monte Carlo integration has been selected. For each case, the number of variables will be adjusted to observe the effect on error. For the Gauss-Legendre quadrature method the permutations increased with the inaccuracy of 9% when the number of nodes increased to 3 but was reduced by 90% and later on the error depicted a drop as the number of nodes raised further. For the stochastic method, that was chosen from large sample size, the inaccuracy was found to be inversely proportional to the sample size. This concluded that the monte-carlo approach was not affected by the impact of dimensionality moreover, deterministic method also seemed to overcome the dimensionality constraint.

Keywords: Numerical methods; integration; Monte Carlo; Gauss-Legendre; stochastic; deterministic.

Received on april 20, 2022
Accepted on july 19, 2023

Introduction

According to Davis and Rabinowitz (2014), numerical integration is the process of the numerical approximation of the answer of a definite integral over a set of defined limits (Davis & Rabinowitz, 2007). Typically, most integrals can be solved using analytical methods such as integration by substitution, integration by parts and others. However, Li,Li, and Li states (2019), that sometimes the integrals tend to become difficult or impossible to solve using typical analytical methods. Therefore, to find the integrals of such functions, we must rely on the numerical methods which are algorithms that aid in finding the numerical value of the definite integral of a function. The objective of our paper will be the numerical calculation of definite integrals in two variables (Li et al., 2019).

Of the algorithms used, there is a term known as numerical quadrature which is another term for numerical integration. Specifically, this term is used to represent one dimensional integrals, which are integrals in one variable. Other terminologies are also used which define higher dimension integrals as cubature. Thus, numerical quadrature can be defined as the estimated answer to the following integral

$$\int_a^b f(x)dx$$

This is an integral of a function from a defined range, known as a definite integral, which often has varying degrees of accuracy. For this reason, there are two conditions that must be satisfied. Firstly, the function should be smooth over a small, limited amount of dimensions. Secondly, the function should be bounded. Provided that these two conditions are satisfied, we can come up with many different numerical methods to find estimates of the solution of the integral to a certain level of precision (Paulos, Rychkov, van Rees, & Zan, 2016). There are also multiple advantages of numerical integration over analytical techniques. According to the VLSI Journal publication on integration, the integrand may only be required at certain points, for example, through sampling, which is why many embedded applications tend to use numerical integration instead of analytical integration (https://www.sciencedirect.com/journal/integration). Sometimes it's also possible where we know the technique for the analytical integration, but cannot find the anti-derivative which exists as an elementary function (Zipkin et al., 2021). Finally, sometimes, it's possible to calculate the analytical solution but it's easier to find the solution numerically which can save not only time but also processing power. The basic definition of any numerical

integration algorithm is defined by Lloyd, Irani, and Ahmadi (2020), as the combination of the integrands' evaluations which leads to an approximate solution to the integral. The integration points are finite in number and integrands are calculated at these points. Then, by calculating the weighted sum of these values, we can approximate the integral (Lloyd et al., 2020). These integration points and weights vary from method to method and affect the level of accuracy of the calculation (Thiagarajan & Shapiro, 2016). One of the critical aspects of the analysis of numerical techniques is to assess the trend of the approximate error. The ideal method for this is to specify the error as a function of the number of numerical evaluations (Dolejší, Šebestová, & Vohralík, 2015).

Usually, the quality of the method is judged by assessing which function yields minimal error while keeping number of iterations minimum. These algorithms are then considered to be the better algorithms because they reduce the number of machine cycles by reducing the number of mathematical operations that have to be conducted. This eventually leads to a lower error. This is particularly considerable in cases where the time taken per iteration is high and the integrand to be evaluated is complex. There is a category of numerical integration which is classified as a brute-force numerical integration. This is only applicable to integrands which are piecewise and continuous in nature, and have bounds specified to them. This leads to a possibility of completing the integral using small increments in the values of the variables using the bounds of the function as limits (Milovanović, Igić, & Turnić, 2015).

Most quadrature rules are usually designed to compute integrals in one variable only. There are techniques to extend these methods to multiple variables which involve the consideration of the multiple integral as a series of single one dimensional integrals using Fubini's theorem (Elvira, Martino, & Closas, 2020). This approach, however, causes the function evaluations to grow in an exponential nature as the number of variables increases. Consequently, this is known as the curse of dimensionality. Three techniques are known to overcome this problem. One of these techniques is known as the Monte Carlo integration. This technique uses random distributions to carry out numerical integration. This method computes numerically a definite integral over a given bound and is a subset of a series of methods known as Monte Carlo methods. This is an example of a stochastic method in which random points are chosen between the bound of the integrand at which this integrand is evaluated. Other methods compute the integrand at a regular grid. The Monte Carlo method is quite useful for integrals of higher dimensions (Sauter, Schwab, Sauter, & Schwab, 2011)

There are also multiple methods which can be used to carry out the Monte Carlo integration. Some examples of these methods are particle filter, mean-field particle filter, uniform sampling, etc. Two of the most common methods are finite element method and boundary element methods. These methods are quite popular in engineering applications. Some differential equations, considered to be boundary value problems, can be numerically solved using discretization which results in a series of linear algebraic equation which allows us to approximate a solution to the problem. The accuracy of these methods rely on the quadrature method used for the integration of the functions of the elements. Darbas and Louër (2015), has defined the derivation of integral equations for solutions of different boundary value problems have been defined in. The study funrtheremohasized thatt here are other techniques of numerical integrations in one and multiple dimensions using Gauss quadrature methods which can be more effcient (Darbas & Le Louër, 2015).

For curves that are smooth, and exist over finite intervals often defined as [a, b], the interval can often be linearly converted to the standardized interval [-1,1] in which we can apply the easier method known as the Gauss Legendre quadrature (Bartoň & Calo, 2016). According to Quéau, Durou, and Aujol (2018), the Gaussian quadrature methods are a series of methods that can be classified as deterministic methods. The Gauss Legendre quadrature allows the numerical integration of a function to a certain degree of accuracy; however, the process leads to a slowly converging error for functions based on logarithms (Quéau et al., 2018). In such cases, other weighted quadrature's for Gaussian methods are recommended. The Gauss Legendre quadrature allows the numerical integration of a function to a certain degree of accuracy; however, the process leads to a slowly converging error for functions based on logarithms (Liu, Hager, & Rao, 2017). In such cases, other weighted quadrature for Gaussian methods is recommended (Dao, De Sa, & Ré, 2017).

Methodology

A. equations to be integrated

For the purposes of our paper, we will simulate the numerical integration of two functions by using two separate methods. The first method will be a stochastic method, which will be the Monte Carlo method. The second method will be a deterministic method, or a quadrature method. In each case, the limits for the integration will be taken as [-1,1].

1) First Equation

The first equation to be integrated will be

$$f(x,y) = e^{x^2+y^2} \quad (1)$$

This can be analytically integrated to give us the result:

$$\int_{-1}^{1}\int_{-1}^{1}\left(e^{x^2+y^2}\right)dxdy = 8.5574 \quad (2)$$

Hence, the true value of this integral is 8.5574.

2) Second Equation

The second function to be integrated will be

$$f(x,y) = (cos50x + sin20y)^2 \quad (3)$$

This can be analytically integrated to give us the result:

$$\int_{-1}^{1}\int_{-1}^{1}(cos50x + sin20y)^2 dxdy = 3.9526 \quad (4)$$

Hence, the true value of this integral is 3.9526.

B. Monte Carlo Integration

The first method of integration that we will implement will be the Monte Carlo integration. This is a stochastic form of integration which relies on random numbers. Since the integral of a function is equivalent to the area under the graph of the function plotted, we can use this method to estimate the integral of the function. The method consists of starting with the goal of finding the integral of a function over a given range. For our experiment, our range is going to be limited to [-1,1]. Therefore, we have to find the area of the function between -1 and 1. To do this, we can take a random value of x between -1 and 1 and multiply the value of the function with this given value of x to find the area of the rectangle. This is an approximation of the integral of a function. The Monte Carlo method approximates the integral value by taking the average of the area of the rectangles which is computed for a random value of x between the given range, in this case, -1 and 1. As we keep adding the area of the rectangles and then find the average value of the sum, we can approximate our answer with more and more accuracy. One of the consequences of this is that, the over-estimated areas are compensated for by the under-estimated areas. This leads us to the derivation for the Monte Carlo method:

$$I \approx Q_N \equiv V\frac{1}{N}\sum_{I=1}^{N} f\left(\overline{X}_i\right) = V(f) \quad (5)$$

Where

$$V = \int_{\Omega} d\overline{X} \quad (6)$$

And Ω is a subset of R^m where all the values of x,

$$\overline{X}_i, \ldots, \overline{X}_N \in \Omega \quad (7)$$

And N is the number of uniform samples. These points are sampled and this technique relies on the law of large numbers that states:

$$\lim_{N\to\infty} Q_N = 1 \quad (8)$$

Using the equation 5, we can estimate a value for the integral which will be done for both the given functions. For our experiment, we varied the values of N to observe the effect of increasing the sample size on the accuracy of the algorithm to confirm that the error is inversely related to the sample size. Furthermore, we solve equation 6 to find the value of V which comes out to be 4.

A. Gauss-Legendre quadrature integration

In Numerical Analysis, a classification of algorithm for the approximation of the area under the curve of a function is the quadrature rule. This technique consists of taking the weight sum of the function values at predefined points between the given range of the function. There is a sub-classification of these rules known as the Gaussian quadrature rules wherein the rule yields an exact result for polynomials of degrees up to 2n-1 with weights w_i for i=1, n which is then known as the n-point Gaussian Legendre Quadrature rule. This

method requires the domain of the function to be [-1,1] which allows this rule to be defined as:

$$\int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i) \quad (9)$$

And this method has been known to be exact for polynomials of degree 2n-1 or less. This rule is specifically known as the Gauss-Legendre quadrature rule. There are other limitations to the use of this algorithm. For example, this rule is most accurate when the domain is [-1,1] and the integrand can be written as:

$$f(x) = (1-x)^a(1+x)^b g(x) \quad (10)$$

where a and b are both greater than -1 and g(x) is a low degree polynomial.

Furthermore, the Gauss-Legendre quadrature requires the weights of the function to be known which can be calculated using the equation:

$$w_i = \frac{2}{(1-x_i^2)[P_n'(x_i)]^2} \quad (11)$$

For the interval [-1,1], the quadrature rules can be calculated a varying number of points. For the purposes of our experiment, we varied the number of points from 2 to 5; this is due to the fact that selecting one point leads to an undefined value of the integral.[6]. In the case of the limits being something other than [-1,1], such as [a, b], we can conduct a coordinate transform to change the coordinates into [-1,1] using the following equation:

$$x = \frac{a+b}{2} + \frac{b-a}{2} x_d \quad (12)$$

$$\int_a^b f(x)dx = \int_{-1}^{1} f(\frac{a+b}{2} + \frac{b-a}{2} x_d)(\frac{b-a}{2})dx_d \quad (13)$$

$$= \int_{-1}^{1} g(x_d)dx_d \quad (14)$$

Consequently, for the second variable, the co-ordinate transform becomes:

$$y = \frac{a+b}{2} + \frac{b-a}{2} y_d \quad (15)$$

$$\int_a^b f(y)dy = \int_{-1}^{1} f(\frac{a+b}{2} + \frac{b-a}{2} y_d)(\frac{b-a}{2})dy_d \quad (16)$$

$$= \int_{-1}^{1} g(y_d)dy_d \quad (17)$$

Furthermore, to conduct a Gauss-Legendre for a bivariate function, we have to extend the one-dimensional formula to a two-dimensional formula.

Conversion of the one-dimensional formula into a two-dimensional formula is relatively simple. We have to simple perform repeated integrations as per the number of variables to achieve the two dimensional formula.

Thus, from equation 9, we can derive:

$$\int_{-1}^{1}\int_{-1}^{1} f(x,y)dxdy = \int dy \left(\int f(x,y)dx\right) \quad (18)$$

$$= \int dy G(y) = \sum_{j=1}^{n} w_j \left(\sum_{i=1}^{n} w_i f(x_i, y_j)\right) \quad (19)$$

Results and discussion

For the purposes of our experiment, we have compared the relative accuracies of one stochastic method and one deterministic method. Due to the different nature of the two types of numerical integration methods, it's difficult to compare their performances with each other. However, we can study the effects of adjusting different parameters within the methods themselves to observe their accuracy.

A. Monte Carlo Integration

For stochastic methods such as the Monte Carlo integration method, the variable that can be adjusted is the number of samples. Consequently, we adjusted the number of samples from 10^1 to 10^5 and observed the effect of the number of samples to the error present.

From the implementation of the Monte Carlo integration for function 1, we observe a relatively large error of 29% for a small sample size of N = 10, as shown in Table 1. However, as we increase it to a sample size of N = 100, the error is significantly reduced to nearly 0%). Increasing the sample size further leads to smaller amounts of reduction in errors, as well as a small fluctuation in the error. Consequently, we

may want to stop the sample size at 10^2 if our target error margin is below 1% or at N = 10^5 if smaller errors of less than 0.02% are required. Another trend in the error that we can observe is the oscillating nature of the error from positive to negative, as illustrated in Figure 1. This can be further studied by observing continuous increments of the number of sample size; however, such a test is resource intensive and the results would be difficult to show.

Table 1. Results of Monte Carlo Integration for Function 1

ki	i	true i	error
10	11.062	8.5574	29.266
100	8.6505	8.5574	0.036074
1000	9.517	8.5574	-0.57201
10000	8.4965	8.5574	-0.71165
1e+05	8.5597	8.5574	0.028665

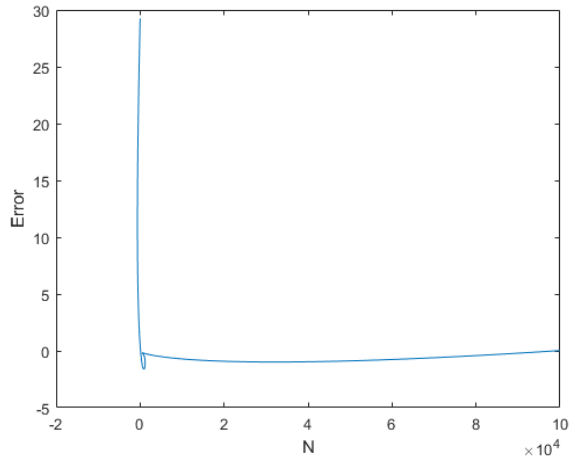

Figure 1. A plot of the error versus the number of samples for Monte Carlo – Function 1.

From the figure above, we can observe that the trend of the error approaches zero as the number of samples increases. The plot would be smoother if the number of samples were reduced to observe the trend of errors for a lower number of samples; the sharp turn in the graph is a result of the large number of samples compared to the small error size.

Table 2 displays the results of the implementation of the Monte Carlo integration for function 2. As observed, with a small sample size of N = 10, the error is quite large; 48%, as in the previous case. The increase to N = 100 reduces this error to approximately 9% and N = 100 leads to a small error of 1%. Similar to the previous observation, the error reduces to below 1% at a sample size of N = 10^4 and small amounts of acceptable error of 1% can be obtained at N = 1000.

Table 2. Results of Monte Carlo integration for function 2

ki	i	true i	error
10	2.0314	3.9526	-48.606
100	3.5847	3.9536	-9.3093
1000	3.997	3.9536	1.1229
10000	3.9396	3.9526	-0.32984
1e+05	3.9632	3.9536	0.26659

The plot in Figure 2 shows a similar trend to the previous graph where as the number of samples increases, the error reduces to nearly zero. There is once again a sharp turn on the graph due to the large number of samples present compared to the small error.

B. Gauss-Legendre Quadrature

For deterministic methods such as the Gauss Quadrature methods, there are no longer a large number of random samples that are randomly selected. For the purposes of weighted Gauss Quadrature methods, we can compare the different values of weights or the number of selected points between the bounds which lead to

the different values of weights. For our case, we selected the Gauss-Legendre Quadrature method which allowed us to compare the error with respect to the number of points. Before proceeding with the Gauss-Legendre quadrature, it's important to use Equation 11 to calculate the weights of the n-point quadrature formula. To display the individual values, we specifically calculated the weights for different values of n. The results of this simulation are shown below in Tables 3,4,5.6, 7 and 8.

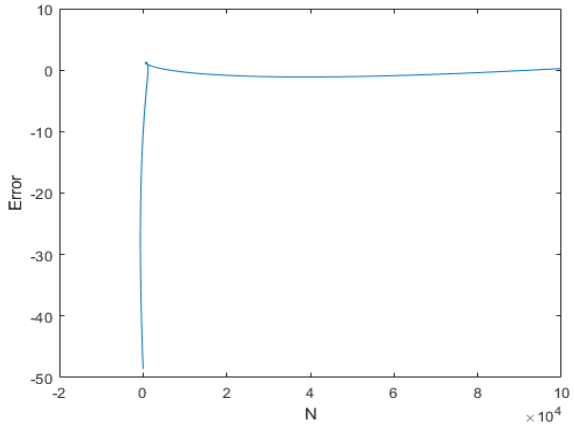

Figure 2. A plot of the error versus the number of samples for Monte Carlo – Function 2.

Table 3. Results of the simulation for k = 1.

Nodes	Weights
0	2

Table 4. Results of the simulation for k = 2.

Nodes	Weights
-0.57735	1
0.57735	1

Table 5. Results of the simulation for k = 3.

Nodes	Weights
-0.7746	0.55556
0	0.88889
0.7746	0.55556

Table 6. Results of the simulation for k = 4.

Nodes	Weights
-0.86114	0.34785
-0.33998	0.65215
033998	0.65215
0.86114	0.34785

Table 7. Results of the simulation for k = 5.

Nodes	Weights
-0.90618	0.23693
-0.53847	0.47863
0	0.56889
0.53847	0.47863
0.90618	0.23693

The results of the first execution of the simulation of the Gauss-Legendre integration lead to the conclusion that for function 1, the iterations proceed to start with a large error of 9% at k=2 which is then reduced by 90% when the number of nodes is increased to 3. The error shows a sharp decrease as the number of nodes is further increased.

By fitting a polynomial to the trend of the error, we can observe the rapid descent towards the y-axis, showing in Figure 3 that as the number of nodes k is increased, the error tends to approach zero. For more diverging functions, we can increase the number of nodes to trade processing power for higher accuracy.

Table 8. Results of gauss quadrature integration for function 1.

k	i	true i	error
-----	------	-----	-----
2	7.7909	8.5574	8.9567
3	8.4883	8.5574	0.80772
4	8.4883	8.5574	0.052122
5	8.5572	8.5574	0.00265

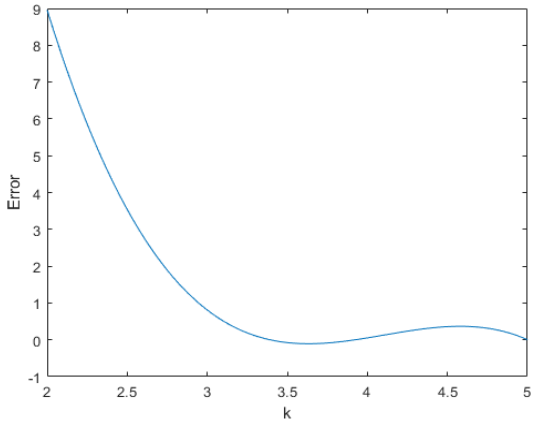

Figure 3. A plot of the error versus the number of points for Gauss Quadrature – Function 1.

For the final part of our simulation, we can observe the trend of the error to be decreasing with a slight oscillation in the Gauss-Legendre integration. As we increase the number of points 'k' the error can visibly be seen to be decreasing, presneted in Table 9. Consequently, we can expect the error to decrease further as the number of points increase and the number of weights increase. Once again, with more resources we can increase the number of points and observe the effect further to confirm this theory.

Table 9. Results of gauss quadrature integration for function 2.

ki	i	true i	error
---------	----------	----------	----------
2	5.6522	3.562	-42.999
3	2.4674	3.9526	37.564
4	2.7252	3.9526	31.052
5	3.5201	3.9526	10.9941

Figure 4. displays the plot of the trend of the error versus the number of points 'k' which shows an initial large error that seems to oscillate a little and is expected to flatten out. We do expect the error to go further down towards zero provided that we increase the number of points 'k' which will eventually lead to an increase in the number of computational resources required.

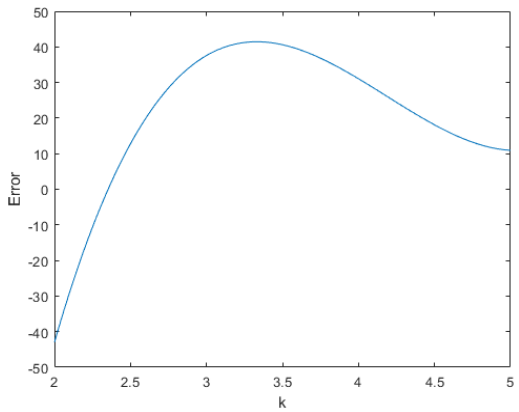

Figure 4. A plot of the error versus the number of points for Gauss Quadrature – Function 2.

Conclusion

To conclude our experimentation, we can observe that for stochastic methods that rely on a random number chosen from a large sample size, the error can be seen to be inversely proportional to the number of samples.

$$error \propto \frac{1}{N}$$

And to be specific, this observation is quite similar to the true relationship between the error and the number of samples which, to be specific, is actually:

$$error \propto \frac{1}{\sqrt{N}}$$

This seems to be the true observation for both of our experiments, for function 1 and function 2. Consequently, we can further experiment with different functions to observe the results for a logarithmic function to see how the method can keep up with the rapidly increasing area under the graph. This method also is not resource intensive on its own which reinforces the idea that the Monte Carlo method is immune to the curse of dimensionality. Secondly, for the deterministic methods such as weighted quadrature methods, we also observe that these methods also tend to overcome the curse of dimensionality by using weights that depend on the number of points chosen between two bounded limits given. This leads to the increasing accuracy of the system without leading too much into complex calculations. Consequently, we can observe the fact that the percentage error is inversely related to the number of points selected 'k'.

$$error \propto \frac{1}{k}$$

Acknowledgments

The researchers would like to thank the Deanship of Scientific Research, Qassim University for funding the publication of this project.

References

Bartoň, M., & Calo, V. M. (2016). Optimal quadrature rules for odd-degree spline spaces and their application to tensor-product-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, *305*, 217-240.

Dao, T., De Sa, C. M., & Ré, C. (2017). Gaussian quadrature for kernel features. *Advances in neural information processing systems*, *30*.

Darbas, M., & Le Louër, F. (2015). Well-conditioned boundary integral formulations for high-frequency elastic scattering problems in three dimensions. *Mathematical Methods in the Applied Sciences*, *38*(9), 1705-1733.

Davis, P. J., & Rabinowitz, P. (2007). *Methods of numerical integration*. Courier Corporation.

Dolejší, V., Šebestová, I., & Vohralík, M. (2015). Algebraic and discretization error estimation by equilibrated fluxes for discontinuous Galerkin methods on nonmatching grids. *Journal of Scientific Computing*, *64*(1), 1-34.

Elvira, V., Martino, L., & Closas, P. (2020). Importance gaussian quadrature. *IEEE Transactions on Signal Processing*, *69*, 474-488.

Li, H., Li, Y., & Li, S. (2019). Dual neural network method for solving multiple definite integrals. *Neural computation*, *31*(1), 208-232.

Liu, F., Hager, W. W., & Rao, A. V. (2017). Adaptive mesh refinement method for optimal control using decay rates of Legendre polynomial coefficients. *IEEE Transactions on Control Systems Technology*, *26*(4), 1475-1483.

Lloyd, S., Irani, R. A., & Ahmadi, M. (2020). Using neural networks for fast numerical integration and optimization. *IEEE Access*, *8*, 84519-84531.

Milovanović, G. V., Igić, T. S., & Turnić, D. (2015). Generalized quadrature rules of Gaussian type for numerical evaluation of singular integrals. *Journal of Computational and Applied Mathematics*, *278*, 306-325.

Paulos, M. F., Rychkov, S., van Rees, B. C., & Zan, B. (2016). Conformal invariance in the long-range Ising model. *Nuclear Physics B*, *902*, 246-291.

Quéau, Y., Durou, J.-D., & Aujol, J.-F. (2018). Normal integration: a survey. *Journal of Mathematical Imaging and Vision*, *60*, 576-593.

Sauter, S. A., Schwab, C., Sauter, S. A., & Schwab, C. (2011). *Boundary element methods*. Springer.

Thiagarajan, V., & Shapiro, V. (2016). Adaptively weighted numerical integration in the finite cell method. *Computer Methods in Applied Mechanics and Engineering*, *311*, 250-279.

Zipkin, E. F., Zylstra, E. R., Wright, A. D., Saunders, S. P., Finley, A. O., Dietze, M. C., Itter, M. S., & Tingley, M. W. (2021). Addressing data integration challenges to link ecological processes across scales. *Frontiers in Ecology and the Environment*, *19*(1), 30-38.