Archives of Computational Methods in Engineering (2021) 28:553–576
https://doi.org/10.1007/s11831-019-09391-3

ORIGINAL PAPER

New Formulas of Numerical Quadrature Using Spline Interpolation

Pedro Americo Almeida Magalhaes[1] · Pedro Americo Almeida Magalhaes Junior[1] · Cristina Almeida Magalhaes[2] · Ana Laura Mendonca Almeida Magalhaes[1]

Received: 16 April 2019 / Accepted: 23 December 2019 / Published online: 4 January 2020

Abstract
This work develops formulas for numerical integration with spline interpolation. The new formulas are shown to be alternatives to the Newton–Cotes integration formulas. These methods have important application in integration of tables or for discrete functions with constant steps. An error analysis of the technique was conducted. A new type of spline interpolation is proposed in which a polynomial passes through more than two tabulated points. The results show that the proposed formulas for numerical integration methods have high precision and absolute stability. The obtained methods can be used for the integration of stiff equations. This paper opens a new field of research on numerical integration formulas using splines.

1 Introduction

Texts on numerical methods abound with formulas for numerical integration sometimes called quadrature or mechanical quadrature [1, 2]. The function f(x), which is to be integrated, may be a known function or a set of discrete data. This is not surprising, since there are so many possibilities for selecting the base-point spacing, the degree of the interpolating polynomial, and the location of base points with respect to the interval of integration. Many known functions, however, do not have an exact integral, and an approximate numerical procedure is required to compute the integral. In many cases, the function f(x) is known only as a set of discrete points, in which case an approximate numerical procedure is required to compute the integral [1, 2]. Numerical integration formulas can be developed by fitting approximating functions to discrete data and integrating the approximating function:

$$I = \int_{x_1}^{x_n} f(x)dx \cong \int_{x_1}^{x_1+(n-1)h} P(x)dx \tag{1}$$

when the function to be integrated has known values at equally spaced points ($\Delta x = h$=constant) and n is number of points with x ranging as: x_1, $x_2 = x_1 + h$, $x_3 = x_1 + 2h$, …, $x_{n-1} = x1 + (n-2)h$, $x_n = x_1 + (n-1)h$, a polynomial P(x) can be fit to the discrete data [2, 3]. The resulting formulas are called Newton–Cotes formulas that employ functional values at equally spaced base points.

The distance between the lower and upper limits of an integral is called the range of integration. The distance between any two data points is called an increment or step ($\Delta x = h$) [1, 3–5].

2 State of the Art on Numerical Quadrature

There is a large literature on numerical integration, also called quadrature. Of special importance are the midpoint rule and Simpson's rule. They are simple to use and bring enormous improvements for smooth functions in low dimensions [6–8]. The advantage of classical quadrature methods decays rapidly with increasing dimension. This phenomenon is a manifestation of Bellman's 'curse of dimensionality', with Monte Carlo versions in two classic theorems of Bakhvalov.

The trapezoid rule is based on a piecewise linear approximation. The trapezoid rule integrates correctly any function f that is piecewise linear on each segment $[x_{i-1}, x_i]$, by using two evaluation points at the ends of the segments [9–11]. The midpoint rule also integrates such a function correctly using just one point in the middle of each segment. The midpoint rule has benefitted from an error cancellation. This

✉ Pedro Americo Almeida Magalhaes Junior
pamerico@pucminas.br

[1] Pontifícia Universidade Católica de Minas Gerais, Av: Dom José Gaspar, 500 – Prédio 10 - Coração Eucarístico, Belo Horizonte, MG CEP 30535-901, Brazil

[2] Centro Federal de Educação Tecnológica de Minas Gerais - CEFET-MG, Av. Amazonas, 5.253, Nova Suiça, Belo Horizonte, MG CEP 30421-169, Brazil

kind of cancellation plays a big role in the development of classical quadrature methods.

The midpoint rule has a big practical advantage over the trapezoid rule. It does not evaluate f at either endpoint a or b. Many of the integrals that we apply Monte Carlo methods to diverge to infinity at one or both endpoints. In such cases, the midpoint rule avoids the singularity. There are numerous mathematical techniques for removing singularities [12–14]. When we have no such analysis of our integrand, perhaps because it has a complicated problem-dependent formulation, or because we have hundreds of integrands to consider simultaneously, then avoiding the singularity is attractive. By contrast, the trapezoid rule does not avoid the endpoints $x = a$ and $x = b$. For such methods a second, less attractive principle is to ignore the singularity, perhaps by using $f(x_i) = 0$ at any sample point x_i where f is singular.

The midpoint and trapezoid rules are based on correctly integrating piecewise constant and linear approximations to the integrand. That idea extends naturally to methods that locally integrate higher order polynomials [15–17]. The result is much more accurate integration, at least when the integrand is smooth. The idea behind Simpson's rule generalizes easily to higher orders. We split the interval [a, b] into panels, find a rule that integrates a polynomial correctly within a panel, and then apply it within every panel to get a compound rule.

There are two main varieties of compound quadrature rule. For open rules we do not evaluate f at the end-points of the panel. The midpoint rule is open. For closed rules we do evaluate f at the end-points of the panel [18, 19]. The trapezoid rule and Simpson's rule are both closed. Closed rules have the advantage that some function evaluations get reused when we increase n. Open rules have a perhaps greater advantage that they avoid the ends of the interval where singularities often appear.

The trapezoid rule and Simpson's rule use $n = 2$ and $n = 3$ points respectively within each panel. In general, one can use m points to integrate polynomials of degree $n - 1$, to yield the Newton–Cotes formulas, of which the trapezoid rule and Simpson's rule are special cases [12, 20, 21]. The Newton–Cotes rule for $n = 4$ is another of Simpson's rules, called Simpson's 3/8 rule. Newton–Cotes rules of odd order have the advantage that, by symmetry, they also correctly integrate polynomials of degree m, as we saw already in the case of Simpson's rule.

High order rules should be used with caution [22–24]. They exploit high order smoothness in the integrand, but can give poor outcomes when the integrand is not as smooth as they require. In particular if a genuinely smooth quantity has some mild nonsmoothness in its numerical implementation f, then high order integration rules can behave very badly, magnifying this numerical noise.

As a further caution, note that taking f fixed and letting the order n in a Newton–Cotes formula increase does not always converge to the right answer even for f with infinitely many derivatives. Lower order rules applied in panels are more robust [23–25]. The Newton–Cotes rules can be made into compound rules similarly to the way Simpson's rule was compounded. When the basic method integrates polynomials of degree r exactly within panels, then the compound method has error $O(m^{-r})$, assuming that $f^{(r)}$ is continuous on $[a, b]$.

The rules considered above evaluate f at equispaced points. The basic panel for a Gauss rule is conventionally $[-1, 1]$ or sometimes $\mathbb{R}$, and not $[0, h]$ as we used for Newton–Cotes rules. Also the target integration problem is generally weighted. The widely used weight functions are multiples of standard probability density functions, such as the uniform, gamma, Gaussian and beta distributions [12, 24, 26]. The idea is that having f be nearly a polynomial can be much more appropriate than requiring the whole integrand $f(x)w(x)$ to be nearly a polynomial. Choosing w_i and x_i together yields $2n$ parameters and it is then possible to integrate polynomials of degree up to $2n - 1$ without error.

Unlike Newton–Cotes rules, Gauss rules of high order have non-negative weights. We could in principle use a very large n. For the uniform weighting $w(x) = 1$ though, we could also break the region into panels. Then for m function evaluations the error will be $O(m^{-2n})$ assuming as usual that $f^{(2n)}$ is continuous on [a, b]. Gauss rules for uniform weights on $[-1, 1]$ have the advantage that they can be used within panels.

Quadrature rules offer an elegant and efficient way to numerically evaluate integrals of functions from a linear space under consideration [24–27]. These rules typically require function evaluation at specific points, called nodes, and these values are multiplied by constants, called weights, to give the final value of the integral as a weighted sum.

There is an extensive number of various quadrature rules depending on n (f is univariate, bivariate, multivariate), domain shape (disc, hypercube, simplex), and the type of the linear space (polynomials, splines, rational functions, smooth non-polynomial) [12, 13, 28, 29]. For polynomial multivariate integration, the field is well studied. In the univariate case, a lot of research has been devoted to piecewise polynomials to address integration for spline spaces arising in isogeometric analysis. Introduced so called half-point rule for splines that needs the minimum number of quadrature points. However, this rule is in general exact only over the whole real line (infinite domain).

For finite domains, one may introduce additional quadrature points which make the rule non-Gaussian (slightly suboptimal in terms of the number of quadrature points), but more importantly, it yields quadrature weights that can be negative, unlike in Gaussian quadratures. When computing Galerkin approximations, assembling mass and stiffness

matrices is the bottleneck of the whole computation and efficient quadrature rules for specific spline spaces are needed to efficiently evaluate the matrix entries [12, 13, 26, 29–31].

In the multivariate case where spline spaces possess a tensor-product structure, univariate quadrature rules are typically used in each direction, resulting in tensor-product rules. Recently, it have changed the paradigm of the mass and stiffness matrix computation from the traditional element-wise assembly to a row-wise concept [32–34]. When building the mass matrix, one B-spline basis function of the scalar product involved is considered as a positive measure (i.e., a weight function), and a weighted quadrature with respect to that weight is computed for each matrix row. Such an approach brings significant computational savings because the number of quadrature points in each parameter dimension is independent on the polynomial degree and requires asymptotically (for a large number of elements) only two points per element. For the multivariate case that is unstructured such as triangular meshes in 2D, however, constructing efficient quadrature rules from tensor product counterparts is unnatural, resulting in rules that are often not symmetric even though they act on a symmetric domain [35–37].

Classical quadrature methods are very well tuned to one-dimensional problems with smooth integrands. A natural way to extend them to multi-dimensional problems is to write them as iterated one-dimensional integrals, via Fubini's theorem [12, 13, 38, 39]. When we estimate each of those one-dimensional integrals by a quadrature rule, we end up with a set of sample points on a multi-dimensional grid. Unfortunately, there is a curse of dimensionality that severely limits the accuracy of this approach. This curse of dimensionality is not confined to sampling on grids formed as products of one-dimensional rules. Any quadrature rule in high dimensions will suffer from the same problem. Two important theorems of Bakhvalov, make the point.

Bakhvalov's theorem makes high-dimensional quadrature seem intractable. There is no way to beat the rate $O(n^{-r/d})$, no matter where you put your sampling points x_i or how cleverly you weight them. At first, this result looks surprising, because we have been using Monte Carlo methods which get an root mean square error $O(n^{-1/2})$ in any dimension. The explanation is that in Monte Carlo sampling we pick one single function $f(\cdot)$ with finite variance $\sigma 2$ and then in sampling n uniform random points, get an root mean square error of $\sigma n^{-1/2}$ for the estimate of that function's integral. Bahkvalov's theorem works in the opposite order [40]. We pick our points $x_1,..., x_n$, and their weights w_i. Then Bakhvalov finds a function f with r derivatives on which our rule makes a large error. When we take a Monte Carlo sample, there is always some smooth function for which we would have got a very bad answer. Such worst case analysis is very pessimistic because the worst case functions could behave very oddly right near our sampled $x_1,.., x_n$, and the worst case functions might look nothing like the ones we are trying to integrate. We can hybridize quadrature and Monte Carlo methods by using each of them on some of the variables. Hybrids of Monte Carlo and quasi-Monte Carlo methods are often used [5–7, 41–43].

The Laplace approximation is a classical device for approximate integration. The Laplace approximation is very accurate when $log(f(x))$ is smooth and the quadratic approximation is good where f is not negligible. Such a phenomenon often happens when x is a statistical parameter subject to the central limit theorem, $f(x)$ is its posterior distribution, and the sample size is large enough for the Central Limit Theorem to apply [20, 21, 44–46].

The Laplace approximation is now overshadowed by Markov Chain Monte Carlo. One reason is that the Laplace approximation is designed for unimodal functions. When prior distribution has two or more important modes, then the space can perhaps be cut into pieces containing one mode each, and Laplace approximations applied separately and combined, but such a process can be cumbersome. Markov Chain Monte Carlo by contrast is designed to find and sample from multiple modes, although on some problems it will have difficulty doing so. The Laplace approximation also requires finding the optimum of a d-dimensional function and working with the Hessian at the mode. In some settings that optimization may be difficult, and when d is extremely large, then finding the determinant of the Hessian can be a challenge. Finally, posterior distributions that are discrete or are mixtures of continuous and discrete parts can be handled by Markov Chain Monte Carlo but are not suitable for the Laplace approximation. The Laplace approximation is not completely superceded by Markov Chain Monte Carlo. In particular, the fully exponential version is very accurate for problems with modest dimension d and large n. When the optimization problem is tractable then it may provide a much more automatic and fast answer than Markov Chain Monte Carlo does [12, 17, 23, 31, 34, 47].

There is some mild controversy about the use of adaptive methods. There are theoretical results showing that adaptive methods cannot improve significantly over non-adaptive ones. There are also theoretical and empirical results showing that adaptive methods may do much better than non-adaptive ones. These results are not contradictory, because they make different assumptions about the problem. For a high level survey of when adaptation helps [4, 47–49].

Sparse grids were originally developed for the quadrature of high-dimensional functions. The method is always based on a one-dimensional quadrature rule, but performs a more sophisticated combination of univariate results. However, whereas the tensor product rule guarantees that the weights of all of the cubature points will be positive if the weights of

the quadrature points were positive, Smolyak's rule does not guarantee that the weights will all be positive [39–41, 48].

Bayesian Quadrature is a statistical approach to the numerical problem of computing integrals and falls under the field of probabilistic numerics [42–44]. It can provide a full handling of the uncertainty over the solution of the integral expressed as a Gaussian Process posterior variance. It is also known to provide very fast convergence rates which can be up to exponential in the number of quadrature points n.

The problem of evaluating the integral can be reduced to an initial value problem for an ordinary differential equation by applying the fundamental theorem of calculus. For instance, the standard fourth-order Runge–Kutta method applied to the differential equation yields Simpson's rule. Thus, in our view, numerical quadrature problems are often wrongly studied as a numerical solution of differential equations. Numerical integration is a much more general and distinct study of the differential equations [50–53].

3 Newton–Cotes Numerical Integration Formulas

The Newton–Cotes formulas are shown for comparison with the new formulas obtained using splines. The closed integration formulas use information about f(x), that is, they have base points, at both limits of integration. The trapezoid rule for a single interval is obtained by fitting a first-degree polynomial to two discrete points [1, 2, 4, 26]. Simpson's 1/3 rule is obtained by fitting a second-degree polynomial to three equally spaced discrete points. Simpson's 3/8 rule is obtained by fitting a third-degree polynomial to four equally spaced discrete points. Boole's rule is obtained by fitting a fourth-degree polynomial to five equally spaced discrete points. Table 1 shows Newton–Cotes closed integration formulas.

Generally, for closed equations where n is the number of points, *ci* are integer coefficients, *num* is the integer numerator that multiplies step $h = \Delta x$ and *den* is the integer denominator, the closed integration formula is:

$$I = \frac{num}{den} h(ci_1 y_1 + ci_2 y_2 + ci_3 y_3 + ci_4 y_4 + \cdots + ci_{n-2} y_{n-2} + ci_{n-1} y_{n-1} + ci_n y_n) \tag{2}$$

Table 1 Newton–Cotes closed integration formulas

Formulas	Error ≈
Trapezoid Rule $I = \frac{h}{2}(y_1 + y_2)$	$O(h^2)$
Simpson's 1/3 Rule $I = \frac{h}{3}(y_1 + 4y_2 + y_3)$	$O(h^4)$
Simpson's 3/8 Rule $I = \frac{3h}{8}(y_1 + 3y_2 + 3y_3 + y_4)$	$O(h^4)$
Boole's Rule $I = \frac{2h}{45}(7y_1 + 32y_2 + 12y_3 + 32y_4 + 7y_5)$	$O(h^6)$
$I = \frac{5h}{288}(19y_1 + 75y_2 + 50y_3 + 50y_4 + 75y_5 + 19y_6)$	$O(h^6)$
$I = \frac{h}{140}(41y_1 + 216y_2 + 27y_3 + 272y_4 + 27y_5 + 216y_6 + 41y_7)$	$O(h^8)$
$I = \frac{7h}{17280}\left(\begin{array}{l} 751y_1 + 3577y_2 + 1323y_3 + 2989y_4 + 2989y_5 + 1323y_6 \\ + 3577y_7 + 751y_8 \end{array}\right)$	$O(h^8)$
$I = \frac{4h}{14175}\left(\begin{array}{l} 989\ y_1 + 5888\ y_2 - 928\ y_3 + 10496\ y_4 - 4540\ y_5 \\ + 10496\ y_6 - 928\ y_7 + 5888\ y_8 + 989\ y_9 \end{array}\right)$	$O(h^{10})$
$I = \frac{9h}{89600}\left(\begin{array}{l} 2857\ y_1 + 15741\ y_2 + 1080\ y_3 + 19344\ y_4 + 5778\ y_5 \\ + 5778\ y_6 + 19344\ y_7 + 1080\ y_8 + 15741\ y_9 + 2857\ y_{10} \end{array}\right)$	$O(h^{10})$
$I = \frac{5h}{299376}\left(\begin{array}{l} 16067\ y_1 + 106300\ y_2 - 48525\ y_3 + 272400\ y_4 - 260550\ y5 \\ + 427368\ y_6 - 260550\ y_7 + 272400\ y_8 - 48525\ y_9 + 106300\ y_{10} \\ + 16067\ y_{11} \end{array}\right)$	$O(h^{12})$
$I = \frac{11h}{87091200}\left(\begin{array}{l} 2171465\ y_1 + 13486539\ y_2 - 3237113\ y_3 + 25226685\ y_4 \\ - 9595542\ y_5 + 15493566\ y_6 + 15493566\ y_7 - 9595542\ y_8 \\ + 25226685\ y_9 - 3237113\ y_{10} + 13486539\ y_{11} + 2171465\ y_{12} \end{array}\right)$	$O(h^{12})$
$I = \frac{h}{5255250}\left(\begin{array}{l} 1364651\ y_1 + 9903168\ y_2 - 7587864\ y_3 + 35725120\ y_4 \\ - 51491295\ y_5 + 87516288 y_6 - 87797136\ y_7 + 87516288\ y_8 \\ - 51491295\ y_9 + 35725120\ y_{10} - 7587864\ y_{11} + 9903168\ y_{12} \\ + 1364651 y_{13} \end{array}\right)$	$O(h^{14})$

The step size is given by $h = \Delta x = x_{i+1} - x_i$

In the open integration formulas, the first (y_1) and last (y_n) points do not appear in the formula. The open integration formulas do not require information about f(x) at limits of integration [1, 2, 4, 5, 27]. The midpoint rule for a double interval is obtained by fitting a zero-degree polynomial to three discrete points. The upper limit of integration is $x_3 = x_1 + 2h$. Table 2 shows Newton–Cotes open integration formulas.

Generally, for the open formulas, where n is the number of points, *ci* are the integer coefficients, *num* is the integer numerator that multiplies step $h = \Delta x$ and *den* is the integer denominator, the open integration formula is:

$$I = \frac{num}{den} h(ci_2 y_2 + ci_3 y_3 + ci_4 y_4 + \cdots + ci_{n-2} y_{n-2} + ci_{n-1} y_{n-1}) \tag{3}$$

In the semi-open integration formulas, the last point (y_n) does not appear in the formula. In the semi-closed integration formulas, the first point (y_1) does not appear in formula [1, 2, 28–30]. The formula for a double interval is obtained by fitting a zero-degree polynomial to three discrete points. When N is odd, the semi-closed or semi-open integration formulas are the same as the open integration formulas. The upper limit of integration is $x_3 = x_1 + 2h$, and the integral (I) has the following formula:

$$I = 2h(y_2) \quad or \quad I = 2hy_2 \quad Error \approx O(h^2) \tag{4}$$

For three intervals, the semi-open formula is obtained by fitting a first-degree polynomial to four discrete points. The upper limit of the integral is $x_4 = x_1 + 3h$; then, the integral (I) has the following formula:

$$I = \frac{3h}{4}(y_1 + 3y_3) \quad or \quad I = 0.75hy_1 + 2.25hy_3 \quad Error \approx O(h^3) \tag{5}$$

Table 3 shows Newton–Cotes semi-open integration formulas [1–3]. Generally, for semi-open formulas, where n is the number of points, the semi-open integration formula is:

Table 2 Newton–Cotes open integration formulas

Formula	Error ≈
Midpoint Rule $I = 2hy_2$	$O(h^2)$
$I = \frac{3h}{2}(y_2 + y_3)$	$O(h^2)$
$I = \frac{4h}{3}(2y_2 - y_3 + 2y_4)$	$O(h^4)$
$I = \frac{5h}{24}(11y_2 + y_3 + y_4 + 11y_5)$	$O(h^4)$
$I = \frac{3h}{10}(11y_2 - 14y_3 + 26y_4 - 14y_5 + 11y_6)$	$O(h^6)$
$I = \frac{7h}{1440}(611y_2 - 453y_3 + 562y_4 + 562y_5 - 453y_6 + 611y_7)$	$O(h^6)$
$I = \frac{8h}{945}(460y_2 - 954y_3 + 2196y_4 - 2459y_5 + 2196y_6 - 954y_7 + 460y_8)$	$O(h^8)$
$I = \frac{9h}{4480}(1787\,y_2 - 2803\,y_3 + 4967\,y_4 - 1711\,y_5 - 1711\,y_6 + 4967\,y_7 - 2803\,y_8 + 1787\,y_9)$	$O(h^8)$
$I = \frac{5h}{4536}(4045\,y_2 - 11690\,y_3 + 33340\,y_4 - 55070\,y_5 + 67822\,y_6 - 55070\,y_7 + 33340\,y_8 - 11690\,y_9 + 4045y_{10})$	$O(h^{10})$

The step size is given by $h = \Delta x = x_{i+1} - x_i$

Table 3 Newton–Cotes semi-open integration formulas

Formulas	Error ≈
$I = 2hy_2$	$O(h^2)$
$I = \frac{3h}{4}(y_1 + 3y_3)$	$O(h^3)$
$I = \frac{4h}{3}(2y_2 - y_3 + 2y_4)$	$O(h^4)$
$I = \frac{5h}{144}(19y_1 - 10y_2 + 120y_3 - 70y_4 + 85y_5)$	$O(h^5)$
$I = \frac{3h}{10}(11y_2 - 14y_3 + 26y_4 - 14y_5 + 11y_6)$	$O(h^6)$
$I = \frac{7h}{8640}(751y_1 - 840y_2 + 8547y_3 - 11648y_4 + 14637y_5 - 7224y_6 + 4417y_7)$	$O(h^7)$
$I = \frac{8h}{945}(460y_2 - 954y_3 + 2196y_4 - 2459y_5 + 2196y_6 - 954y_7 + 460y_8)$	$O(h^8)$
$I = \frac{9h}{44800}(2857\,y_1 - 4986\,y_2 + 51966\,y_3 - 110322\,y_4 + 182880\,y_5 - 177102\,y_6 + 129666\,y_7 - 50886\,y_8 + 20727\,y_9)$	$O(h^9)$
$I = \frac{5h}{4536}(4045\,y_2 - 11690\,y_3 + 33340\,y_4 - 55070\,y_5 + 67822\,y_6 - 55070\,y_7 + 33340\,y_8 - 11690\,y_9 + 4045y_{10})$	$O(h^{10})$

The step size is given by $h = \Delta x = x_{i+1} - x_i$. The formulas for n odd are equal to those of Newton–Cotes open

$$I = \frac{num}{den} h(ci_1 y_1 + ci_2 y_2 + ci_3 y_3 + ci_4 y_4 + \cdots + ci_{n-2} y_{n-2} + ci_{n-1} y_{n-1}) \tag{6}$$

The semi-closed integration formulas are the same as the semi-open formulas. For example, in case of n = 3:

$$I = 2h(y_2) \quad or \quad I = 2hy_2 \quad Error \approx O(h^2) \tag{7}$$

For N = 4:

$$I = \frac{3h}{4}(y_2 + 3y_4) \quad or \quad I = 0.75hy_2 + 2.25hy_4 \quad Error \approx O(h^3) \tag{8}$$

Table 4 shows Newton–Cotes semi-closed integration formulas. Generally, for semi-closed formulas, where n is the number of points, *ci* are integer coefficients, the semi-closed integration formula is:

$$I = \frac{num}{den} h(ci_2 y_2 + ci_3 y_3 + ci_4 y_4 + \cdots + ci_{n-2} y_{n-2} + ci_{n-1} y_{n-1} + ci_n y_n) \tag{9}$$

The semi-open or semi-closed formulas can be used on a type of improper integral, that is one with a lower limit of $-\infty$ or an upper limit of $+\infty$. Such integrals can usually be evaluated by making a change in the variable that transforms the infinite limit to one that is finite [1–3, 29]. The following identity serves this purpose and works for any function that decreases toward zero at least as fast as the function $1/x^2$ as x approaches infinity:

$$\int_a^b f(x)dx = \int_{1/b}^{1/a} \frac{1}{w^2} f\left(\frac{1}{w}\right) dw \tag{10}$$

where $ab > 0$. Therefore, it can be used only when a is positive and b is ∞ or when a is $-\infty$ and b is negative. For cases where the limits are from $-\infty$ to ∞, the integral can be implemented in three steps [1–4]. For example:

$$\int_{-\infty}^{\infty} f(x)dx = \int_{-\infty}^{-A} f(x)dx + \int_{-A}^{A} f(x)dx + \int_{A}^{\infty} f(x)dx \tag{11}$$

where A is a positive number. One problem with using Eq. (10) to evaluate an integral is that the transformed function will be singular at one of the limits [1, 3, 30, 31]. The semi-open or semi-closed integration formula can be used to circumvent this dilemma as these formulas allow evaluation of the integral without employing the data at the end points of the integration range.

4 Spline Interpolation

In applying the Newton–Cotes method, (n − 1)th-order polynomials were used to interpolate between n data points. This curve captures of all the meandering suggested by the points. However, there are cases where these functions can lead to erroneous results because of round-off errors and overshooting. An alternative approach is to apply lower-order polynomials to subsets of the data points. These connecting polynomials are called spline functions [1, 2, 4, 31–33].

For example, third-order curves, which are employed to connect each pair of data points, are called cubic splines. These functions can be constructed so that the connections between adjacent cubic equations are visually smooth. On the surface, it would seem that the third-order approximation of the splines would be inferior to the higher-order expressions. You might wonder why a spline would ever be preferable. There are situations in which a spline performs better than a higher-order polynomial. This is the

Table 4 Newton–Cotes semi-closed integration formulas

Formulas	Error ≈
$I = 2hy_2$	$O(h^2)$
$I = \frac{3h}{4}(3y_2 + y_4)$	$O(h^3)$
$I = \frac{4h}{3}(2y_2 - y_3 + 2y_4)$	$O(h^4)$
$I = \frac{5h}{144}(85y_2 - 70y_3 + 120y_4 - 10y_5 + 19y_6)$	$O(h^5)$
$I = \frac{3h}{10}(11y_2 - 14y_3 + 26y_4 - 14y_5 + 11y_6)$	$O(h^6)$
$I = \frac{7h}{8640}(4417y_2 - 7224y_3 + 14637y_4 - 11648y_5 + 8547y_6 - 840y_7 + 751y_8)$	$O(h^7)$
$I = \frac{8h}{945}(460y_2 - 954y_3 + 2196y_4 - 2459y_5 + 2196y_6 - 954y_7 + 460y_8)$	$O(h^8)$
$I = \frac{9h}{44800}\begin{pmatrix} 20727\ y_2 - 50886\ y_3 + 129666\ y_4 - 177102\ y_5 + 182880\ y_6 \\ - 110322\ y_7 + 51966\ y_8 - 4986\ y_9 + 2857\ y_{10} \end{pmatrix}$	$O(h^9)$
$I = \frac{5h}{4536}\begin{pmatrix} 4045\ y_2 - 11690\ y_3 + 33340\ y_4 - 55070\ y_5 + 67822\ y_6 \\ - 55070\ y_7 + 33340\ y_8 - 11690\ y_9 + 4045 y_{10} \end{pmatrix}$	$O(h^{10})$

The formulas for n odd are equal to those of Newton–Cotes open. Note that, rules are reflected in relation to semi-open formulas

case where a function is generally smooth but undergoes an abrupt change somewhere in the region of interest. In this case a higher-order polynomial will tend to erratically oscillate in the vicinity of the abrupt change. In contrast, the spline also connects the points, but because it is limited to lower-order changes, the oscillations are kept to a minimum. Thus the spline usually provides a superior approximation of the behavior of functions that have local, abrupt changes [1, 2, 4, 34–37].

The concept of the spline originated from the drafting technique of using a thin, flexible strip (called a spline) to draw smooth curves through a set of points. The process is depicted in Fig. 1 for a series of five pins (data points). In this technique, the draftsman places paper over a wooden board and hammers nails or pins into the paper (and board) at the location of the data points [38–41]. A smooth cubic curve results from interweaving the strip between the pins. Hence, the name "cubic spline" has been adopted for polynomials of this type [1, 3, 4, 12].

Using this practical historical interpolation device, one could also calculate the area under the curve, by using the weight of the sand beneath the spline, as shown in Fig. 2.

In interpolation by splines, between every two points, we have a polynomial of a certain degree [1, 2, 4, 41–44]. Therefore, the interpolation is not made by a single polynomial but by many polynomials. Below, an example with 5 points and 4 third-degree polynomials forming a cubic spline is given. We can see from the equations that there are 12 unknown coefficients, (a_0,…,a_3,b_0,…,b_3,c_0,…,c_3,d_0…,d_3).

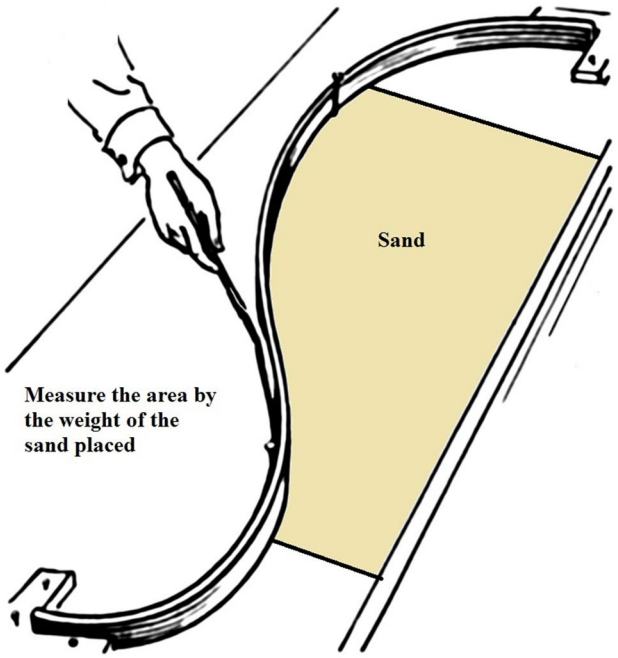

Fig. 2 Primitive integration, using the weight of the sand to calculate the area under the curve generated by a spline interpolation

$$f(x) = \begin{cases} P_a(x) = a_0 + a_1x + a_2x^2 + a_3x^3, \ se \ x_1 \le x \le x_2 \\ P_b(x) = b_0 + b_1x + b_2x^2 + b_3x^3, \ se \ x_2 \le x \le x_3 \\ P_c(x) = c_0 + c_1x + c_2x^2 + c_3x^3, \ se \ x_3 \le x \le x_4 \\ P_d(x) = d_0 + d_1x + d_2x^2 + d_3x^3, \ se \ x_4 \le x \le x_5 \end{cases} \quad (12)$$

As shown in Fig. 3, the objective in using cubic splines is to derive a third-order polynomial for each interval between the knots (between two data points). Thus, for n data points

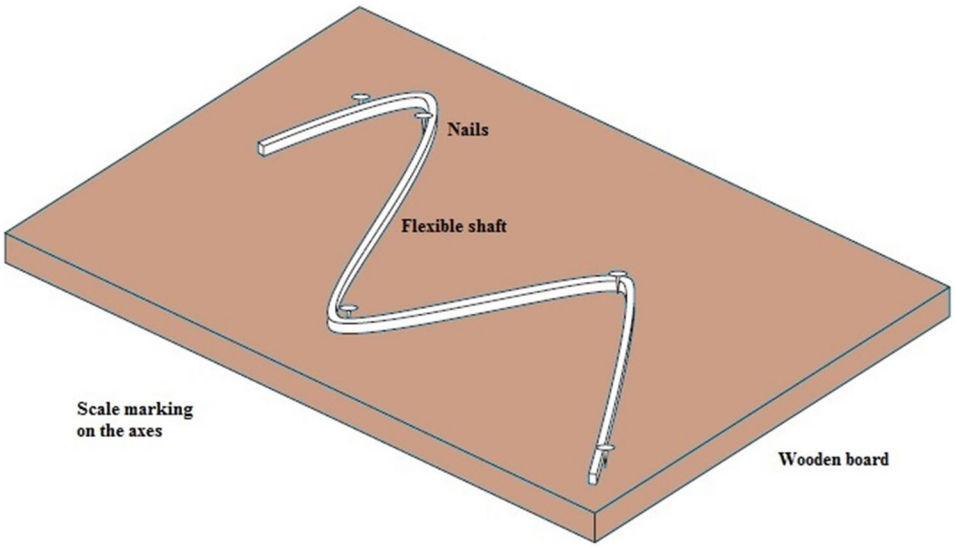

Fig. 1 The practical experimental technique of using a spline to draw smooth curves through a series of points. Notice how, at the end points, the spline straightens out. This is called a "natural" spline

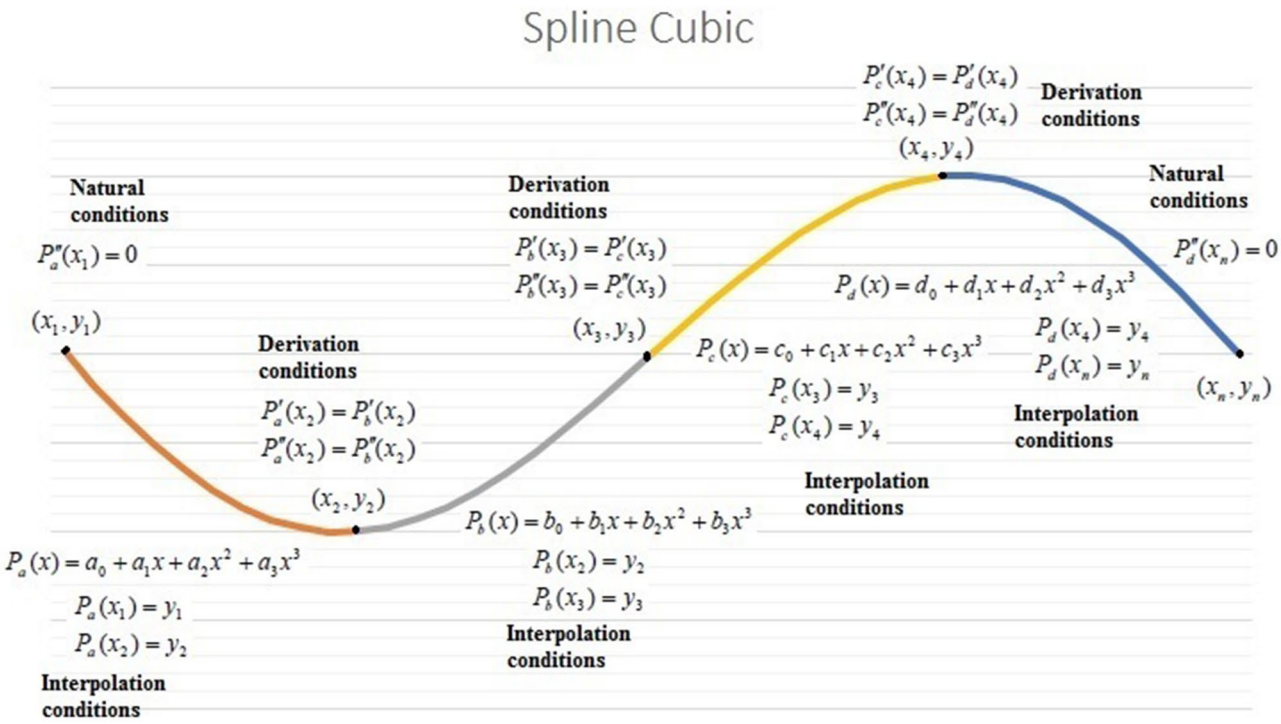

Fig. 3 Cubic spline of 3rd order polynomials for each interval between knots (g) of 5 points (n = 5) and 4 polynomials (P_a, P_b, P_c and P_d)

(i = 1, 2,…, n), there are (n − 2) internal points, without the first and last point. There are (n − 1) intervals and (n − 1) third-order polynomials. Consequently, there are 4(n − 1) unknown constants to evaluate and thus 4n − 4 conditions are required to evaluate the unknown constants. These are as follows:

1. The function values must be equal at the interior knots (2 conditions for each internal point = 2n − 4 conditions).
2. The first and last functions must pass through the end points (2 conditions).
3. The first derivatives at the interior knots must be equal (n − 2 conditions).
4. The second derivatives at the interior knots must be equal (n − 2 conditions).
5. Two derivatives at the first or end knots are zero (2 conditions), chosen from the first to third derivatives of the first and last polynomials: $P'_1(x_1) = 0$, $P''_1(x_1) = 0$, $P'''_1(x_1) = 0$, $P'_{n-1}(x_n) = 0$, $P''_{n-1}(x_n) = 0$ and $P'''_{n-1}(x_n) = 0$.

Therefore, $(2n-4) + 2 + (n-2) + (n-2) + 2 = 4n - 4$, is the number of conditions that is equal to the number of unknown polynomial coefficients.

The visual interpretation of condition 5 is that the function becomes a straight line at the end knots [1, 4, 43–47]. Specification of such an end condition leads to what is termed a "natural" spline. It is given this name because the drafting spline naturally behaves in this fashion. If the value of the second derivative at the end knots is nonzero (that is, there is some curvature), this information can be used alternatively to supply the two final conditions [1, 2, 4, 12].

Generalizing for an order polynomial equal to g, the objective in gth-order splines is to derive a gth-order polynomial for each interval between knots (between two data points), as in

$$P_j(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + \cdots + a_gx^g \qquad (13)$$

Thus, for n data points (i = 1, 2,…, n), there are (n − 2) internal points, without the first and last point. There are (n − 1) intervals and (n − 1) gth-order polynomials consequently, and (g + 1)(n − 1) unknown constants need to be evaluated. Therefore, gn + n − g − 1 conditions are required to evaluate the unknown constants. These are as follows:

1. The function values must be equal at the interior knots (2 conditions for each internal point = 2n − 4 conditions).
2. The first and last functions must pass through the end points (2 conditions).
3. The first to (g − 1) order derivatives at the interior knots must be equal ([g − 1][n − 2] conditions).
4. The (g − 1) derivatives at the first or end knots are zero (g − 1 conditions), chosen from the first to g order derivatives of the first and last polynomials: $P'_1(x_1) = 0$,

$P_1''(x_1)=0$, $P_1'''(x_1)=0$, $P_1^{(4)}(x_1)=0,\ldots$, $P_1^{(g)}(x_1)=0$ and $P'_{n-1}(x_n)=0$, $P''_{n-1}(x_n)=0$, $P'''_{n-1}(x_n)=0$, $P^{(4)}_{n-1}(x_n)=0,\ldots$, $P^{(g)}_{n-1}(x_n)=0$.

Therefore, $(2n-4)+2+(g+1)(n-2)+(g-1)=gn+n-g-1$, is the number of conditions which equals the number of unknown polynomial coefficients.

5 New Type of Spline Interpolation

In this new type of interpolation by splines, instead of each polynomial passing through only two points, the polynomial passes through m points, as shown in Fig. 4. At each m point, the interpolator polynomial is changed, and the derivatives of these two polynomials are equated at these points to give a degree of continuity to the overall curve.

Thus, at the points of change of the polynomial, the value and the successive derivatives of these interpolating polynomials are matched. The number of points (n), the number of polynomials (np), the degree of the polynomial (g), the number of points through which the polynomial passes (m), and the order of the derivatives (d) that will be equalized at the polynomial exchange points are then altered so that we always have the number of equations equal to the number of coefficients of unknown polynomials. To complete the equations, the natural conditions are used, where the successive derivatives of the first and last polynomials in the first and last points, respectively, are made equal to zero. Thus, obtaining different interpolations and different polynomials for the same data points.

The coefficients of the interpolating polynomials are obtained by the resolution of a linear system. The equations of the linear system come from the interpolation conditions, where the polynomials pass through some points $P_j(x_i)=y_i$; from the derivative conditions, where the derivatives of successive orders are equalized $P_j^{(g)}(x_i)=P_{j+1}^{(g)}(x_i)$; and the natural conditions, where the derivatives in the first and last point are equalized to zero, $P_1^{(g)}(x_1)=0$ or $P_{np}^{(g)}(x_n)=0$. The number of equations in the linear system must be equal to the number of polynomials (np) multiplied by the degree of the polynomials plus one $(g+1)$.

Final corrections can still be made before the resolution of a system of linear equations, where the equations that contain one or more of a certain point can be excluded. For example, we can remove equations that contain the first (x_1,y_1) or last (xn, yn) point or the middle point $(x_{n/2},y_{n/2})$. This results in generating integration formulas that do not contain these points.

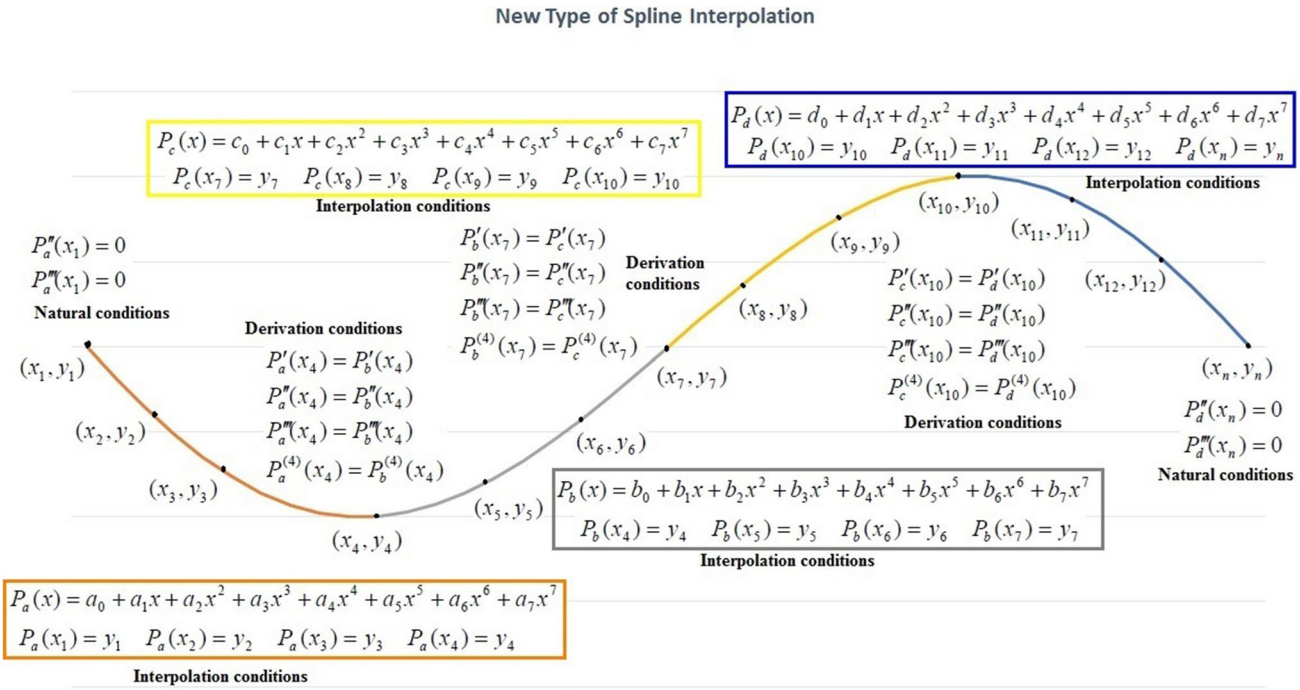

Fig. 4 New type of splines interpolation where order polynomial for each interval between knots (g) is 7, for 13 points (n = 13), where each polynomial passes through 4 points (m = 4) and there are 4 polynomials (P_a, P_b, P_c and P_d)

6 Integration of Polynomials Obtained by Spline Interpolation

Once the interpolating polynomials are obtained by splines, they must be integrated. Each polynomial is integrated onto its x-range [1–3, 12]. Because the spacing of the abscissa x is constant ($\Delta x = h$) and the value start, x_1, has no influence, the integration formulas obtained are all functions of the values of y (y_1, y_2, y_3, y_4,…, y_n). The formula below shows this procedure:

$$I = \int_{x_1}^{x_n} f(x)dx \cong \sum_{j=1}^{np}\left[\int_{x_k}^{x_{k+m}} P_j(x)dx\right] = h\left(cr_1\, y_1 + cr_2\, y_2 + \cdots + cr_{n-1}\, y_{n-1} + cr_n\, y_n\right) \quad (14)$$

Once the integration formula is obtained, it is tested in many examples, and the truncation error is estimated as a function of the order $\Delta x = h$. The stability and convergence of the formula are also tested in several examples where the exact values of the integrals are known. This allows for verification and validation of the new numerical integration formulas.

7 Algorithm for Obtaining Different Integration Formulas by Spline Interpolation

An algorithm is proposed to obtain thousands of integration formulas for different interpolations by splines. A maximum of 25 points (mn = 25) was used, considering a large number of data points, to obtain integration formulas by following the steps below:

1. Set the maximum number of points to 25 (mn = 25)
2. Vary the number of points (n) from 2 to mn (for n = 2 to mn)
3. Vary the number of polynomials (np) from 1 to mn (for np = 1 to mn)
4. Vary the degree of the polynomials (g) from 0 to mn (for g = 1 to mn)
5. Vary the greater order of the derivative that will be matched in the polynomials (d) from 0 to mn (i.e., from d = 0 to mn)
6. Define the data points (x_1,y_1), (x_2,y_2), (x_3,y_3), …, (x_{n-1},y_{n-1}), (x_n,y_n)
7. Set the equally spaced abscissa ($\Delta x = h$) value so that $x_2 = x_1 + h$, $x_3 = x_1 + 2h$,…, $x_n = x_1 + (n-1)h$
8. Define the interpolating polynomials P_1, P_2, P_3, …, P_{np}
9. Calculate the number of polynomial coefficients to be obtained np(g + 1)
10. Obtain the np(g + 1) linear equations
11. Determine the number of data points (m) through which each polynomial will pass (for m = 2 to mn)
12. Obtain the linear equations using the interpolation conditions $P_j(x_i) = y_i$
13. Obtain the linear equations using the derivation conditions $P_j^{(g)}(x_i) = P_{j+1}^{(g)}(x_i)$
14. Obtain the linear equations using the natural conditions $P_1^{(g)}(x_1) = 0$ or $P_{np}^{(g)}(x_n) = 0$
15. Optionally, eliminate the equations that contain a certain point, the first (x_1,y_1) or last point (x_n,y_n) or the middle point ($x_{n/2}$,$y_{n/2}$)
16. Perform the test to continue if the number of equations is equal to the number of unknown polynomial coefficients
17. Solve the linear system to calculate the coefficients of the interpolating polynomials
18. Integrate the obtained polynomials
19. Obtain the formulas for numerical integration
20. Test the integration formulas obtained against known integrals
21. Estimate the truncation error of the formulas
22. Test the convergence, applicability and accuracy of the formulas
23. Select the best verified and validated integration formulas

8 Results Obtained and the Best Integration Formulas

The following tables show some integration formulas obtained. Tables 5, 6, 7, 8, 9 and 10 show integration formulas similar to Newton–Cotes closed formulas in increasing order of truncation error. Tables 11, 12, 13, 14, 15 and 16 show integration formulas similar to Newton–Cotes open formulas in increasing order of truncation error. Tables 17, 18, 19, 20, 21, 22, 23, 24 and 25 show integration formulas similar to Newton–Cotes semi-closed formulas in increasing order of truncation error. Tables 26 and 27 show integration formulas similar to Newton–Cotes semi-closed formulas in

Table 5 Integration formulas similar to Newton–Cotes closed form formulas with truncation error = $O(h^2)$ and degree polynomials g = 1

Points (n)	Formulas
2	$h(y_1 + y_2)/2$
3	$2h(y_1 + y_2 + y_3)/3$
4	$3h(y_1 + y_2 + y_3 + y_4)/4$
5	$4h(y_1 + y_2 + y_3 + y_4 + y_5)/5$
for n	$(n-1)h(y_1 + y_2 + y_3 + y_4 + y_5 + \cdots + y_{n-1} + y_n)/n$

Table 6 Integration formulas similar to Newton–Cotes closed form formulas with truncation error = $O(h^4)$ and degree polynomials g = 2

Points (n)	Formulas
3	$h(y_1 + 4y_2 + y_3)/3$
4	$3h(y_1 + 3y_2 + 3y_3 + y_4)/8$
5	$4h(11y_1 + 26y_2 + 31y_3 + 26y_4 + 11y_5)/105$
6	$5h(31y_1 + 61y_2 + 76y_3 + 76y_4 + 61y_5 + 31y_6)/336$
7	$h(7y_1 + 12y_2 + 15y_3 + 16y_4 + 15y_5 + 12y_6 + 7y_7)/14$
8	$7h(11y_1 + 17y_2 + 21y_3 + 23y_4 + 23y_5 + 21y_6 + 17y_7 + 11y_8)/144$
9	$8h(49y_1 + 70y_2 + 85y_3 + 94y_4 + 97y_5 + 94y_6 + 85y_7 + 70y_8 + 49y_9)/693$
10	$9h(58y_1 + 78y_2 + 93y_3 + 103y_4 + 108y_5 + 108y_6 + 103y_7 + 93y_8 + 78y_9 + 58y_{10})/880$
11	$5h(159y_1 + 204y_2 + 239y_3 + 264y_4 + 279y_5 + 284y_6 + 279y_7 + 264y_8 + 239y_9 + 204y_{10} + 159y_{11})/1287$
12	$11h(127y_1 + 157y_2 + 181y_3 + 199y_4 + 211y_5 + 217y_6 + 217y_7 + 211y_8 + 199y_9 + 181y_{10} + 157y_{11} + 127y_{12})/2184$
13	$12h(55y_1 + 66y_2 + 75y_3 + 82y_4 + 87y_5 + 90y_6 + 91y_7 + 90y_8 + 87y_9 + 82y_{10} + 75y_{11} + 66y_{12} + 55y_{13})/1001$
14	$13h(35y_1 + 41y_2 + 46y_3 + 50y_4 + 53y_5 + 55y_6 + 56y_7 + 56y_8 + 55y_9 + 53y_{10} + 50y_{11} + 46y_{12} + 41y_{13} + 35y_{14})/672$
15	$7h(1313y_1 + 1508y_2 + 1673y_3 + 1808y_4 + 1913y_5 + 1988y_6 + 2033y_7 + 2048y_8 + 2033y_9 + 1988y_{10} + 1913y_{11} + 1808y_{12} + 1673y_{13} + 1508y_{14} + 1313y_{15})/13260$
16	$5h(539y_1 + 609y_2 + 669y_3 + 719y_4 + 759y_5 + 789y_6 + 809y_7 + 819y_8 + 819y_9 + 809y_{10} + 789y_{11} + 759y_{12} + 719y_{13} + 669y_{14} + 609y_{15} + 539y_{16})/3808$
17	$16h(131y_1 + 146y_2 + 159y_3 + 170y_4 + 179y_5 + 186y_6 + 191y_7 + 194y_8 + 195y_9 + 194y_{10} + 191y_{11} + 186y_{12} + 179y_{13} + 170y_{14} + 159y_{15} + 146y_{16} + 131y_{17})/2907$

Table 7 Integration formulas similar to Newton–Cotes closed form formulas with truncation error = $O(h^6)$ and degree polynomials g = 4

Points (n)	Formulas
5	$2h(7y_1 + 32y_2 + 12y_3 + 32y_4 + 7y_5)/45$
6	$5h(19y_1 + 75y_2 + 50y_3 + 50y_4 + 75y_5 + 19y_6)/288$
7	$h(13y_1 + 54y_2 + 27y_3 + 52y_4 + 27y_5 + 54y_6 + 13y_7)/40$
7	$h(268y_1 + 933y_2 + 786y_3 + 646y_4 + 786y_5 + 933y_6 + 268y_7)/770$
8	$7h(1657y_1 + 5157y_2 + 4947y_3 + 4079y_4 + 4079y_5 + 4947y_6 + 5157y_7 + 1657y_8)/31680$
9	$8h(309y_1 + 869y_2 + 904y_3 + 779y_4 + 713y_5 + 779y_6 + 904y_7 + 869y_8 + 309y_9)/6435$
10	$9h(1022y_1 + 2622y_2 + 2877y_3 + 2597y_4 + 2322y_5 + 2322y_6 + 2597y_7 + 2877y_8 + 2622y_9 + 1022y_{10})/22880$
11	$5h(216y_1 + 510y_2 + 580y_3 + 545y_4 + 490y_5 + 466y_6 + 490y_7 + 545y_8 + 580y_9 + 510y_{10} + 216y_{11})/2574$
12	$11h(2973y_1 + 6513y_2 + 7587y_3 + 7367y_4 + 6732y_5 + 6268y_6 + 6268y_7 + 6732y_8 + 7367y_9 + 7587y_{10} + 6513y_{11} + 2973y_{12})/74880$
13	$3h(1837y_1 + 3762y_2 + 4452y_3 + 4438y_4 + 4133y_5 + 3832y_6 + 3712y_7 + 3832y_8 + 4133y_9 + 4438y_{10} + 4452y_{11} + 3762y_{12} + 1837y_{13})/12155$

Table 8 Integration formulas similar to Newton–Cotes closed form formulas with truncation error = $O(h^8)$ and degree polynomials g = 6

Points (n)	Formulas
7	$h(41y_1 + 216y_2 + 27y_3 + 272y_4 + 27y_5 + 216y_6 + 41y_7)/140$
8	$7h(751y_1 + 3577y_2 + 1323y_3 + 2989y_4 + 2989y_5 + 1323y_6 + 3577y_7 + 751y_8)/17280$
9	$8h(79811y_1 + 348212y_2 + 188378y_3 + 241004y_4 + 312215y_5 + 241004y_6 + 188378y_7 + 348212y_8 + 79811y_9)/2027025$
10	$9h(231599y_1 + 934217y_2 + 623690y_3 + 618038y_4 + 795656y_5 + 795656y_6 + 618038y_7 + 623690y_8 + 934217y_9 + 231599y_{10})/6406400$
11	$5h(1464y_1 + 7125y_2 + 2000y_3 + 7500y_4 + 3000y_5 + 6206y_6 + 3000y_7 + 7500y_8 + 2000y_9 + 7125y_{10} + 1464y_{11})/24192$
11	$5h(246633y_1 + 926088y_2 + 708731y_3 + 629904y_4 + 749172y_5 + 830288y_6 + 749172y_7 + 629904y_8 + 708731y_9 + 926088y_{10} + 246633y_{11})/3675672$

Table 9 Integration formulas similar to Newton–Cotes closed form formulas with truncation error = $O(h^{10})$ and degree polynomials g = 8

n	Formulas
9	$4h(989y_1 + 5888y_2 - 928y_3 + 10496y_4 - 4540y_5 + 10496y_6 - 928y_7 + 5888y_8 + 989y_9)/14175$
10	$9h(2857y_1 + 15741y_2 + 1080y_3 + 19344y_4 + 5778y_5 + 5778y_6 + 19344y_7 + 1080y_8 + 15741y_9 + 2857y_{10})/89600$

Table 11 Integration formulas similar to Newton–Cotes open form formulas with truncation error = $O(h^2)$ and degree polynomials g = 1

Points (n)	Formulas
3	$2hy_2$
4	$3h(y_2 + y_3)/2$
5	$4h(y_2 + y_3 + y_4)/3$
6	$5h(y_2 + y_3 + y_4 + y_5)/4$
7	$6h(y_2 + y_3 + y_4 + y_5 + y_6)/5$
for n	$(n-1)h(y_2 + y_3 + y_4 + y_5 + \cdots + y_{n-1})/(n-2)$

Table 10 Integration formulas similar to Newton–Cotes closed form formulas with truncation error = $O(h^{12})$ and degree polynomials g = 10

n	Formulas
11	$5h(16067y_1 + 106300y_2 - 48525y_3 + 272400y_4 - 260550y_5 + 427368y_6 - 260550y_7 + 272400y_8 - 48525y_9 + 106300y_{10} + 16067y_{11})/299376$

increasing order of truncation error; it can be noticed the similarity with the previous tables of semi-closed formulas, having the same coefficients and changing only the index values of the y-ordinates. Tables 28, 29, 30, 31 and 32 show integration formulas similar to Newton–Cotes closed formulas with no middle point in increasing order of truncation error, note that there are no y-values for indexes n/2 or (n + 1)/2 or both. Tables 33, 34, 35, 36 and 37 show integration formulas similar to Newton–Cotes open formulas with no middle point in increasing order of truncation error, note

Table 12 Integration formulas similar to Newton–Cotes open form formulas with truncation error = $O(h^4)$ and degree polynomials g = 2

n	Formulas
5	$4h(2y_2 - y_3 + 2y_4)/3$
6	$5h(11y_2 + y_3 + y_4 + 11y_5)/24$
7	$3h(3y_2 + 2y_4 + 3y_6)/4$
7	$h(4y_2 - 7y_3 + 12y_4 - 7y_5 + 4y_6)$
7	$h(8y_2 - 5y_3 + 12y_4 - 5y_5 + 8y_6)/3$
7	$3h(15y_2 - 12y_3 + 26y_4 - 12y_5 + 15y_6)/16$
7	$3h(24y_2 + 9y_3 + 4y_4 + 9y_5 + 24y_6)/35$
8	$7h(13y_2 + 7y_3 + 4y_4 + 4y_5 + 7y_6 + 13y_7)/48$
9	$8h(14y_2 + 9y_3 + 6y_4 + 5y_5 + 6y_6 + 9y_7 + 14y_8)/63$
10	$9h(21y_2 + 15y_3 + 11y_4 + 9y_5 + 9y_6 + 11y_7 + 15y_8 + 21y_9)/112$
11	$5h(448y_2 + 343y_3 + 268y_4 + 223y_5 + 208y_6 + 223y_7 + 268y_8 + 343y_9 + 448y_{10})/1386$
12	$11h(102y_2 + 82y_3 + 67y_4 + 57y_5 + 52y_6 + 52y_7 + 57y_8 + 67y_9 + 82y_{10} + 102y_{11})/720$
13	$4h(54y_2 + 45y_3 + 38y_4 + 33y_5 + 30y_6 + 29y_7 + 30y_8 + 33y_9 + 38y_{10} + 45y_{11} + 54y_{12})/143$
14	$13h(209y_2 + 179y_3 + 155y_4 + 137y_5 + 125y_6 + 119y_7 + 119y_8 + 125y_9 + 137y_{10} + 155y_{11} + 179y_{12} + 209y_{13})/1848$
15	$7h(88y_2 + 77y_3 + 68y_4 + 61y_5 + 56y_6 + 53y_7 + 52y_8 + 53y_9 + 56y_{10} + 61y_{11} + 68y_{12} + 77y_{13} + 88y_{14})/429$
16	$15h(273y_2 + 243y_3 + 218y_4 + 198y_5 + 183y_6 + 173y_7 + 168y_8 + 168y_9 + 173y_{10} + 183y_{11} + 198y_{12} + 218y_{13} + 243y_{14} + 273y_{15})/2912$
17	$16h(2002y_2 + 1807y_3 + 1642y_4 + 1507y_5 + 1402y_6 + 1327y_7 + 1282y_8 + 1267y_9 + 1282y_{10} + 1327y_{11} + 1402y_{12} + 1507y_{13} + 1642y_{14} + 1807y_{15} + 2002y_{16})/23205$
18	$17h(161y_2 + 147y_3 + 135y_4 + 125y_5 + 117y_6 + 111y_7 + 107y_8 + 105y_9 + 105y_{10} + 107y_{11} + 111y_{12} + 117y_{13} + 125y_{14} + 135y_{15} + 147y_{16} + 161y_{17})/2016$
19	$9h(192y_2 + 177y_3 + 164y_4 + 153y_5 + 144y_6 + 137y_7 + 132y_8 + 129y_9 + 128y_{10} + 129y_{11} + 132y_{12} + 137y_{13} + 144y_{14} + 153y_{15} + 164y_{16} + 177y_{17} + 192y_{18})/1292$

Table 13 Integration formulas similar to Newton–Cotes open form formulas with truncation error = $O(h^6)$ and degree polynomials g = 4

n	Formulas
7	$3h(11y_2 - 14y_3 + 26y_4 - 14y_5 + 11y_6)/10$
8	$7h(611y_2 - 453y_3 + 562y_4 + 562y_5 - 453y_6 + 611y_7)/1440$
9	$8h(1181y_2 - 464y_3 + 467y_4 + 1097y_5 + 467y_6 - 464y_7 + 1181y_8)/3465$
10	$9h(993y_2 - 147y_3 + 203y_4 + 711y_5 + 711y_6 + 203y_7 - 147y_8 + 993y_9)/3520$
11	$5h(1230y_2 + 40y_3 + 185y_4 + 670y_5 + 898y_6 + 670y_7 + 185y_8 + 40y_9 + 1230y_{10})/2574$
12	$11h(3858y_2 + 658y_3 + 603y_4 + 1683y_5 + 2558y_6 + 2558y_7 + 1683y_8 + 603y_9 + 658y_{10} + 3858y_{11})/18720$
13	$2h(208y_2 - 216y_3 + 432y_4 - 329y_5 + 544y_6 - 468y_7 + 544y_8 - 329y_9 + 432y_{10} - 216y_{11} + 208y_{12})/135$
13	$2h(774y_2 + 216y_3 + 146y_4 + 291y_5 + 456y_6 + 524y_7 + 456y_8 + 291y_9 + 146y_{10} + 216y_{11} + 774y_{12})/715$
13	$2h(736y_2 - 1404y_3 + 4320y_4 - 9611y_5 + 18208y_6 - 22068y_7 + 18208y_8 - 9611y_9 + 4320y_{10} - 1404y_{11} + 736y_{12})/405$
14	$13h(10131y_2 + 3711y_3 + 2349y_4 + 3529y_5 + 5364y_6 + 6596y_7 + 6596y_8 + 5364y_9 + 3529y_{10} + 2349y_{11} + 3711y_{12} + 10131y_{13})/63360$
15	$7h(125301y_2 + 54846y_3 + 34896y_4 + 42834y_5 + 61069y_6 + 77036y_7 + 83196y_8 + 77036y_9 + 61069y_{10} + 42834y_{11} + 34896y_{12} + 54846y_{13} + 125301y_{14})/437580$

Table 14 Integration formulas similar to Newton–Cotes open form formulas with truncation error = $O(h^8)$ and degree polynomials g = 6

n	Formulas
9	$8h(460y_2 - 954y_3 + 2196y_4 - 2459y_5 + 2196y_6 - 954y_7 + 460y_8)/945$
10	$9h(1787y_2 - 2803y_3 + 4967y_4 - 1711y_5 - 1711y_6 + 4967y_7 - 2803y_8 + 1787y_9)/4480$
11	$5h(217120y_2 - 259055y_3 + 365080y_4 + 99955y_5 - 197552y_6 + 99955y_7 + 365080y_8 - 259055y_9 + 217120y_{10})/324324$
12	$11h(4501083y_2 - 4068611y_3 + 4707730y_4 + 3934446y_5 - 1212248y_6 - 1212248y_7 + 3934446y_8 + 4707730y_9 - 4068611y_{10} + 4501083y_{11})/15724800$
13	$h(5112y_2 - 8955y_3 + 19120y_4 - 18180y_5 + 16920y_6 - 11234y_7 + 16920y_8 - 18180y_9 + 19120y_{10} - 8955y_{11} + 5112y_{12})/1400$
13	$2h(634470y_2 - 428088y_3 + 423154y_4 + 590601y_5 + 132168y_6 - 152060y_7 + 132168y_8 + 590601y_9 + 423154y_{10} - 428088y_{11} + 634470y_{12})/425425$

Table 15 Integration formulas similar to Newton–Cotes open form formulas with truncation error = $O(h^{10})$ and degree polynomials g = 8

n	Formulas
11	$5h(4045y_2 - 11690y_3 + 33340y_4 - 55070y_5 + 67822y_6 - 55070y_7 + 33340y_8 - 11690y_9 + 4045y_{10})/4536$
12	$11h(2752477y_2 - 6603199y_3 + 15673880y_4 - 17085616y_5 + 8891258y_6 + 8891258y_7 - 17085616y_8 + 15673880y_9 - 6603199y_{10} + 2752477y_{11})/7257600$

Table 16 Integration formulas similar to Newton–Cotes open form formulas with truncation error = $O(h^{12})$ and degree polynomials g = 10

n	Formulas
13	$h(9626y_2 - 35771y_3 + 123058y_4 - 266298y_5 + 427956y_6 - 494042y_7 + 427956y_8 - 266298y_9 + 123058y_{10} - 35771y_{11} + 9626y_{12})/1925$

Table 17 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^1)$ and degree polynomials g = 0

Points (n)	Formulas
4	$h(y_2 + y_3 + y_4)$
5	$h(y_2 + y_3 + y_4 + y_5)$
6	$h(y_2 + y_3 + y_4 + y_5 + y_6)$
for n	$h(y_2 + y_3 + y_4 + y_5 + \cdots + y_{n-1} + y_n)$

also that there are no y-values for indexes n/2 or (n + 1)/2 or both. Tables 38, 39, 40, 41 and 42 show integration formulas similar to Newton–Cotes open formulas without two points in increasing order of truncation error, note that there are no y-values for indexes 1, 2, n − 1 and n.

Table 18 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^2)$ and degree polynomials g = 1

n	Formulas
4	$h(7y_2 + 4y_3 + y_4)/4$
5	$2h(4y_2 + 3y_3 + 2y_4 + y_5)/5$
6	$h(6y_2 + 5y_3 + 4y_4 + 3y_5 + 2y_6)/4$
7	$2h(25y_2 + 22y_3 + 19y_4 + 16y_5 + 13y_6 + 10y_7)/35$
8	$h(11y_2 + 10y_3 + 9y_4 + 8y_5 + 7y_6 + 6y_7 + 5y_8)/8$
9	$2h(14y_2 + 13y_3 + 12y_4 + 11y_5 + 10y_6 + 9y_7 + 8y_8 + 7y_9)/21$
10	$h(52y_2 + 49y_3 + 46y_4 + 43y_5 + 40y_6 + 37y_7 + 34y_8 + 31y_9 + 28y_{10})/40$
11	$2h(21y_2 + 20y_3 + 19y_4 + 18y_5 + 17y_6 + 16y_7 + 15y_8 + 14y_9 + 13y_{10} + 12y_{11})/33$
12	$h(25y_2 + 24y_3 + 23y_4 + 22y_5 + 21y_6 + 20y_7 + 19y_8 + 18y_9 + 17y_{10} + 16y_{11} + 15y_{12})/20$
13	$2h(88y_2 + 85y_3 + 82y_4 + 79y_5 + 76y_6 + 73y_7 + 70y_8 + 67y_9 + 64y_{10} + 61y_{11} + 58y_{12} + 55y_{13})/143$
14	$h(34y_2 + 33y_3 + 32y_4 + 31y_5 + 30y_6 + 29y_7 + 28y_8 + 27y_9 + 26y_{10} + 25y_{11} + 24y_{12} + 23y_{13} + 22y_{14})/28$
15	$2h(39y_2 + 38y_3 + 37y_4 + 36y_5 + 35y_6 + 34y_7 + 33y_8 + 32y_9 + 31y_{10} + 30y_{11} + 29y_{12} + 28y_{13} + 27y_{14} + 26y_{15})/65$
16	$h(133y_2 + 130y_3 + 127y_4 + 124y_5 + 121y_6 + 118y_7 + 115y_8 + 112y_9 + 109y_{10} + 106y_{11} + 103y_{12} + 100y_{13} + 97y_{14} + 94y_{15} + 91y_{16})/112$
17	$2h(50y_2 + 49y_3 + 48y_4 + 47y_5 + 46y_6 + 45y_7 + 44y_8 + 43y_9 + 42y_{10} + 41y_{11} + 40y_{12} + 39y_{13} + 38y_{14} + 37y_{15} + 36y_{16} + 35y_{17})/85$
18	$h(56y_2 + 55y_3 + 54y_4 + 53y_5 + 52y_6 + 51y_7 + 50y_8 + 49y_9 + 48y_{10} + 47y_{11} + 46y_{12} + 45y_{13} + 44y_{14} + 43y_{15} + 42y_{16} + 41y_{17} + 40y_{18})/48$
19	$2h(187y_2 + 184y_3 + 181y_4 + 178y_5 + 175y_6 + 172y_7 + 169y_8 + 166y_9 + 163y_{10} + 160y_{11} + 157y_{12} + 154y_{13} + 151y_{14} + 148y_{15} + 145y_{16} + 142y_{17} + 139y_{18} + 136y_{19})/323$
20	$h(69y_2 + 68y_3 + 67y_4 + 66y_5 + 65y_6 + 64y_7 + 63y_8 + 62y_9 + 61y_{10} + 60y_{11} + 59y_{12} + 58y_{13} + 57y_{14} + 56y_{15} + 55y_{16} + 54y_{17} + 53y_{18} + 52y_{19} + 51y_{20})/60$
21	$2h(76y_2 + 75y_3 + 74y_4 + 73y_5 + 72y_6 + 71y_7 + 70y_8 + 69y_9 + 68y_{10} + 67y_{11} + 66y_{12} + 65y_{13} + 64y_{14} + 63y_{15} + 62y_{16} + 61y_{17} + 60y_{18} + 59y_{19} + 58y_{20} + 57y_{21})/133$

Table 19 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^3)$ and degree polynomials g = 2

n	Formulas
4	$3h(3y_2 + y_4)/4$
5	$h(29y_2 + 13y_3 + 7y_4 + 11y_5)/15$
6	$h(146y_2 + 95y_3 + 64y_4 + 53y_5 + 62y_6)/84$
7	$3h(75y_2 + 57y_3 + 44y_4 + 36y_5 + 33y_6 + 35y_7)/140$
8	$h(109y_2 + 90y_3 + 75y_4 + 64y_5 + 57y_6 + 54y_7 + 55y_8)/72$
9	$h(91y_2 + 79y_3 + 69y_4 + 61y_5 + 55y_6 + 51y_7 + 49y_8 + 49y_9)/63$
10	$3h(1428y_2 + 1281y_3 + 1154y_4 + 1047y_5 + 960y_6 + 893y_7 + 846y_8 + 819y_9 + 812y_{10})/3080$
11	$h(534y_2 + 490y_3 + 451y_4 + 417y_5 + 388y_6 + 364y_7 + 345y_8 + 331y_9 + 322y_{10} + 318y_{11})/396$
12	$h(3075y_2 + 2868y_3 + 2681y_4 + 2514y_5 + 2367y_6 + 2240y_7 + 2133y_8 + 2046y_9 + 1979y_{10} + 1932y_{11} + 1905y_{12})/2340$
13	$3h(429y_2 + 405y_3 + 383y_4 + 363y_5 + 345y_6 + 329y_7 + 315y_8 + 303y_9 + 293y_{10} + 285y_{11} + 279y_{12} + 275y_{13})/1001$
14	$h(1166y_2 + 1111y_3 + 1060y_4 + 1013y_5 + 970y_6 + 931y_7 + 896y_8 + 865y_9 + 838y_{10} + 815y_{11} + 796y_{12} + 781y_{13} + 770y_{14})/924$
15	$h(1937y_2 + 1859y_3 + 1786y_4 + 1718y_5 + 1655y_6 + 1597y_7 + 1544y_8 + 1496y_9 + 1453y_{10} + 1415y_{11} + 1382y_{12} + 1354y_{13} + 1331y_{14} + 1313y_{15})/1560$
16	$3h(10101y_2 + 9750y_3 + 9419y_4 + 9108y_5 + 8817y_6 + 8546y_7 + 8295y_8 + 8064y_9 + 7853y_{10} + 7662y_{11} + 7491y_{12} + 7340y_{13} + 7209y_{14} + 7098y_{15} + 7007y_{16})/24752$

Table 20 Integration formulas to Newton–Cotes semi-closed formulas with truncation error = $O(h^4)$ and degree polynomials g = 3

n	Formulas
6	$h(801y_2 - 54y_3 + 256y_4 + 646y_5 + 31y_6)/336$
7	$h(92y_2 + 17y_3 + 20y_4 + 52y_5 + 64y_6 + 7y_7)/42$
8	$h(295y_2 + 103y_3 + 73y_4 + 128y_5 + 191y_6 + 185y_7 + 33y_8)/144$
9	$4h(336y_2 + 156y_3 + 104y_4 + 131y_5 + 188y_6 + 226y_7 + 196y_8 + 49y_9)/693$
10	$h(2282y_2 + 1253y_3 + 857y_4 + 891y_5 + 1152y_6 + 1437y_7 + 1543y_8 + 1267y_9 + 406y_{10})/1232$
11	$h(4584y_2 + 2814y_3 + 2004y_4 + 1889y_5 + 2204y_6 + 2684y_7 + 3064y_8 + 3079y_9 + 2464y_{10} + 954y_{11})/2574$
12	$h(3222y_2 + 2142y_3 + 1586y_4 + 1427y_5 + 1538y_6 + 1792y_7 + 2062y_8 + 2221y_9 + 2142y_{10} + 1698y_{11} + 762y_{12})/1872$
13	$4h(1254y_2 + 885y_3 + 678y_4 + 598y_5 + 610y_6 + 679y_7 + 770y_8 + 848y_9 + 878y_{10} + 825y_{11} + 654y_{12} + 330y_{13})/3003$
14	$h(12023y_2 + 8888y_3 + 7010y_4 + 6144y_5 + 6045y_6 + 6468y_7 + 7168y_8 + 7900y_9 + 8419y_{10} + 8480y_{11} + 7838y_{12} + 6248y_{13} + 3465y_{14})/7392$

Table 21 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^5)$ and degree polynomials g = 4

n	Formulas
6	$5h(85y_2 - 70y_3 + 120y_4 - 10y_5 + 19y_6)/144$
7	$h(559y_2 - 212y_3 + 298y_4 + 458y_5 + 23y_6 + 134y_7)/210$
8	$h(19381y_2 - 1699y_3 + 5067y_4 + 13352y_5 + 11557y_6 + 2811y_7 + 4971y_8)/7920$
9	$h(7903y_2 + 923y_3 + 1573y_4 + 4141y_5 + 5281y_6 + 4013y_7 + 1723y_8 + 2163y_9)/3465$
10	$h(24574y_2 + 6559y_3 + 5299y_4 + 10449y_5 + 15048y_6 + 15519y_7 + 11669y_8 + 6689y_9 + 7154y_{10})/11440$
11	$5h(1050y_2 + 400y_3 + 275y_4 + 400y_5 + 574y_6 + 670y_7 + 635y_8 + 490y_9 + 330y_{10} + 324y_{11})/2574$
12	$h(18246y_2 + 8574y_3 + 5794y_4 + 6779y_5 + 9114y_6 + 11096y_7 11734y_8 + 10749y_9 + 8574y_{10} + 6354y_{11} + 5946y_{12})/9360$
13	$h(16071y_2 + 8691y_3 + 6009y_4 + 6149y_5 + 7604y_6 + 9236y_7 + 10276y_8 + 10324y_9 + 9349y_{10} + 7689y_{11} + 6051y_{12} + 5511y_{13})/8580$

Table 22 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^6)$ and degree polynomials g = 5

n	Formulas
8	$h(95751y_2 - 79704y_3 + 111403y_4 + 53408y_5 - 44907y_6 + 84152y_7 + 1657y_8)/31680$
9	$2h(9038y_2 - 4727y_3 + 5588y_4 + 7487y_5 + 1262y_6 - 401y_7 + 7184y_8 + 309y_9)/6435$
10	$3h(40260y_2 - 11865y_3 + 14560y_4 + 30795y_5 + 20064y_6 + 3829y_7 + 8064y_8 + 29529y_9 + 2044y_{10})/45760$
11	$h(6438y_2 - 772y_3 + 1573y_4 + 4178y_5 + 4058y_6 + 2162y_7 + 997y_8 + 2252y_9 + 4422y_{10} + 432y_{11})/2574$
12	$h(178671y_2 + 3186y_3 + 35451y_4 + 97836y_5 + 116516y_6 + 88768y_7 + 50268y_8 + 42388y_9 + 79493y_{10} + 116238y_{11} + 14856y_{12})/74880$
13	$3h(18546y_2 + 2391y_3 + 3532y_4 + 8767y_5 + 11672y_6 + 10764y_7 + 7664y_8 + 5260y_9 + 5870y_{10} + 9405y_{11} + 11532y_{12} + 1837y_{13})/24310$

Table 23 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^7)$ and degree polynomials g = 6

n	Formulas
8	$7h(4417y_2 - 7224y_3 + 14637y_4 - 11648y_5 + 8547y_6 - 840y_7 + 751y_8)/8640$
9	$h(446429y_2 - 532699y_3 + 836193y_4 - 19711y_5 - 281161y_6 + 584655y_7 - 32437y_8 + 79811y_9)/135135$
10	$h(4939076y_2 - 4270873y_3 + 5443568y_4 + 3411419y_5 - 1452160y_6 + 579989y_7 + 4761488y_8 + 75497y_9 + 926396y_{10})/1601600$
11	$h(1255641y_2 - 767905y_3 + 844454y_4 + 1050018y_5 + 217592y_6 - 100936y_7 + 515610y_8 + 958526y_9 + 104687y_{10} + 246633y_{11})/432432$

Table 24 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^8)$ and degree polynomials g = 7

n	Formulas
10	$h(23230289y_2 - 37927402y_3 + 70410062y_4 - 34990114y_5 - 5808640y_6 + 50955746y_7 - 29589838y_8 + 21145898y_9 + 231599y_{10})/6406400$
11	$h(12522696y_2 - 16186985y_3 + 24853224y_4 + 292998y_5 - 9660008y_6 + 10651584y_7 + 13014840y_8 - 9527894y_9 + 10549632y_{10} + 246633y_{11})/3675672$

Table 25 Integration formulas similar to Newton–Cotes semi-closed formulas with truncation error = $O(h^9)$ and degree polynomials g = 8

n	Formulas
10	$9h(20727y_2 - 50886y_3 + 129666y_4 - 177102y_5 + 182880y_6 - 110322y_7 + 51966y_8 - 4986y_9 + 2857y_{10})/44800$

Table 26 Integration formulas similar to Newton–Cotes semi-open formulas with truncation error = $O(h^1)$ and degree polynomials g = 0

Points	Formulas
4	$h(y_1 + y_2 + y_3)$
5	$h(y_1 + y_2 + y_3 + y_4)$
6	$h(y_1 + y_2 + y_3 + y_4 + y_5)$
for n	$h(y_1 + y_2 + y_3 + y_4 + \cdots + y_{n-2} + y_{n-1})$

Rules are reflected in relation to semi-closed formulas

Table 27 Integration formulas similar to Newton–Cotes semi-open formulas with truncation error = $O(h^2)$ and degree polynomials g = 1

n	Formulas
4	$h(y_1 + 4y_2 + 7y_3)/4$
5	$2h(y_1 + 2y_2 + 3y_3 + 4y_4)/5$

Rules are reflected in relation to semi-closed formulas

9 Conclusion

The integration of the polynomials obtained by interpolation using splines allowed us to obtain new and previously unknown integration formulas with a high order of truncation errors. Many different integration formulas, similar to the Newton–Cotes formulas, were obtained in this study. These new integration formulas can be used in many engineering, mathematics, and physics research applications. These new integration formulas may also be compared with the application of Newton–Cotes formulas in different fields of science. There seems to be a strong relationship between the order of degree of the polynomials and the order of the truncation errors. The present article opens a new field of research to obtain numerical methods for derivation, integration and resolution of differential equations. Another interesting observation is that the interpolation by splines approach generates a smooth adjustment between intervals and a minimum variation in curve fitting, which can help in the stability of the integration. It is believed that more new and important research on this subject can be made, given the tremendous evolution of numerical methods in engineering.

Table 28 Integration formulas similar to Newton–Cotes closed formulas with no middle point and truncation error = $O(h^2)$ and degree polynomials g = 1

Points (n)	Formulas
3	$h(y_1 + y_3)$
4	$3h(y_1 + y_4)/2$
5	$h(y_1 + y_2 + y_4 + y_5)$
6	$5h(y_1 + y_2 + y_4 + y_5)/4$
7	$h(y_1 + y_2 + y_3 + y_5 + y_6 + y_7)$
8	$7h(y_1 + y_2 + y_3 + y_6 + y_7 + y_8)/6$
9	$h(y_1 + y_2 + y_3 + y_4 + y_6 + y_7 + y_8 + y_9)$
10	$9h(y_1 + y_2 + y_3 + y_4 + y_7 + y_8 + y_9 + y_{10})/8$
Odd n	$h(y_1 + y_2 + y_3 + \cdots + y_{\frac{n+1}{2}-1} + y_{\frac{n+1}{2}+1} + y_{\frac{n+1}{2}+2} + y_{\frac{n+1}{2}+3} + \cdots + y_{n-1} + y_n)$
Even n	$(n-1)h(y_1 + y_2 + y_3 + \cdots + y_{\frac{n}{2}-1} + y_{\frac{n}{2}+2} + y_{\frac{n}{2}+3} + y_{\frac{n}{2}+4} + \cdots + y_{n-1} + y_n)/(n-2)$

Table 29 Integration formulas similar to Newton–Cotes closed formulas with no middle point and truncation error = $O(h^4)$ and degree polynomials g = 2

n	Formulas
5	$2h(y_1 + 8y_2 + 8y_4 + y_5)/9$
6	$5h(-y_1 + 25y_2 + 25y_5 - y_6)/48$
7	$3h(11y_1 + 36y_2 + 51y_3 + 51y_5 + 36y_6 + 11y_7)/98$
8	$7h(16y_1 + 169y_2 + 271y_3 + 271y_6 + 169y_7 + 16y_8)/912$
9	$2h(83y_1 + 174y_2 + 239y_3 + 278y_4 + 278y_6 + 239y_7 + 174y_8 + 83y_9)/387$
10	$9h(11y_1 + 39y_2 + 60y_3 + 74y_4 + 74y_7 + 60y_8 + 39y_9 + 11y_{10})/368$
11	$h(281y_1 + 461y_2 + 601y_3 + 701y_4 + 761y_5 + 761y_7 + 701y_8 + 601y_9 + 461y_{10} + 281y_{11})/561$
12	$11h(512y_1 + 1137y_2 + 1637y_3 + 2012y_4 + 2262y_5 + 2262y_8 + 2012y_9 + 1637y_{10} + 1137y_{11} + 512y_{12})/15120$
13	$6h(499y_1 + 708y_2 + 879y_3 + 1012y_4 + 1107y_5 + 1164y_6 + 1164y_8 + 1107y_9 + 1012y_{10} + 879y_{11} + 708y_{12} + 499y_{13})/5369$
14	$13h(721y_1 + 1243y_2 + 1678y_3 + 2026y_4 + 2287y_5 + 2461y_6 + 2461y_9 + 2287y_{10} + 2026y_{11} + 1678y_{12} + 1243y_{13} + 721y_{14})/20832$
15	$h(485y_1 + 628y_2 + 749y_3 + 848y_4 + 925y_5 + 980y_6 + 1013y_7 + 1013y_9 + 980y_{10} + 925y_{11} + 848y_{12} + 749y_{13} + 628y_{14} + 485y_{15})/804$
16	$5h(160y_1 + 237y_2 + 303y_3 + 358y_4 + 402y_5 + 435y_6 + 457y_7 + 457y_{10} + 435y_{11} + 402y_{12} + 358y_{13} + 303y_{14} + 237y_{15} + 160y_{16})/1568$
17	$2h(343y_1 + 418y_2 + 483y_3 + 538y_4 + 583y_5 + 618y_6 + 643y_7 + 658y_8 + 658y_{10} + 643y_{11} + 618y_{12} + 583y_{13} + 538y_{14} + 483y_{15} + 418y_{16} + 343y_{17})/1071$

Table 30 Integration formulas similar to Newton–Cotes closed formulas with no middle point and truncation error = $O(h^6)$ and degree polynomials g = 4

n	Formulas
7	$3h(13y_1 + 32y_2 + 55y_3 + 55y_5 + 32y_6 + 13y_7)/100$
8	$7h(562y_1 + 245y_2 + 2793y_3 + 2793y_6 + 245y_7 + 562y_8)/7200$
9	$4h(123y_1 + 249y_2 + 346y_3 + 407y_4 + 407y_6 + 346y_7 + 249y_8 + 123y_9)/1125$
10	$9h(7993y_1 + 6729y_2 + 17499y_3 + 30259y_4 + 30259y_7 + 17499y_8 + 6729y_9 + 7993y_{10})/124960$
11	$5h(35082y_1 + 62445y_2 + 80610y_3 + 91915y_4 + 98030y_5 + 98030y_7 + 91915y_8 + 80610y_9 + 62445y_{10} + 35082y_{11})/368082$
12	$11h(126637y_1 + 126687y_2 + 203857y_3 + 306727y_4 + 396732y_5 + 396732y_8 + 306727y_9 + 203857y_{10} + 126687y_{11} + 126637y_{12})/2321280$

Table 31 Integration formulas similar to Newton–Cotes closed formulas with no middle point and truncation error = $O(h^8)$ and degree polynomials g = 6

n	Formulas
9	$4h(2459y_1 + 12528y_2 + 2072y_3 + 16016y_4 + 16016y_6 + 2072y_7 + 12528y_8 + 2459y_9)/33075$
10	$9h(1711y_1 + 13041y_2 - 5022y_3 + 21630y_4 + 21630y_7 - 5022y_8 + 13041y_9 + 1711y_{10})/62720$

Table 32 Integration formulas similar to Newton–Cotes closed formulas with no middle point and truncation error = $O(h^{10})$ and degree polynomials g = 8

n	Formulas
11	$5h(33911y_1 + 170560y_2 + 53055y_3 + 131520y_4 + 182490y_5 + 182490y_7 + 131520y_8 + 53055y_9 + 170560y_{10} + 33911y_{11})/571536$

Table 33 Integration formulas similar to Newton–Cotes open formulas with no middle point and truncation error = $O(h^2)$ and degree polynomials g = 1

Points (n)	Formulas
5	$2h(y_2 + y_4)$
6	$5h(y_2 + y_5)/2$
7	$3h(y_2 + y_3 + y_5 + y_6)/2$
8	$7h(y_2 + y_3 + y_6 + y_7)/4$
9	$4h(y_2 + y_3 + y_4 + y_6 + y_7 + y_8)/3$
10	$3h(y_2 + y_3 + y_4 + y_7 + y_8 + y_9)/2$
11	$5h(y_2 + y_3 + y_4 + y_5 + y_7 + y_8 + y_9 + y_{10})/4$
12	$11h(y_2 + y_3 + y_4 + y_5 + y_8 + y_9 + y_{10} + y_{11})/8$
13	$6h(y_2 + y_3 + y_4 + y_5 + y_6 + y_8 + y_9 + y_{10} + y_{11} + y_{12})/5$
14	$13h(y_2 + y_3 + y_4 + y_5 + y_6 + y_9 + y_{10} + y_{11} + y_{12} + y_{13})/10$
Odd n	$(n-1)h(y_2 + y_3 + \cdots + y_{\frac{n+1}{2}-1} + y_{\frac{n+1}{2}+1} + y_{\frac{n+1}{2}+2} + y_{\frac{n+1}{2}+3} + \cdots + y_{n-1})/(n-3)$
Even n	$(n-1)h(y_2 + y_3 + \cdots + y_{\frac{n}{2}-1} + y_{\frac{n}{2}+2} + y_{\frac{n}{2}+3} + y_{\frac{n}{2}+4} + \cdots + y_{n-1})/(n-4)$

Table 34 Integration formulas similar to Newton–Cotes open formulas with no middle point and truncation error = $O(h^4)$ and degree polynomials g = 2

n	Formulas
7	$h(2y_2 + y_3 + y_5 + 2y_6)$
8	$7h(11y_2 + 13y_3 + 13y_6 + 11y_7)/48$
9	$4h(62y_2 + 47y_3 + 38y_4 + 38y_6 + 47y_7 + 62y_8)/147$
10	$9h(48y_2 + 51y_3 + 53y_4 + 53y_7 + 51y_8 + 48y_9)/304$
11	$5h(236y_2 + 201y_3 + 176y_4 + 161y_5 + 161y_7 + 176y_8 + 201y_9 + 236y_{10})/774$
12	$11h(133y_2 + 137y_3 + 140y_4 + 142y_5 + 142y_8 + 140y_9 + 137y_{10} + 133y_{11})/1104$
13	$3h(444y_2 + 399y_3 + 364y_4 + 339y_5 + 324y_6 + 324y_8 + 339y_9 + 364y_{10} + 399y_{11} + 444y_{12})/935$
14	$13h(1472y_2 + 1497y_3 + 1517y_4 + 1532y_5 + 1542y_6 + 1542y_9 + 1532y_{10} + 1517y_{11} + 1497y_{12} + 1472y_{13})/15120$
15	$7h(446y_2 + 413y_3 + 386y_4 + 365y_5 + 350y_6 + 341y_7 + 341y_9 + 350y_{10} + 365y_{11} + 386y_{12} + 413y_{13} + 446y_{14})/2301$
16	$15h(567y_2 + 573y_3 + 578y_4 + 582y_5 + 585y_6 + 587y_7 + 587y_{10} + 585y_{11} + 582y_{12} + 578y_{13} + 573y_{14} + 567y_{15})/6944$
17	$8h(273y_2 + 217y_3 + 206y_4 + 197y_5 + 190y_6 + 185y_7 + 182y_8 + 182y_{10} + 185y_{11} + 190y_{12} + 197y_{13} + 206y_{14} + 217y_{15} + 230y_{16})/1407$
18	$17h(992y_2 + 999y_3 + 1005y_4 + 1010y_5 + 1014y_6 + 1017y_7 + 1019y_8 + 1019y_{11} + 1017y_{12} + 1014y_{13} + 1010y_{14} + 1005y_{15} + 999y_{16} + 992y_{17})/14112$
19	$9h(336y_2 + 321y_3 + 308y_4 + 297y_5 + 288y_6 + 281y_7 + 276y_8 + 273y_9 + 273y_{11} + 276y_{12} + 281y_{13} + 288y_{14} + 297y_{15} + 308y_{16} + 321y_{17} + 336y_{18})/2380$

Table 35 Integration formulas similar to Newton–Cotes open formulas with no middle point and truncation error = $O(h^6)$ and degree polynomials g = 4

n	Formulas
9	$2h(321y_2 - 206y_3 + 335y_4 + 335y_6 - 206y_7 + 321y_8)/225$
10	$9h(258y_2 - 195y_3 + 337y_4 + 337y_7 - 195y_8 + 258y_9)/800$
11	$h(2469y_2 - 303y_3 + 563y_4 + 2221y_5 + 2221y_7 + 563y_8 - 303y_9 + 2469y_{10})/990$
12	$11h(23727y_2 - 4769y_3 + 5261y_4 + 26901y_5 + 26901y_8 + 5261y_9 - 4769y_{10} + 23727y_{11})/102240$
13	$6h(38466y_2 + 6366y_3 + 6086y_4 + 19111y_5 + 32216y_6 + 32216y_8 + 19111y_9 + 6086y_{10} + 6366y_{11} + 38466y_{12})/102245$
14	$13h(350339y_2 + 35889y_3 + 39839y_4 + 191849y_5 + 364164y_6 + 364164y_9 + 191849y_{10} + 39839y_{11} + 35889y_{12} + 350339y_{13})/1964160$

Table 36 Integration formulas similar to Newton–Cotes open formulas with no middle point and truncation error = $O(h^8)$ and degree polynomials g = 6

n	Formulas
11	$h(17944y_2 - 22977y_3 + 36232y_4 - 4739y_5 - 4739y_7 + 36232y_8 - 22977y_9 + 17944y_{10})/5292$
12	$11h(494057y_2 - 503433y_3 + 693486y_4 + 162610y_5 + 162610y_8 + 693486y_9 - 503433y_{10} + 494057y_{11})/1693440$
13	$2h(3414170y_2 - 2394394y_3 + 2461902y_4 + 3143563y_5 + 206584y_6 + 206584y_8 + 3143563y_9 + 2461902y_{10} - 2394394y_{11} + 3414170y_{12})/2277275$

Table 37 Integration formulas similar to Newton–Cotes open formulas with no middle point and truncation error = $O(h^{10})$ and degree polynomials g = 8

n	Formulas
13	$h(87805y_2 - 185176y_3 + 399033y_4 - 355548y_5 + 186186y_6 + 186186y_8 - 355548y_9 + 399033y_{10} - 185176y_{11} + 87805y_{12})/22050$

Table 38 Integration formulas similar to Newton–Cotes open formulas without two points with truncation error = $O(h^2)$ and degree polynomials g = 1

Points (n)	Formulas
n = 5	$4hy_3$
n = 6	$5h(y_3 + y_4)/2$
n = 7	$2h(y_3 + y_4 + y_5)$
n = 8	$7h(y_3 + y_4 + y_5 + y_6)/4$
n = 9	$8h(y_3 + y_4 + y_5 + y_6 + y_7)/5$
for n	$(n-1)h(y_3 + y_4 + y_5 + \cdots + y_{n-4} + y_{n-3} + y_{n-2})/(n-4)$

Table 39 Integration formulas similar to Newton–Cotes open formulas without two points with truncation error = $O(h^4)$ and degree polynomials g = 2

n	Formulas
7	$3h(3y_3 - 4y_4 + 3y_5)$
8	$7h(23y_3 - 11y_4 - 11y_5 + 23y_6)/24$
9	$8h(71y_3 - 4y_4 - 29y_5 - 4y_6 + 71y_7)/105$
10	$9h(57y_3 + 11y_4 - 12y_5 - 12y_6 + 11y_7 + 57y_8)/112$
11	$5h(101y_3 + 36y_4 - 3y_5 - 16y_6 - 3y_7 + 36y_8 + 101y_9)/126$
12	$11h(329y_3 + 155y_4 + 39y_5 - 19y_6 - 19y_7 + 39y_8 + 155y_9 + 329y_{10})/1008$
13	$4h(63y_3 + 35y_4 + 15y_5 + 3y_6 - y_7 + 3y_8 + 15y_9 + 35y_{10} + 63y_{11})/77$
14	$13h(1842y_3 + 1142y_4 + 617y_5 + 267y_6 + 92y_7 + 92y_8 + 267y_9 + 617y_{10} + 1142y_{11} + 1842y_{12})/7920$
15	$7h(519y_3 + 348y_4 + 215y_5 + 120y_6 + 63y_7 + 44y_8 + 63y_9 + 120y_{10} + 215y_{11} + 348y_{12} + 519y_{13})/1287$
16	$15h(1419y_3 + 1009y_4 + 681y_5 + 435y_6 + 271y_7 + 189y_8 + 189y_9 + 271y_{10} + 435y_{11} + 681y_{12} + 1009y_{13} + 1419y_{14})/8008$
17	$16h(43y_3 + 32y_4 + 23y_5 + 16y_6 + 11y_7 + 8y_8 + 7y_9 + 8y_{10} + 11y_{11} + 16y_{12} + 23y_{13} + 32y_{14} + 43y_{15})/273$
18	$17h(1235y_3 + 953y_4 + 718y_5 + 530y_6 + 389y_7 + 295y_8 + 248y_9 + 248y_{10} + 295y_{11} + 389y_{12} + 530y_{13} + 718y_{14} + 953y_{15} + 1235y_{16})/8736$
19	$9h(7917y_3 + 6292y_4 + 4917y_5 + 3792y_6 + 2917y_7 + 2292y_8 + 1917y_9 + 1792y_{10} + 1917y_{11} + 2292y_{12} + 2917y_{13} + 3792y_{14} + 4917y_{15} + 6292y_{16} + 7917y_{17})/30940$

Table 40 Integration formulas similar to Newton–Cotes open formulas without two points with truncation error = $O(h^6)$ and degree polynomials g = 4

n	Formulas
9	$8h(86y_3 - 224y_4 + 321y_5 - 224y_6 + 86y_7)/45$
10	$9h(219y_3 - 397y_4 + 258y_5 + 258y_6 - 397y_7 + 219y_8)/160$
11	$5h(260y_3 - 335y_4 + 50y_5 + 302y_6 + 50y_7 - 335y_8 + 260y_9)/126$
12	$11h(2327y_3 - 2133y_4 - 483y_5 + 1729y_6 + 1729y_7 - 483y_8 - 2133y_9 + 2327y_{10})/2880$
13	$2h(2796y_3 - 1789y_4 - 999y_5 + 1101y_6 + 2072y_7 + 1101y_8 - 999y_9 - 1789y_{10} + 2796y_{11})/715$
14	$13h(8526y_3 - 3634y_4 - 3339y_5 + 1341y_6 + 5026y_7 + 5026y_8 + 1341y_9 - 3339y_{10} - 3634y_{11} + 8526y_{12})/15840$
15	$7h(11664y_3 - 2994y_4 - 4324y_5 + 121y_6 + 4946y_7 + 6914y_8 + 4946y_9 + 121y_{10} - 4324y_{11} - 2994y_{12} + 11664y_{13})/12870$
16	$15h(7095y_3 - 845y_4 - 2295y_5 - 523y_6 + 2020y_7 + 3700y_8 + 3700y_9 + 2020y_{10} - 523y_{11} - 2295y_{12} - 845y_{13} + 7095y_{14})/18304$
17	$16h(36762y_3 - 198y_4 - 9723y_5 - 4242y_6 + 6578y_7 + 15832y_8 + 19377y_9 + 15832y_{10} + 6578y_{11} - 4242y_{12} - 9723y_{13} - 198y_{14} + 36762y_{15})/109395$

Table 41 Integration formulas similar to Newton–Cotes open formulas without two points with truncation error = $O(h^8)$ and degree polynomials g = 6

n	Formulas
11	$5h(3445y_3 - 13320y_4 + 28575y_5 - 35888y_6 + 28575y_7 - 13320y_8 + 3445y_9)/756$
12	$11h(211069y_3 - 656261y_4 + 999729y_5 - 494057y_6 - 494057y_7 + 999729y_8 - 656261y_9 + 211069y_{10})/120960$
13	$2h(621128y_3 - 1579924y_4 + 1696619y_5 + 197492y_6 - 1420180y_7 + 197492y_8 + 1696619y_9 - 1579924y_{10} + 621128y_{11})/75075$

Table 42 Integration formulas similar Newton–Cotes open no two points with truncation error = $O(h^{10})$ and degree polynomials g = 8

n	Formulas
13	$h(5499y_3 - 28192y_4 + 80802y_5 - 144864y_6 + 175610y_7 - 144864y_8 + 80802y_9 - 28192y_{10} + 5499y_{11})/175$

Acknowledgements The authors thank the Conselho Nacional de Desenvolvimento Cientifico e Tecnologico-CNPq-"National Counsel of technological and scientific Development" and the Fundaçao de Amparo a Pesquisa de Minas Gerais-FAPEMIG-"Foundation for Research Support of Minas Gerais". This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Compliance with Ethical Standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Chapra SC (2017) Applied numerical methods with MATLAB® for engineers and scientists, 4th edn. McGraw-Hill Education, ISBN-13: 978-0073397962
2. Yang WY, Cao W, Chung TS, Morris J (2005) Applied numerical methods using MATLAB®. Wiley, Hoboken
3. Greenspan D, Carnahan B, Luther HA, Wilkes JO (2006) Applied numerical methods. Math Comput. https://doi.org/10.2307/2004855
4. Canova F (2019) Methods for Applied macroeconomic research. Princeton University Press, Princeton
5. van der Meer FP (2012) Mesolevel modeling of failure in composite laminates: constitutive, kinematic and algorithmic aspects. Arch Comput Methods Eng 19:381. https://doi.org/10.1007/s11831-012-9076-y
6. Davis PJ, Rabinowitz P (2007) Methods of numerical integration (Dover Books on Mathematics), 2nd edn. ISBN-13: 978-0486453392
7. Arthur DW, Davis PJ, Rabinowitz P (1986) Methods of numerical integration. Math Gaz. https://doi.org/10.2307/3615859
8. Dimov IT (2005) Monte Carlo methods for applied scientists. World Scientific Publishing Company, ISBN-13: 978-9810223298
9. Evans M, Swartz T (2000) Approximating integrals via Monte Carlo and deterministic methods. Oxford University Press, ISBN-13: 978-0198502784
10. Press WH, Teukolsky SA, Vetterling WT, Flannery BP (2007) Numerical recipes, 3rd edn. In: The art of scientific computing. Cambridge UniversityPress, ISBN-13: 978-0521880688
11. Tierney L, Kadane JB (1986) Accurate approximations for posterior moments and marginal densities. J Am Stat Assoc. https://doi.org/10.1080/01621459.1986.10478240
12. Bartoň M, Kosinka J (2019) On numerical quadrature for C1 quadratic Powell–Sabin 6-split macro-triangles. J Comput Appl Math. https://doi.org/10.1016/j.cam.2018.07.051
13. Kosinka J, Bartoň M (2019) Gaussian quadrature for C1 cubic Clough–Tocher macro-triangles. J Comput Appl Math. https://doi.org/10.1016/j.cam.2018.10.036
14. Busenberg SN, Fisher D (1984) Spline quadrature formulas. J Approx Theory. https://doi.org/10.1016/0021-9045(84)90040-6
15. Patriarca M, Farrell P, Fuhrmann J, Koprucki T (2019) Highly accurate quadrature-based Scharfetter–Gummel schemes for charge transport in degenerate semiconductors. Comput Phys Commun. https://doi.org/10.1016/j.cpc.2018.10.004
16. Bailey DH, Borwein JM (2011) High-precision numerical integration: progress and challenges. J Symb Comput. https://doi.org/10.1016/j.jsc.2010.08.010
17. Skrainka BS, Judd KL (2012) High performance quadrature rules: how numerical integration affects a popular model of product differentiation. SSRN Electron J. https://doi.org/10.2139/ssrn.1870703
18. Reeger JA, Fornberg B (2018) Numerical quadrature over smooth surfaces with boundaries. J Comput Phys. https://doi.org/10.1016/j.jcp.2017.11.010
19. Aslanyan V, Aslanyan AG, Tallents GJ (2017) Efficient calculation of degenerate atomic rates by numerical quadrature on GPUs. Comput Phys Commun. https://doi.org/10.1016/j.cpc.2017.06.003
20. Mohammed OH, Saeed MA (2019) Numerical solution of thin plates problem via differential quadrature method using G-spline. J King Saud Univ Sci. https://doi.org/10.1016/j.jksus.2018.04.001
21. Burg COE (2012) Derivative-based closed Newton–Cotes numerical quadrature. Appl Math Comput. https://doi.org/10.1016/j.amc.2011.12.060
22. Chakrabarti A (1996) Modified quadrature rules based on a generalised mixed interpolation formula. J Comput Appl Math. https://doi.org/10.1016/S0377-0427(96)00107-0
23. Schoenberg IJ (1964) Spline Interpolation and best quadrature formulae. Bull Am Math Soc. https://doi.org/10.1090/S0002-9904-1964-11054-5
24. Taghvafard H (2011) A new quadrature rule derived from spline interpolation with error analysis. World Acad Sci Eng Technol. https://doi.org/10.5281/zenodo.1070225
25. Karlin S (1971) Best quadrature formulas and splines. J Approx Theory. https://doi.org/10.1016/0021-9045(71)90040-2
26. Nouy A (2009) Recent developments in spectral stochastic methods for the numerical solution of stochastic partial differential equations. Arch Comput Methods Eng 16:251. https://doi.org/10.1007/s11831-009-9034-5
27. Caicedo M, Mroginski JL, Toro S et al (2018) High performance reduced order modeling techniques based on optimal energy quadrature: application to geometrically non-linear multiscale inelastic material modeling. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-9258-3
28. Badia S, Martín AF, Principe J (2018) FEMPAR: an object-oriented parallel finite element framework. Arch Comput Methods Eng 25:195. https://doi.org/10.1007/s11831-017-9244-1
29. Chatterjee T, Chakraborty S, Chowdhury R (2019) A critical review of surrogate assisted robust design optimization. Arch Comput Methods Eng 26:245. https://doi.org/10.1007/s11831-017-9240-5
30. Gleim T, Kuhl D (2019) Electromagnetic analysis using high-order numerical schemes in space and time. Arch Comput Methods Eng 26:405. https://doi.org/10.1007/s11831-017-9249-9
31. Huan Z, Zhenghong G, Fang X et al (2018) Review of robust aerodynamic design optimization for air vehicles. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-9259-2
32. Mengaldo G, Wyszogrodzki A, Diamantakis M et al (2018) Current and emerging time-integration strategies in global numerical weather and climate prediction. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-9261-8
33. Rozema W, Verstappen RWCP, Veldman AEP et al (2018) Low-dissipation simulation methods and models for turbulent subsonic flow. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-09307-7
34. Scalet G, Auricchio F (2018) Computational methods for elastoplasticity: an overview of conventional and less-conventional

approaches. Arch Comput Methods Eng 25:545. https://doi.org/10.1007/s11831-016-9208-x
35. Rodríguez JM, Carbonell JM, Jonsén P (2018) Numerical methods for the modelling of chip formation. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-09313-9
36. Marussig B, Hughes TJR (2018) A review of trimming in isogeometric analysis: challenges, data exchange and simulation aspects. Arch Comput Methods Eng 25:1059. https://doi.org/10.1007/s11831-017-9220-9
37. Meier C, Popp A, Wall WA (2019) Geometrically exact finite element formulations for slender beams: Kirchhoff–love theory versus Simo–Reissner theory. Arch Comput Methods Eng 26:163. https://doi.org/10.1007/s11831-017-9232-5
38. Nodargi NA (2018) An overview of mixed finite elements for the analysis of inelastic bidimensional structures. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-9293-0
39. Meyghani B, Awang M (2019) A comparison between the flat and the curved friction stir welding (FSW) thermomechanical behaviour. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-019-09319-x
40. Laurent L, Le Riche R, Soulier B et al (2019) An overview of gradient-enhanced metamodels with applications. Arch Comput Methods Eng 26:61. https://doi.org/10.1007/s11831-017-9226-3
41. Fraz MM, Badar M, Malik AW et al (2018) Computational methods for exudates detection and macular edema estimation in retinal images: a survey. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-9281-4
42. Rappel H, Beex LAA, Hale JS et al (2019) A tutorial on Bayesian inference to identify material parameters in solid mechanics. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-09311-x
43. Moreno-García P, dos Santos JVA, Lopes H (2018) A review and study on Ritz method admissible functions with emphasis on buckling and free vibration of isotropic and anisotropic beams and plates. Arch Comput Methods Eng 25:785. https://doi.org/10.1007/s11831-017-9214-7
44. Borzacchiello D, Aguado JV, Chinesta F (2019) Non-intrusive sparse subspace learning for parametrized problems. Arch Comput Methods Eng 26:303. https://doi.org/10.1007/s11831-017-9241-4
45. Fambri F (2019) Discontinuous Galerkin methods for compressible and incompressible flows on space–time adaptive meshes: toward a novel family of efficient numerical methods for fluid dynamics. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-09308-6
46. Ramírez L, Nogueira X, Ouro P et al (2018) A higher-order chimera method for finite volume schemes. Arch Comput Methods Eng 25:691. https://doi.org/10.1007/s11831-017-9213-8
47. Zhang LW, Ademiloye AS, Liew KM (2018) Meshfree and particle methods in biomechanics: prospects and challenges. Arch Comput Methods Eng. https://doi.org/10.1007/s11831-018-9283-2
48. Sarmavuori J, Särkkä S (2019) Numerical integration as a finite matrix approximation to multiplication operator. J Comput Appl Math. https://doi.org/10.1016/j.cam.2018.12.031
49. Magalhães Cristina, Junior Pedro (2010) Higher-order Newton–Cotes formulas. J Math Stat. https://doi.org/10.3844/jmssp.2010.193.204
50. Miclăuş D, Pişcoran LI (2019) A new method for the approximation of integrals using the generalized Bernstein quadrature formula. Appl Math Comput. https://doi.org/10.1016/j.amc.2018.08.008
51. Liu G, Xiang S (2019) Clenshaw–Curtis-type quadrature rule for hypersingular integrals with highly oscillatory kernels. Appl Math Comput. https://doi.org/10.1016/j.amc.2018.08.004
52. Grylonakis ENG, Filelis-Papadopoulos CK, Gravvanis GA, Fokas AS (2019) An iterative spatial-stepping numerical method for linear elliptic PDEs using the unified transform. J Comput Appl Math. https://doi.org/10.1016/j.cam.2018.11.025
53. Sam CN, Hon YC (2019) Generalized finite integration method for solving multi-dimensional partial differential equations. Eng Anal Bound Elem. https://doi.org/10.1016/j.enganabound.2018.11.012