Calcolo (2016) 53:217–226
DOI 10.1007/s10092-015-0145-0

Creating stable quadrature rules with preassigned points by interpolation

Hassan Majidian[1]

Received: 30 September 2014 / Accepted: 15 May 2015 / Published online: 27 May 2015

Abstract A new approach for creating stable quadrature rules with preassigned points is proposed. The idea is to approximate a known stable quadrature rule by a local interpolation at the preassigned points. The construction cost of the method does not grow as the number of the preassigned points increases. The accuracy of the rule depends only on the accuracy of the chosen stable rule and that of the interpolation. The efficiency of the rule is illustrated by some numerical examples.

Keywords Numerical integration · Polynomial interpolation · Gaussian quadrature rule

Mathematics Subject Classification (2000) 41A55 · 65D30

1 Introduction

Let $[a, b]$ be a compact interval and consider the integral operator I on $C([a, b])$ defined by

$$I(f) := \int_a^b w(x) f(x)\, \mathrm{d}x, \tag{1}$$

where $w \in L^1([a, b])$ is a positive weight function. The aim of this paper is to provide efficient approximations of $I[f]$ by quadrature rules of the form

✉ Hassan Majidian
majidian@iecf.ir

[1] Department of Basic Sciences, Iranian Institute for Encyclopedia Research, PO Box 14655-478, Tehran, Iran

$$Q_N(f) = \sum_{n=1}^{N} w_n f(s_n), \tag{2}$$

when $a = s_1 < \cdots < s_N = b$ are N distinct given points. Newton–Cotes rules, which correspond to equidistant points, are some examples of such rules.

If the points are not specified, then the degree of freedom double and one can design more efficient quadrature rules such as Gaussian quadrature formulas, which have positive weights w_j. The positivity of the weights results in the stability of the rule and its convergence for any continuous integrand f.

In many applications, however, it is beneficial to consider preassigned points. For example in finite difference methods, finite volume methods, and collocation methods for ordinary and partial differential equations, as well as some discretization methods for integral equations, considering uniform meshes makes significant simplifications in their implementation. Then, in any of these methods, one needs to use Newton–Cotes rules (e.g. composite Trapezoidal and Simpson rules) in some stages of the discretization. However, low-order Newton–Cotes rules are not accurate enough, and the high-order ones suffer from instability.

The problem of designing efficient quadrature rules with equidistant points has been considered by many authors. One recent progress is due to [8], where the number of points N is assumed to be greater than the order of the quadrature rule d. If N is large enough ($N \sim d^2$, for $d \gg 1$), a least squares solution of the corresponding underdetermined system of equations for the weights yields nonnegative solutions. Then we have a quadrature rule with equidistant points and positive weights. Another strategy is by the so-called *Mock-Chebyshev* interpolation, which employs this fact that the Runge phenomenon disappears if the asymptotic distribution of the interpolation points is so that the density of the points gradually and in a specific manner[1] increases as they get close to either of the endpoints [1]. The idea is to choose a subset of the equidistant points, which best mimic a grid points with such an asymptotic distribution, e.g. the Chebyshev–Lobatto grid points (see [2,4,9,11] and references therein).

In the present paper we follow a different approach. The idea is to consider a stable rule, e.g. the Gaussian one, and approximate the abscissas by interpolation at some of the preassigned points. The main idea came from [3]. The theory and its implementation are the same for both equidistant and nonequidistant points.

2 The method of abscissa approximation (MAA)

Let $a = s_1 < \cdots < s_N = b$ be an arbitrary set of points in $[a, b]$ and $t_1^{(M)} < \cdots < t_M^{(M)}$ be the abscissas of the M-point Gauss–Christoffel quadrature formula on the interval $[a, b]$ for the weight $w(x)$. Choose a fixed integer $r > 0$, not greater than N. For each $k \in \{1, \ldots, M\}$, we select an r-tube $\mathcal{N}_k = (s_j, \ldots, s_{j+r-1})$ of r adjacent points $s_j, \ldots, s_{j+r-1}$ such that $s_j \le t_k^{(M)} \le s_{j+r-1}$ (see Fig. 1). Note that there may be several r-tubes, encompassing $t_k^{(M)}$. Although, a strategy is suggested in Sect. 3 for

[1] For the interval $[-1, 1]$, proportional to $(1 - x^2)^{-1/2}$.

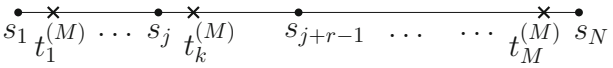

Fig. 1 Typical positions of the given (*disk*) and the Gaussian (*cross*) points in $[a, b]$

selecting one of these r-tubes, the selection of $\mathscr{N}_k$ is free of choice. Note also that such a selection is always possible since $s_1 < t_k^{(M)} < s_N$, and $r \leq N$.

Let $\bar{f}_k$ be the approximation of $f(t_k^{(M)})$ by interpolation of f at r points of $\mathscr{N}_k$, i.e.

$$\bar{f}_k = \sum_{m=j}^{j+r-1} f(s_m) l_m\left(\mathscr{N}_k; t_k^{(M)}\right), \tag{3}$$

where

$$l_m(\mathscr{N}_k; \cdot) = \prod_{\substack{n=j \\ n\neq m}}^{j+r-1} \frac{\cdot - s_n}{s_m - s_n}, \quad m = j, \ldots, j+r-1,$$

are *Lagrange fundamental polynomials* of the set $\mathscr{N}_k$.

Now define the M-point quadrature rule $Q_{N,M}(f)$ as

$$Q_{N,M}(f) := \sum_{k=1}^{M} w_k^{(M)} \bar{f}_k, \tag{4}$$

where $w_k^{(M)} > 0$ are the weights of the M-point Gauss–Christoffel quadrature formula on the interval $[a, b]$ for the weight $w(x)$.

It is possible to rewrite (4) in the form of (2) if the terms of the sum are rearranged suitably. By substituting (3) into (4),

$$Q_{N,M}(f) = \sum_{k=1}^{M} \sum_{m=j}^{j+r-1} w_k^{(M)} f(s_m) l_m\left(\mathscr{N}_k; t_k^{(M)}\right). \tag{5}$$

For each $n \in \{1, \ldots, N\}$, $f(s_n)$ may appear in several terms of the sum (5) (corresponding to some different indices k); it is also possible that it never appears. Define the set of indices

$$I(n) := \{k \mid s_n \in \mathscr{N}_k\}. \tag{6}$$

Then the coefficient of $f(s_n)$ in the sum (5) will be

$$c_n^{(M)} := \sum_{k \in I(n)} w_k^{(M)} l_n(\mathscr{N}_k; t_k^{(M)}).$$

Thus (4) can be rewritten as the following rule, that is in the form of (2):

$$Q_{N,M}(f) = \sum_{n=1}^{N} c_n^{(M)} f(s_n). \tag{7}$$

Note that $I(n)$ can be empty for some n. In this case, $c_n^{(M)} = 0$.

However, we never use (7) in practice for two reasons. Firstly, we have no claim that the coefficients $c_n^{(M)}$ are positive, and secondly, specifying the index sets $I(n)$ impose an extra effort. For computing $Q_{N,M}(f)$ in practice, it is recommended that Eq. (4) is used instead, while $\bar{f}_k$ are computed by the following modified Lagrange formula (cf. [7,10]):

$$\bar{f}_k = \ell\left(t_k^{(M)}\right) \sum_{m=j}^{j+r-1} \frac{u_m}{t_k^{(M)} - s_m} f(s_m), \tag{8}$$

where

$$\ell(\cdot) = \prod_{n=j}^{j+r-1} (\cdot - s_n), \quad u_m = \frac{1}{\prod_{n\neq m}(s_m - s_n)}.$$

Equation (8) is indeed the Lagrange interpolating polynomial, rewritten in a different form.

For increasing the accuracy of approximation by the formula (4), M and N are allowed to grow, while r is kept frozen.[2] Then the Runge effect never rises, since the degree of the Lagrange interpolating polynomials

$$x \longmapsto \sum_{m=j}^{j+r-1} f(s_m) l_m(\mathcal{N}_k; x)$$

does not grow as N, M increase. Besides, a nice rounding error analysis has been carried out in [7], due to which it has been established that the modified Lagrange formula (8) is backward stable.

The above discussion together with the positivity of the weights $w_k^{(M)}$ implies the stability of the rule (4) when constructed by (8).

2.1 Error estimate

It is well-known that,

$$\left| f\left(t_k^{(M)}\right) - \bar{f}_k \right| \leq C h^r, \tag{9}$$

where

$$h := \max_{1<n\leq N} |s_n - s_{n-1}|, \tag{10}$$

[2] This can be seen from the notation $Q_{N,M}$, where the dependency on r is suppressed. Another reason for this suppression lies in the simplification of the notation.

and $C > 0$ is a constant independent of h. We can consider the set of the points s_n as a mesh, and then h will be the *norm* of the mesh. Then,

$$\left|\sum_{k=1}^{M} w_k^{(M)} f\left(t_k^{(M)}\right) - \sum_{k=1}^{M} w_k^{(M)} \bar{f}_k\right| \leq \sum_{k=1}^{M} w_k^{(M)} \left|f\left(t_k^{(M)}\right) - \bar{f}_k\right|$$
$$\leq Ch^r \sum_{k=1}^{M} w_k^{(M)} = Ch^r.$$

Thus, the total error of the quadrature rule (4) in the absence of rounding error is bounded by

$$\left|I[f] - Q_{N,M}(f)\right| \leq E_M^{\mathrm{GC}}[f] + Ch^r, \tag{11}$$

where $E_M^{\mathrm{GC}}[f]$ denotes the absolute error of the M-point Gauss–Christoffel quadrature formula on the interval $[a, b]$ for the weight $w(x)$.

2.2 Complexity

The construction of the rule (4) with (8) consists of evaluating the Gaussian weights and abscissas and the values $l_m(\mathcal{N}_k; t_k^{(M)})$. In many cases, when $w(x)$ is a classical

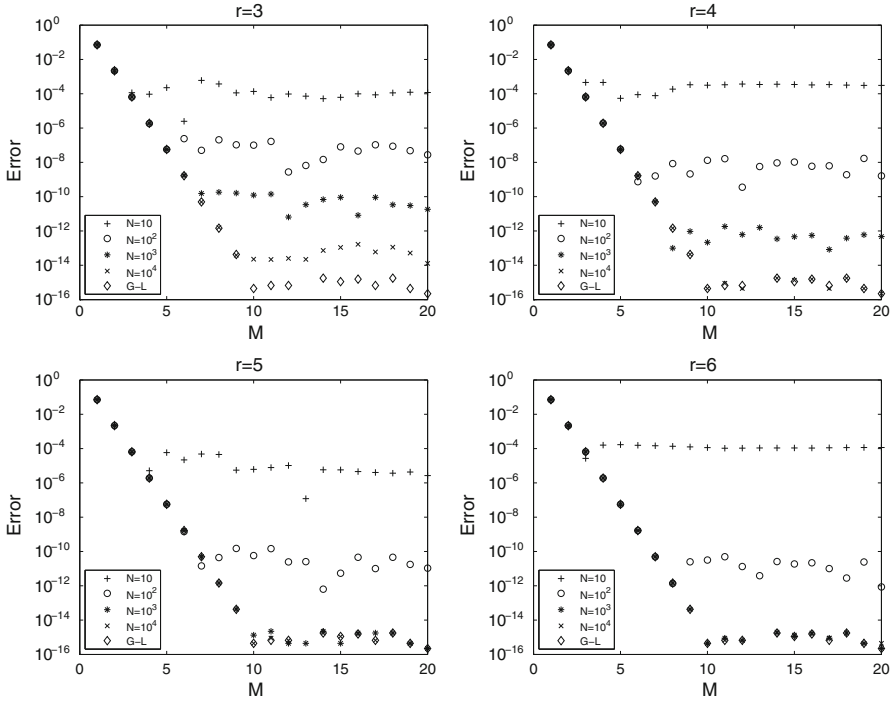

Fig. 2 The absolute error as a function of M for the rule (4) with $f(x) = 1/(1+x^2)$ and N-point uniform mesh, compared with the M-point Gauss–Legendre quadrature rule (G–L)

weight function, the Gaussian weights and abscissas are known from relevant tables, so the first stage may be carried out with no cost. Some efficient algorithms are also available, which can be used for the fast construction of classical quadrature rules as well; for instance, the algorithms proposed in [5,6] require only $\mathcal{O}(M)$ flops. In general, consider a given strategy for constructing the M-point Gauss–Christoffel quadrature formula on $[a, b]$ for the weight $w(x)$, and denote by FCQ(M) the number of flops required by that strategy. On the other hand, evaluating each $\bar{f}_k$ requires $\mathcal{O}(r^2)$ flops (see, e.g. [1]). Thus the total complexity of the rule (4) with (8) will be FCQ$(M) + \mathcal{O}(Mr^2)$.

This shows that the cost required for constructing the rule (4) does not grow by increasing N. In practice, r and M are usually of moderate values, so the construction of the rule is really fast.

3 Numerical experiences

Consider the integral (1) with $[a, b] = [-1, 1]$, $w(x) = 1$, and two samples of $f(x)$:

$$\text{(a) } f(x) = \frac{1}{1+x^2}, \quad \text{(b) } f(x) = \frac{1}{1+8x^2}.$$

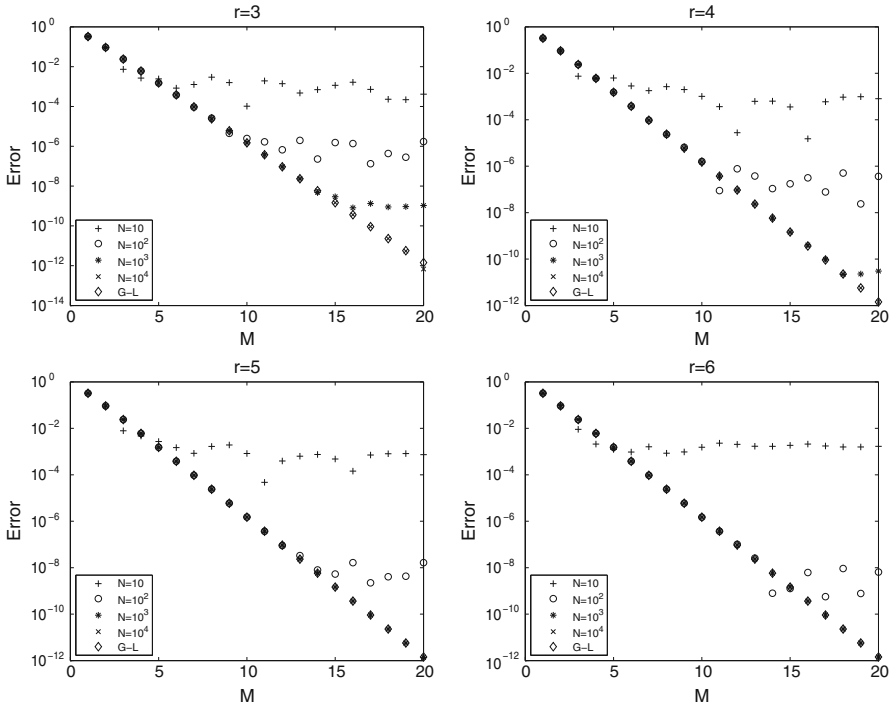

Fig. 3 The absolute error as a function of M for the rule (4) with $f(x) = 1/(1+8x^2)$ and N-point uniform mesh, compared with the M-point Gauss–Legendre quadrature rule (G–L)

3.1 Equidistant data

Assume that f is only known at the equidistant points $-1 = s_1 < \cdots < s_N = 1$ for some N. Newton-Cotes quadrature rules with large N, when applied to the above integrands, result in very poor approximations due to numerical instability (see Fig. 2 of [8]).

We follow the algorithm proposed in Sect. 2 by selecting a stable quadrature rule as the first step. Considering the M-point Gauss–Legendre quadrature rule, we apply the rule (4) with $r = 3, 4, 5, 6$ and some different values of N and M. As recommended in Sect. 2, the modified Lagrange formula (8) is employed in all of our numerical experiments reported in this section.

In order to assign r adjacent points $s_j, \ldots, s_{j+r-1}$ for each $t_k^{(M)}$, we follow a natural selecting strategy so that the first half of the points would lie on the left, and the other half would lie on the right of $t_k^{(M)}$ (if r is odd, the numbers of left and right points would differ by one). If there are not enough points on the left (resp. right), the left-most (resp. right-most) points are selected. The algorithm can then be read as follows. Find the largest n with $s_n < t_k^{(M)}$; then `indxl`, the left-most index and `indxr`, the right-most index of the adjacent points are determined by the following MATLAB code:

```
indxl=n-floor(r/2); % Almost half of the points lie on the left
indxr=indxl+r-1; % The reminder of the points lie on the right
```

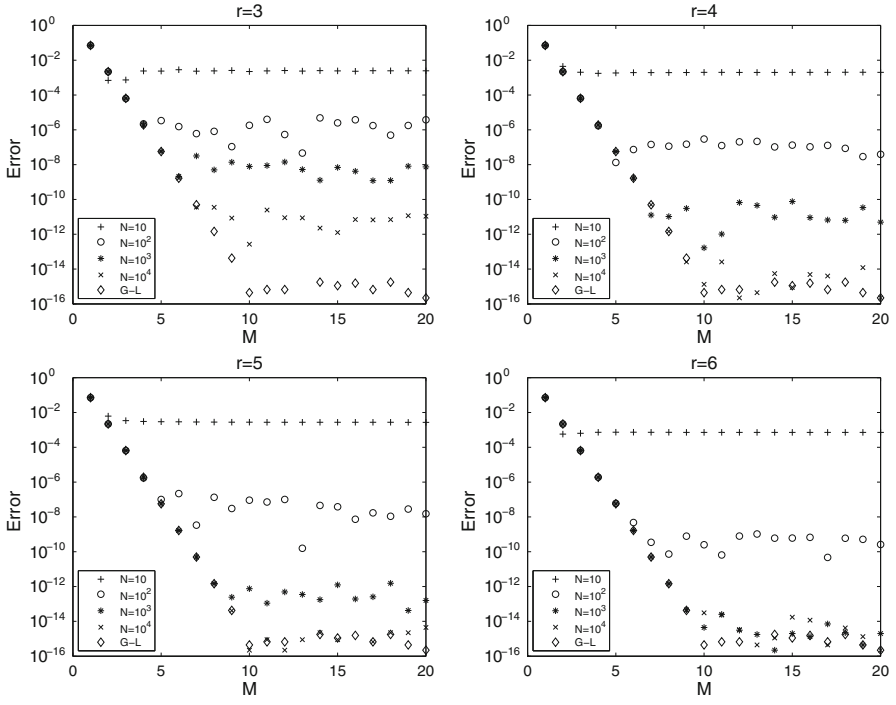

Fig. 4 The absolute error as a function of M for the rule (4) with $f(x) = 1/(1+x^2)$ and N pseudorandom points, compared with the M-point Gauss–Legendre quadrature rule (G-L)

```
if indxl<1 % There are not enough points on the left
    indxl=1;
    indxr=r;
end
if indxr>N % There are not enough points on the right
    indxl=N-r+1;
    indxr=N;
end
```

For each N, the absolute error as a function of M has been illustrated (Figs. 2 and 3). We have also tried other selecting strategies, but no meaningful differences have been observed in the results. The results show that the accuracy of the rule (4) with (8) gets close to that of the Gauss–Legendre quadrature rule as N increases.

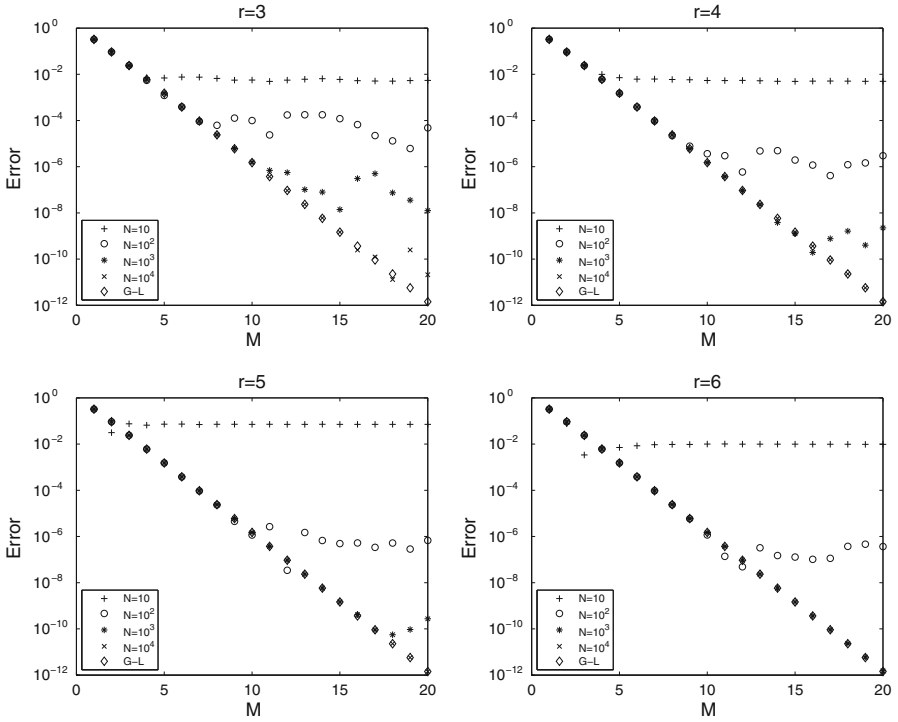

Fig. 5 The absolute error as a function of M for the rule (4) with $f(x) = 1/(1+8x^2)$ and N pseudorandom points, compared with the M-point Gauss–Legendre quadrature rule (G–L)

Table 1 The average of the mesh norm h for different N

N	10	10^2	10^3	10^4
(a)	0.5776	0.1032	0.0151	0.0019
(b)	0.5263	0.1035	0.0148	0.0020

3.2 Non-equidistant data

Assume that f is known at the non-uniform mesh $-1 = s_1 < \cdots < s_N = 1$ for some N. The points s_i are uniformly distributed pseudorandom numbers in $[-1, 1]$, generated by the MATLABcommand `rand`. We repeat the same experiments as above in Equidistant data. The results are shown by Figs. 4 and 5. For each r, we have run '`rand`' once; so for each N, the mesh (together with the mesh norm as a result) may differ for different values of r. For each N, the average of the mesh norms (h) has been calculated (see Table 1). One can see that the accuracy of the rule (4) gets close to that of the Gauss–Legendre quadrature rule as h decreases.

4 Concluding remarks

A novel stable and highly accurate numerical quadrature rule with preassigned points has been proposed. Although, the method seems very simple, it solves many problems raised in traditional quadrature rules with equidistant points. The construction cost of the rule does not grow as the number of the points increases. In some discretization methods for differential and integral equations one usually needs to refine the mesh in order to achieve a given accuracy, so the latter property is very advantageous. The main idea of the rule is to approximate the points of a known stable quadrature rule, e.g. the Gaussian or Clenshaw–Curtis quadrature rule, by interpolation at the preassigned points. Considering the set of preassigned points as a mesh, our numerical experiments show that the accuracy of the rule increases rapidly and becomes comparable to that of the Gaussian quadrature rule as the norm of the mesh decays. Employing other stable rules, e.g. Clenshaw–Curtis rules, is straightforward and does not need additional explanations.

References

1. Berrut, J.P., Trefethen, L.N.: Barycentric Lagrange interpolation. SIAM Rev. **46**(3), 501–517 (2004)
2. Boyd, J.P., Xu, F.: Divergence (Runge phenomenon) for least-squares polynomial approximation on an equispaced grid and Mock–Chebyshev subset interpolation. Appl. Math. Comput. **210**(1), 158–168 (2009)
3. Bruno, O., Hoch, D.: Numerical differentiation of approximated functions with limited order-of-accuracy deterioration. SIAM J. Numer. Anal. **50**(3), 1581–1603 (2012)
4. De Marchi, S., DellAccio, F., Mazza, M.: On the constrained Mock–Chebyshev least-squares. J. Comput. Appl. Math. **280**, 94–109 (2015)
5. Glaser, A., Liu, X., Rokhlin, V.: A fast algorithm for the calculation of the roots of special functions. SIAM J. Sci. Comput. **29**(4), 1420–1438 (2007)
6. Hale, N., Townsend, A.: Fast and accurate computation of Gauss–Legendre and Gauss–Jacobi quadrature nodes and weights. SIAM J. Sci. Comput. **35**(2), A652–A674 (2013)
7. Higham, N.J.: The numerical stability of barycentric Lagrange interpolation. IMA J. Numer. Anal. **24**(4), 547–556 (2004)
8. Huybrechs, D.: Stable high-order quadrature rules with equidistant points. J. Comput. Appl. Math. **231**(2), 933–947 (2009)
9. Platte, R.B., Trefethen, L.N., Kuijlaars, A.B.: Impossibility of fast stable approximation of analytic functions from equispaced samples. SIAM Rev. **53**(2), 308–318 (2011)

10. Rutishauser, H.: Lectures on Numerical Mathematics. Birkhäuser, Basel (1990)
11. Stengle, G.: Chebyshev interpolation with approximate nodes of unrestricted multiplicity. J. Approx. Theory **57**(1), 1–13 (1989)