THE LINEAR RATIONAL PSEUDOSPECTRAL METHOD WITH ITERATIVELY OPTIMIZED POLES FOR TWO-POINT BOUNDARY-VALUE PROBLEMS*

JEAN-PAUL BERRUT† AND HANS D. MITTELMANN‡

Abstract. An algorithm is proposed which improves upon the polynomial pseudospectral method for solving linear two-point boundary value problems. In the latter, the collocation points are the vertical projection onto the interval of points equidistant or nearly equidistant on the circle, and they therefore accumulate in the vicinity of the extremities of the interval. Thus the method is well-suited for solving problems whose solutions have boundary layers, but less good at approximating solutions with large gradients (shocks) away from the extremities of their domain of definition. Our idea is to modify the polynomial ansatz by attaching a denominator so as to make it a rational interpolant. The denominator is then successively optimized in an iterative procedure, with each step consisting of the solution of two problems: an optimization of the denominator for given values of the approximation to the solution at the interpolation points, and a collocation in the linear space of the rational interpolants with the just obtained, fixed denominator, to obtain new approximate values of the solution. We show the efficiency of a Galerkin version of the method and discuss the power of the collocation version with several numerical examples.

Key words. two-point boundary value problems, linear rational collocation, pole optimization

AMS subject classifications. Primary 65L10; secondary 41A20, 65D05

1. Introduction: the problem. Our aim in the present work is to improve upon the polynomial pseudospectral method for solving linear two-point boundary value problems (BVPs)

\[ (1.1a) \quad u''(x) + p(x)u'(x) + q(x)u(x) = f(x), \quad x \in (-1, 1), \]
\[ (1.1b) \quad u(-1) = u_L, \quad u(1) = u_R, \]

where all arising functions belong to \( C^\infty[-1, 1] \) and \( u_L \) and \( u_R \) are given real numbers. We assume that \( p, q \) and \( f \) are such that the problem is well-posed (for conditions guaranteeing the latter, see, e.g., [Kel, p. 12], [As-Ma-Ru, p. 88] or [Sto-Bul]). The generalization of the method to be presented below to nonlinear problems is straightforward, but does not bring more insight.

The version of the pseudospectral method we have in mind is that consisting in replacing the solution in (1.1a) by an interpolating polynomial in Lagrangian form between well-chosen points and collocating at those same points. Since such points are vertical projections on the interval of points (nodes) equidistant or nearly equidistant on the circle, they accumulate in the vicinity of the extremities of the interval. As a consequence, the method is well-suited for solving problems whose solutions have boundary layers. (This property can even be accentuated by conformally shifting the points to make them more tilted toward the extremities, e.g., by a sine-map [Tan-Trul].) Correspondingly, however, the center nodes are about equidistant, and further

---

*Received by the editors October 1, 1994; accepted by the editors Month, x, xxxx.
†Département de Mathématiques, Université de Fréjus, CH-1700 Fréjus/Prèsilles, Switzerland (jean-paul.berrut@unifr.ch). The work of this author has been supported by the Swiss National Science Foundation, grant #21-5272.99.
‡Department of Mathematics, Arizona State University, Tempe, Arizona 85287-1804, USA.
from each other than the same number of nodes equidistant on the whole interval. As a consequence, the pseudospectral method has a hard time approximating solutions with large gradients (shocks) away from the extremities of their domain of definition.

Several methods can be used to cope with this difficulty. The most efficient ones, as measured by the size of the error, are probably methods where the solution is not approximated by a global, but by a piecewise interpolant (see, e.g., [As-Ch-Ru, Bad-Asc, Lee-Gre]). Here we are only interested in methods which preserve the infinite differentiability of the solution. One of them consists in constructing and switching in an analytic function that displaces the interpolation points so as to have them concentrate close to the abscissae where the shocks arise [Mu-Hu-Si]. To obtain this function, the method solves a sequence of problems for values of a parameter corresponding to stiffer and stiffer problems.

We will instead develop a method that improves on the pseudospectral (polynomial) method without any change in the problem nor in the interpolation/collocation points. The idea is to modify the ansatz by attaching a denominator to the polynomial so as to make it a rational interpolant, as in [Ber-Mit2] and [Ba-Be-Du]. The denominator is then successively optimized in an iterative procedure, with each step consisting of the solution of two problems: an optimization of the denominator for given values of the approximation to \( u \) at the interpolation points, and a collocation in the linear space of the rational interpolants with the just obtained, fixed denominator, to obtain new approximate values of \( u \).

We will describe the methods for solving these two problems in Section 2. Section 3 introduces the then very simple full algorithm, whose viability is motivated in Section 4 by a theorem proving that a corresponding Galerkin algorithm yields a sequence of approximations with a certainly nonincreasing, and in most cases decreasing error in the energy norm. Finally, in Section 5, we report on numerical experiments that demonstrate the efficiency of the algorithm.

2. The ingredients of the solution. As just mentioned in the Introduction, the method presented here consists in the iterative application of two algorithms which we describe in this section.

2.1. The linear rational collocation method for boundary value problems. This generalization, suggested in [Ber-Bal] and [Ba-Be-Du], of the now classical polynomial pseudospectral method is the application to BVPs of the corresponding method for time evolution problems [Bal-Ber2]. It is based on the fact [Ber2, Ber-Mit1] that every rational function \( r \in \mathcal{R}_{N,N} \) interpolating a continuous function \( f \) between interpolation points \( x_0, \ldots, x_N \) can be written in its barycentric form

\[
 r(x) = \frac{\sum_{j=0}^{N} \beta_j (x - x_j) f(x_j)}{\sum_{j=0}^{N} \beta_j} 
\]

for some (nonunique) numbers \( \beta_j \), one per node, called weights of the interpolant. Here \( \mathcal{R}_{m,n} \) denotes the set of all rational functions with numerator degree \( \leq m \) and denominator degree \( \leq n \). The polynomial interpolant is the special case of (2.1) in which the \( \beta_j \)'s are proportional to the barycentric weights

\[
 w_j := 1 / \prod_{k \neq j} (x_j - x_k) 
\]

of polynomial interpolation [Hen]. For instance, for equidistant points the \( w_j \)'s are proportional to \((-1)^j j! \), for \( \tilde{\textrm{C}} \)\'by\( \tilde{\textrm{S}} \)ev points \( \cos \phi_j \) of the first kind to \((-1)^j j! \sin \phi_j \)
and for Čebyšev points of the second kind to \((-1)^j \eta_j\), with \(\eta_j = 1\) for all \(j\) but at the boundary points, where \(\eta_0 = \eta_N = 1/2\). In view of the presence in the problem of the boundary values \((1.1b)\) we will restrict ourselves here to sets of nodes containing the extremities \(-1\) and \(1\), i.e., Lobatto points (and in particular to Čebyšev points of the second kind in numerical computations).

For fixed \(\beta := [\beta_0, \ldots, \beta_N]^T\) the set of all interpolants \((2.1)\) is a linear space, which we denote by \(\mathcal{R}_N^{(\beta)}\). The functions

\[
L_j^{(\beta)}(x) := \frac{\beta_j}{x - x_j} \sum_{k=0}^{N} \frac{\beta_k}{x - x_k}, \quad j = 0, 1, \ldots, N,
\]

make up a basis for this space and they satisfy the Lagrange property

\[
L_j^{(\beta)}(x_i) = \delta_{ij}.
\]

In the linear rational collocation method (in barycentric form) for the nodes \(x_j\) one tries to find \(\tilde{u}\) as an interpolant

\[
\tilde{u}(x) = \sum_{j=0}^{N} \tilde{u}_j L_j^{(\beta)}(x) \in \mathcal{R}_N^{(\beta)}
\]

for some given weights \(\beta\) and some unknown values \(\tilde{u}_j\) at the \(x_j\)'s, one inserts \(\tilde{u}\) into \((1.1a)\) and one collocates at the same interior \(x_j\)'s, for simplicity (collocation points different from the interpolation points, as in [Fun], are equally possible). This yields the following linear system of equations for the \(\tilde{u}_j\):

\[
\sum_{j=0}^{N} \tilde{u}_j L_j^{(\beta)'}(x_i) + p(x_i) \sum_{j=0}^{N} \tilde{u}_j L_j^{(\beta)''}(x_i) + q(x_i) \sum_{j=0}^{N} \tilde{u}_j L_j^{(\beta)}(x_i) = f(x_i),
\]

\[
i = 1, \ldots, N - 1,
\]

\[
\tilde{u}_0 = u_r, \quad \tilde{u}_N = u_L.
\]

In order to write this expression in a more concise way, we introduce the following vectors and matrices in \(\mathbb{R}^{N-1}\), resp. \(\mathbb{R}^{(N-1) \times (N-1)}\):

\[
\tilde{u} := [\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_{N-1}]^T,
\]

\[
D^{(1)} = (D_{ij}^{(1)}), \quad D_{ij}^{(1)} := L_j^{(\beta)'}(x_i),
\]

\[
D^{(2)} = (D_{ij}^{(2)}), \quad D_{ij}^{(2)} := L_j^{(\beta)''}(x_i),
\]

\[
P := \text{diag}(p(x_i)), \quad Q := \text{diag}(q(x_i)),
\]

\[
f := [f(x_i) - u_r(L_0^{(\beta)''}(x_i) + p(x_i)L_0^{(\beta)'}(x_i)) - u_L(L_N^{(\beta)''}(x_i) + p(x_i)L_N^{(\beta)'}(x_i))]^T,
\]

\[
i, j = 1, \ldots, N - 1.
\]

In view of \((2.2)\), the system \((2.3)\) for the unknown values \(\tilde{u}\) of the approximant then reads \(A \tilde{u} = f\), with

\[
A := D^{(2)} + PD^{(1)} + Q.
\]
Despite its large condition number for $N$ large, it can be solved very precisely by Gaussian elimination [Ber1, Tan-Tru], for only the differentiation operator $A$ is ill-conditioned, not the integration operator $A^{-1}$. The system can often also be solved efficiently via iterative methods [Ber1], although the ill-conditioned $A$ then slows down the convergence. (A conformal shift of the points can improve on this, see [Ber-Bal].) In our calculations we have alleviated the instability by using the modified Schneider-Werner formulae [Bal-Ber1, Bal]

$$D_{ij}^{(1)} = \begin{cases} \frac{\beta_j}{x_i - x_j}, & i \neq j, \\ -\sum_{k \neq i} D_{ik}^{(1)}, & i = j, \end{cases}$$

and

$$D_{ij}^{(2)} = \begin{cases} 2D_{ij}^{(1)} \left( D_{ii}^{(1)} - \frac{1}{x_i - x_j} \right), & i \neq j, \\ -\sum_{k \neq i} D_{ik}^{(2)}, & i = j, \end{cases}$$

for the differentiation matrices.

In the polynomial case ($\beta_j = w_j$, all $j$) and with the interpolation points used here, the convergence of $\tilde{u}$ toward the exact solution $u$ is exponential if $p$, $q$ and $f$ are analytic in an ellipse containing $[-1, 1]$. This can be seen through subtraction (and use of the exponential convergence of the interpolant of $f$) if $p$ and $q$ are constant, by more elaborate theorems [Can-Qua] in general cases. However, this fast convergence may show up only after too large an $N$ for practical purposes if $u$ has huge gradients (see the Introduction in [Ber-Mit2]). For error bounds through estimates of the norm of the inverse operator, see the work by Wright and collaborators, from [Cru-Wri] to [Ahn-Wri].

2.2. Optimal attachment of poles to the interpolating polynomial. It is intuitively clear, and well-known in practice, that rational interpolation can better accommodate large gradients (for literature on rational interpolation, see the catalogue [Gro]). But, at least for small numbers of nodes, the classical rational interpolation problem (interpolate a given function, here $\tilde{u}$, between $N$ points by a $r \in \mathcal{R}_{m,n}$ with $m + n = N$) is hampered by two drawbacks: the nonexistence of the solution in certain cases, which shows itself in the occurrence of “unattainable points” ([Sto], p. 56), and the possible presence of poles in the interval of interpolation, a common phenomenon for $N$ small.

To overcome these difficulties, we have suggested in [Ber-Mit2] to complement the interpolating polynomial $\tilde{u}$ with a denominator with, say, $P$ poles $z_\ell$, $\ell = 1, \ldots, P$ (and corresponding modification of the numerator so as to maintain interpolation). In practice, if the real part is to lie in the interval of interpolation, the poles will be chosen as pairs of conjugate complex numbers, so as to stay with real interpolants.

It has been noted in [Ber3] ([Ber-Mit2] contains a more obvious derivation) that introducing preassigned poles is easily achieved in the barycentric setting by multiplying the weights $w_j$ of polynomial interpolation by a multiple of

$$d_j := \prod_{\ell=1}^{P} (x_j - z_\ell), \quad j = 0, \ldots, N.$$
By writing \( r \) as

\[
r(x) := \sum_{j=0}^{N} \frac{w_j \prod_{\ell=1}^{P} \left( 1 - \frac{x_j}{x} \right)}{x - x_j} \tilde{u}_j / \sum_{j=0}^{N} \frac{w_j \prod_{\ell=1}^{P} \left( 1 - \frac{x_j}{x} \right)}{x - x_j}
\]

one sees that the polynomial is the case where all \( z_\ell \) are at infinity. The suggestion in [Ber-Mit2] is to move them from there to an optimal position where they minimize some error functional, which we take here as the norm

\[
J(z) := \| r'' + pr' + qr - f \|_{\infty}, \quad z := [z_1, \ldots, z_P]^T,
\]

of the residual of the differential equation for the approximation \( r \) with given values \( \tilde{u} \) of the solution \( u \) at the \( x_j \)'s. The optimization can only decrease the value of \( J \), since the interpolation polynomial belongs to the feasible set.

Note that the interpolated values \( \tilde{u}_j \) at the nodes are not modified as one displaces the poles: interpolation is warranted by the barycentric formula [Wer, Ber2, Ber-Mit1] — \( r'' \) and \( r''' \) do change, however, so that \( r \) does not satisfy (2.3) any longer.

Optimizing the poles \( z_\ell \) is a nonlinear problem to be solved by iteration. There is always an optimal \( z \) but, at least in special cases, there can be several of them (think of the case in which all functions arising in (1.1) are constant). Whether the optimal \( r \) is unique is an open question [Ber-Mit2]. Nevertheless, in every undetermined case among our many tests (there were very few such cases, and none for \( N \) large enough) the optimal set was a continuum and the multiplicity could easily be detected from the divergence of the optimization procedure.

3. The linear rational pseudospectral method with iteratively optimized poles. The algorithm we suggest here for solving (1.1) improves iteratively upon the polynomial pseudospectral method. It consists in recursively performing the two methods described in Section 2.

Let the \( N + 1 \) interpolation points \( x_0, \ldots, x_N \) be given, as well as the number \( P \) of poles to be optimized, which are first supposed at infinity (if no information on their final location is known at the onset). For \( k = 1, 2, \ldots, \) repeat

Step 1) compute the approximate solution \( \tilde{u}^{(k)} = [\tilde{u}_1^{(k)}, \ldots, \tilde{u}_N^{(k)}] \) of (1.1) by the linear rational collocation method with \( \beta_j = w_j d_j, d_j \) from (2.4) (\( d_j \equiv 1 \) for \( k = 1 \)). This modifies \( \tilde{u} \) (for \( k > 1 \)), but not the poles \( z \) nor the weights \( \beta \).

Step 2) for the \( \tilde{u}^{(k)} \) inherited from Step 1), optimize the location of the poles \( z \) by minimizing \( J(z) \). This changes \( \beta \), but not \( \tilde{u}^{(k)} \), and yields a new interpolant \( \tilde{u}^{(k)} \) of the latter values.

When to stop? Roughly speaking, when the decrease in \( J \) becomes too small in comparison with the cost of one more step of the algorithm.

The recurrence of Step 2) makes the algorithm costly. However, for a given problem and a given \( N \), the cost of the optimized solution is a constant multiple of the cost of the polynomial method without memory increase, whereas reaching a better precision by increasing \( N \) requires at least a quadratic increase in computing time (when using iteration methods for solving the systems of equations) and memory. Moreover, at the outcome, when \( \beta \) and \( \tilde{u} \) have been computed, evaluating \( \tilde{u} \) by the formula (2.1) is exactly as expensive as evaluating the polynomial solution. The algorithm presented here may therefore be especially interesting in cases in which the
time for finding the solution is not very relevant, but the latter must be evaluated a
great many times, as in animated graphics or in the use of \( \tilde{u} \) as the reference solution in
optimal control problems.

Notice also that in certain cases the location of the poles of \( u \) may be directly read
from the equations (equations belonging to the Fuchsian class) or approximated with
the WKB-theory [Wei]. Then the \( \beta_j \) are known, there is no need for optimization and
Step 1) yields \( \tilde{u} \); this is precisely the method introduced in [Ba-Be-Du].

4. Motivation: a corresponding Galerkin method. Why should the method work, i.e., bring improvement as compared with the classical pseudospectral solution? A corresponding Galerkin method, more complicated and computationally
more expensive, gives some indication.

Denote by \( L \) the operator which to every function \( u \) in some appropriate space \( V \)
associates the function on the left hand side of (1.1a). Then a weak form of the latter
consists in finding \( u \in V \) for which

\[
a(u, v) := (Lu, Lv) = (f, Lv) \quad \forall \ v \in V,
\]

where \( \langle \ , \ \rangle \) is the \( L_2 \)-scalar product in \( V \). (Notice that, in contrast with the classical
Galerkin method, we apply \( L \) also to the test functions \( v \).) In an appropriate space,
a(\( u, v \)) is a symmetric positive definite form which induces the energy norm

\[
\|v\|_a^2 := a(v, v) = \|Lv\|_2^2.
\]

We may now introduce the linear rational Galerkin solution of (1.1) as the function
\( \tilde{u} \in R^{(\beta)}_N \) such that

\[
a(\tilde{u}, \tilde{v}) = (f, L\tilde{v}) \quad \forall \ \tilde{v} \in R^{(\beta)}_N.
\]

\( \tilde{u} \) exists and it is unique because of the symmetry and \( V \)-ellipticity of \( a \), and it
notoriously possesses the important property of minimizing the norm \( \| \ |_a \) of the
error in \( R^{(\beta)}_N \):

\[
\|\tilde{u} - u\|_a = \min_{v \in R^{(\beta)}_N} \|v - u\|_a.
\]

(4.1)

This Galerkin method would replace Step 1) of the Algorithm to yield another \( \tilde{u}^{(k)} \).
In Step 2) we would simply change the norm in (2.5) from \( \| \ |_\infty \) to \( \| \ |_2 \) when
computing \( \tilde{u}^{(k)} \). Indeed, since \( Lu = f \), \( \|Lr - f\|_2 = \|L(r - u)\|_2 = \|r - u\|_2^2 \). And,
again, the optimal attachment of the poles can only decrease \( J \), not increase it.

Because of (4.1), the next Step 1) (with the new \( \beta \)) can in its turn only lead to a
\( \tilde{u}^{(k+1)} \) with

\[
\|\tilde{u}^{(k+1)} - u\|_a \leq \|\tilde{u}^{(k)} - u\|_a \leq \|\tilde{u}^{(k)} - u\|_a,
\]

and so on. We therefore have the following result.

Theorem. The linear rational Galerkin method with successive optimization of
the poles, as described above, yields, in the energy norm \( \| \ |_a \), a distance decreasing
sequence of approximations to the solution of (1.1).

Since the minimum property (4.1) does not hold for the collocation method, the
above result is merely an indication for the success of our algorithm. Still, in all
problems we have solved, the successive optimal attachment of the poles has resulted
in a decrease of \( J \).
It should be noted, however, that a smaller residuum does not necessarily imply an approximation \( \tilde{u} \) that is everywhere closer to the exact solution \( u \); in rare occasions, the error becomes larger at particular points when one takes too small an \( N \) (see Example 1 with \( N = 12 \) in Table 1).

5. Numerical examples. We now report on computations performed on four examples, taken from the literature for the sake of comparison: three are borrowed from [Gre], the last one from [As-Ch-Ru]. Graphs of the solutions can be found in those articles.

As mentioned in §2.1, our interpolation/collocation points have been in all examples Čebyšev points of the second kind \( x_j := \cos \frac{j\pi}{K} \), \( j = 0, \ldots, N \). We have solved the systems arising in the linear rational collocation method by means of Gaussian elimination, since for some difficult examples the simple iterative procedure of [Ber1] did not converge. The optimization of the poles in Step 2 of the algorithm has been performed as in [Ber-Mit2] by a discrete differential correction algorithm according to [Ka-Le-Ta] for small \( N \), by the simulated annealing method of [C-M-M-R] for larger numbers of nodes. The \( L_\infty \)-norm in (2.5) has been approximated by considering the values at the hundred equally spaced points

\[
\tilde{x}_k = \frac{5}{4} + \frac{k - 1.5}{K - 1.2}, \quad k = 1(1)K, \quad K = 100,
\]
on the interval \([-5/4, 5/4]\) and computing the maximal absolute value at those \( \tilde{x}_k \) lying in \([-1, 1]\). (Tests with 1000 points instead of 100 have shown that the results do not depend much on this number of points.)

The computations were performed in Fortran77 on HP-workstations.

**Example 1.** The first example is from the classical book [Sto-Bul]. Modified by the change of variable \( x = (t + 1)/2 \) to take place on the interval \([-1, 1]\), it reads

\[
2y''(t) - 200y(t) = 200 \cos^2(\pi x) + \pi^2 \cos(2\pi x),
\]
\[
y(-1) = y(1) = 0,
\]
and its exact solution is

\[
y(t) = \frac{e^{-20}}{1 + e^{-20}} e^{20x} + \frac{1}{1 + e^{-20}} e^{-20x} - \cos^2(\pi x).
\]

When applying the algorithm with an increasing number of poles \( P \), one notices that, as soon as \( P \geq 4 \), four of the poles have the tendency of arranging themselves symmetrically about the origin. This is not surprising, in view of the symmetry of the problem with respect to the imaginary axis. Since the difficulty of the optimization (the delicate part of the whole algorithm) grows sharply with the number of variables, it is natural to set the poles in groups of four and to diminish that way the number of variables from 8 to 2. The results we have obtained are summarized in Table 1. The first column gives \( N \), the number of nodes minus 1, the second the number \( P \) of optimized poles, the third the residuum norm (2.5) achieved by the optimized rational, the fourth the maximum error of the latter as above but with \( K = 1000 \) and the last the location of one of the four poles – the others being the same with the three other combinations of signs.
Table 1
Results for Example 1.

<table>
<thead>
<tr>
<th>N</th>
<th>P</th>
<th>res. norm</th>
<th>max. error</th>
<th>poles</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0</td>
<td>1.625e-1</td>
<td>8.09e-03</td>
<td>1.53893345756 + .266240698685</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>3.79e-03</td>
<td>1.287e-05</td>
<td>1.697187497475 + .31842142637906</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>6.61e-01</td>
<td>1.253e-04</td>
<td>1.9793581555787 + .55410076361411i</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1.218e-04</td>
<td>7.488e-07</td>
<td>2.19682034625 + .3486755194479</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1.307e-05</td>
<td>2.572e-06</td>
<td>2.378481979985 + .43569174491111i</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>8.482e-03</td>
<td>7.756e-07</td>
<td>2.54705789043 + .5525676279678</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>3.61e-07</td>
<td>6.600e-10</td>
<td>2.80579639418 + .51113937050601</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>8.918e-08</td>
<td>6.132e-10</td>
<td>2.46354688447 + .5553334498941</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>4.37e-05</td>
<td>2.230e-09</td>
<td>2.867253541946 + .44972407817841</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>5.56e-10</td>
<td>7.49e-13</td>
<td>2.438053136887 + .423930066798</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>3.35e-11</td>
<td>7.139e-14</td>
<td>2.46354688447 + .5553334498941</td>
</tr>
<tr>
<td>24</td>
<td>0</td>
<td>1.06e-07</td>
<td>6.334e-13</td>
<td>2.867253541946 + .44972407817841</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1.057e-11</td>
<td>2.334e-15</td>
<td>2.867253541946 + .44972407817841</td>
</tr>
</tbody>
</table>

Fig. 1. Error curves in Example 1.

As in all our tests, the residuum norm is significantly larger than the maximal error at the nodes. The results with $P = 0$ for increasing $N$ document the exponential convergence of the polynomial pseudospectral method. For every fixed $N$ the first set of four poles improves the residuum by three to five orders of magnitude and the maximal error by two to three, a very significant improvement, especially spectacular for small $N$ if one thinks in relative terms. The next four poles diminish the error
by another power of 10. Note also that the poles come to lie to the left and to the right of the interpolation interval and quite far from it, in order to best help the large gradients at the extremities.

Figure 1 displays the improvement in the error curves for \( N = 12 \) and \( N = 13 \), with \( P = 0, 4 \) and 8 poles. The error pattern is very regular; the attached poles decrease the amplitude of the oscillations, whereas the abscissae of minimal absolute error do not move much.

**Example 2.** The classical problem

\[
 u'' - eu = 0, \quad y(-1) = 1, \quad y(1) = 2,
\]

displays boundary layers at the extremities of the interval. Written in such a way as
to avoid overflow for large $\varepsilon$, the solution reads
\[
    u(x) = \frac{3}{2} \frac{e^{\delta(x-1)} + e^{-\delta(x+1)}}{1 + e^{-2\delta}} + \frac{1}{2} \frac{e^{\delta(x-1)} - e^{-\delta(x+1)}}{1 - e^{-2\delta}}, \quad \delta := \sqrt{\varepsilon}.
\]

The results, as displayed in Table 2, again show that the first four poles improve the residuum by three to almost five orders of magnitude, and that the next four add one to two more orders. Note also that, as $\varepsilon$ increases and the layers become more pronounced, the poles move closer to the extremities of the interval, as could be expected.

**Example 3.** The third example in [Gre] is chosen in such a way that the solution is very oscillatory. In more generality, the problem is
\[
    u''(x) + \frac{a}{2} u'(x) + \frac{b^2}{4} u(x) = -\frac{ab}{2} \cos(bx)e^{-ax}
\]
\[
g(0) = 0, \quad g(1) = \sin be^{-a}
\]
and its solution is given by
\[
u(x) = \sin(bx)e^{-ax},
\]
where $a$ and $b$ are two positive real parameters. $b$ controls the frequency of the oscillations: the bigger $b$, the larger the number of oscillations, and the steeper the

**Table 3**

<table>
<thead>
<tr>
<th>$b$</th>
<th>$P$</th>
<th>res. norm</th>
<th>max. error</th>
<th>poles</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>7.300e-12</td>
<td>1.518e-14</td>
<td>-3.969061491605 + 1.199190616608i</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.487e-12</td>
<td>7.147e-16</td>
<td>-3.943735599518 + 1.613384723663i</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>2.557e-09</td>
<td>5.940e-13</td>
<td>-3.429605845191 + 3.480961608887i</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2.148e-12</td>
<td>9.298e-16</td>
<td>-3.914409512838 + 2.08290328107i</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1.958e-07</td>
<td>3.597e-11</td>
<td>-2.964469353131 + 2.085228033742i</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.522e-11</td>
<td>4.958e-14</td>
<td>-1.811764925632 + 3.44094119296i</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>1.300e-03</td>
<td>3.042e-07</td>
<td>-1.723332736175 + 2.02295508355i</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>7.130e-08</td>
<td>1.910e-10</td>
<td>-5.129729688333 + 2.46167514156i</td>
</tr>
<tr>
<td>25</td>
<td>0</td>
<td>2.028e1</td>
<td>4.226e-03</td>
<td>-2.205181839708e-01 + 9.416587062625</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.390e-02</td>
<td>4.670e-05</td>
<td>-7.564105358912 + 8.763852719388i</td>
</tr>
<tr>
<td>50</td>
<td>0</td>
<td>4.483e2</td>
<td>6.021e-01</td>
<td>-7.940107554821 + 1.643196983098</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.642e1</td>
<td>4.370e-01</td>
<td>.116811887478 + .6287820805922</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>9.562e2</td>
<td>1.494e+00</td>
<td>.2857700467848 + 1.076182707013</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>9.521e1</td>
<td>1.188e+00</td>
<td>-.9531755714530 + .459079144301e-01i</td>
</tr>
</tbody>
</table>
function in each of the latter. As a consequence, the number of optimized poles should be increased in step with \( b \). This, however, is not possible, in view of the difficulty of solving optimization problems with large numbers of variables.

The same change of variable as in Example 1 must be made. In Table 3 we give the numbers obtained with \( a = 5, N = 16 \) and increasing \( b \)'s. They show that up to about \( b = 25 \) the optimal attachment of few poles yields a very significant improvement of the solution, a surprising and heartening result. With \( b = 100 \), the case considered by Greengard, the improvement is not as pronounced any more.

**Table 4**

Results for Example 4.

<table>
<thead>
<tr>
<th>( \epsilon )</th>
<th>( N )</th>
<th>( P )</th>
<th>reg. norm</th>
<th>max. error</th>
<th>poles</th>
</tr>
</thead>
<tbody>
<tr>
<td>1'000</td>
<td>128</td>
<td>0</td>
<td>( 2.299e + 00 )</td>
<td>1.990e - 05</td>
<td>( .6062920633592e - 08 + .1395306410123i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>5.904e - 03</td>
<td>7.310e - 07</td>
<td>( .2974300280300e - 01 + .1420150960183i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>2.288e - 04</td>
<td>1.110e - 07</td>
<td>( - .297426689172e - 01 + .1420150654306i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>3.603e - 05</td>
<td>3.771e - 08</td>
<td>( .4448526224808e - 01 + .145876171085i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( .5843683359294e - 01 + .1515584585260i )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( - .784563513085e - 02 + .148367955646i )</td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>0</td>
<td>9.505e - 08</td>
<td>5.107e - 15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5'000</td>
<td>128</td>
<td>0</td>
<td>( 7.592e + 03 )</td>
<td>4.804e - 02</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>5.890e + 01</td>
<td>3.106e - 03</td>
<td>( - .3094778411082e - 11 + .3545251356566e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>3.347e + 00</td>
<td>1.575e - 03</td>
<td>( - .1195181672283e - 01 + .3329382120232e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>8.637e - 01</td>
<td>4.942e - 04</td>
<td>( - .1195181706750e - 01 + .3329382120563e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( .3075858832885e - 04 + .4053163419770e - 01i )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( - .3069484691270e - 01 + .3827521508881e - 01i )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( .3071355783766e - 01 + .383201556375e - 01i )</td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>0</td>
<td>1.079e + 02</td>
<td>1.212e - 04</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10'000</td>
<td>128</td>
<td>0</td>
<td>( 3.443e + 04 )</td>
<td>1.591e - 01</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>1.695e + 02</td>
<td>8.394e - 03</td>
<td>( .3037129841257e - 08 + .2076400110120e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>7.610e + 01</td>
<td>1.573e - 02</td>
<td>( .6528495978688e - 02 + .2011661878536e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>1.057e + 00</td>
<td>3.371e - 03</td>
<td>( .6528516515182e - 02 + .2011666184988e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( .1729911682815e - 01 + .2502010850800e - 01i )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( - .488557531020e - 06 + .2517679851540e - 01i )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( - .17283873248392e - 01 + .2502158836290e - 01i )</td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>0</td>
<td>5.677e + 03</td>
<td>5.680e - 03</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>2.985e + 00</td>
<td>4.197e - 04</td>
<td>( .1440703847360e - 06 + .2947450734705e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>1.197e - 01</td>
<td>1.146e - 04</td>
<td>( .9344224596270e - 02 + .2962289496016e - 01i )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( - .9345160516667e - 02 + .296287501417e - 01i )</td>
<td></td>
</tr>
</tbody>
</table>

**Example 4.** Finally we comment on results with a problem containing a param-
eter which can make for a large slope in the interior of the interval \([\text{Hem}]\):

\[
\begin{align*}
  u''(x) + \epsilon x u'(x) &= -\pi^2 \cos(\pi x) - \epsilon \pi x \sin(\pi x), \\
  y(-1) &= -2, \quad y(1) = 0.
\end{align*}
\]

The solution

\[
u(x) = \cos \pi x + \frac{\text{erf}(\delta x)}{\text{erf}(\delta)}, \quad \delta = \sqrt{\epsilon/2},
\]

becomes steeper and steeper at zero as \(\epsilon\) grows larger. We have solved the problem for \(\epsilon = 100, 500, 1'000, 5'000\) and \(10'000\).

Some of our results with the larger \(\epsilon\) are summarized in Table 4. Since for too small an \(N\) the optimization procedure may fail to converge [Ber-Mit2], we give numbers only for \(N \geq 128\). They share some common features. For instance, for given \(\epsilon\) and \(N\), the imaginary parts of the optimal poles are quite close to one another. Moreover, if four poles are optimized, they have the tendency to gather as the vertices of a rectangle about the origin, where the maximum gradient arises.

As for the errors, the optimization improves the residuum by 4–6 digits, much more than it does with the maximum error (less than 3 digits) — see the comment on the condition in the conclusion. Nevertheless, with \(\epsilon = 5'000\) and \(N = 64\) or \(\epsilon = 10'000\) and \(N = 128\), attaching poles decreases the maximum error more than doubling \(N\).

Our results are not quite as good as those obtained in [As-Ch-Ru] with the same example. We recall, however, that they are not comparable, for our method yields \(C^\infty\)-approximations of the \(C^\infty\)-solutions of the problems considered in the present work.

Finally, we give in Figure 2 error curves for a fixed \(\epsilon-N\)-pair and an increasing number of attached poles: even with the large gradient the error behaves nicely as \(P\) increases.
6. Conclusion. In the present article we have applied to the solution of two-point boundary value problems the fact that rational interpolation is often more efficient than its polynomial counterpart. (We may mention in passing that this fact has been applied to the solution of Cauchy-type integral equations, see [Dri-Sri] and [Kai-Nod].) Our approach consists of an iterative improvement of the polynomial pseudospectral method, which is known to converge exponentially for good interpolation points and infinitely differentiable problems. After having obtained the solution at some (collocation) points by the polynomial method, we compute (one of the) rational interpolant(s) of these same values, with a denominator of given degree, by minimizing the residuum of the differential equation. This defines the new linear space of all rationals interpolating between these same points and sharing that same denominator. We then just have to start again with the solution of the original equation in the new space, and so on. Although we can prove the effectiveness of a Galerkin version of the method, in practice we solve the problem with the much simpler collocation method.

The computed examples show the somewhat surprising result that one can usually gain between 3 and 5 digits of accuracy in comparison with classical polynomial collocation, and this almost independently of \( N \). This is especially significant in cases where the precision obtained with the latter method is low and one does not want to increase the number of points so as to keep consequent evaluation of the solution as cheap as possible.

The placement of the poles is a very well-conditioned problem in the sense that many of their locations around the optimal one yield merely slightly larger residua. The tables show, however, that the gain in the residual error is usually much larger than the improvement in the precision of \( \hat{u} \). This is probably due in part to the fact that the computation of the residuum is smeared by the ill-condition of the differentiation matrices, despite the improvement by the methods in [Bal-Ber]. We hope to improve on this in a not too distant future.

Although not our purpose here, the method seems applicable also to problems whose solution \( u \) is not infinitely smooth, in fact, even when \( u \) displays discontinuities: it could then be compared with, e.g., methods which use especially constructed jump functions such as those advocated by Geer [Gee-Ban] or methods that call upon the help of several grids and fictitious points, as suggested in [Dri-For].

The generalization of the method to elliptic problems in parallelepipeds seems straightforward. In the two-dimensional case, on a tensor grid in a rectangle, the ansatz would become

\[
\hat{u}(x, y) = \sum_{j,k=0}^{N} \hat{u}_{jk} L_j^{(\beta)}(x) L_k^{(\gamma)}(y) \in \mathcal{R}_N^{(\beta)} \otimes \mathcal{R}_N^{(\gamma)}
\]

and the new approximation of \( u \) would be obtained in Step 2) of the algorithm by minimizing \( J(z, t) \) with respect to the poles \( z \) and \( t = [t_1, \ldots, t_Q] \) in

\[
\begin{align*}
\sum_{j=0}^{N} \sum_{k=0}^{M} \frac{w_j \prod_{t=1}^{P} (x_j - z_t) s_k \prod_{m=1}^{Q} (y_k - t_m)}{x - x_j} \frac{y - y_k}{y - y_k} \hat{u}_{jk} \\
\sum_{j=0}^{N} \sum_{k=0}^{M} \frac{w_j \prod_{t=1}^{P} (x_j - z_t) s_k \prod_{m=1}^{Q} (y_k - t_m)}{x - x_j} \frac{y - y_k}{y - y_k}
\end{align*}
\]
(i.e., by optimizing the weights $\beta_j = w_j \prod_{j=1}^{P} (x_j - z_j)$ and $\gamma_k = s_k \prod_{m=1}^{Q} (y_k - t_m)$), where the $s_k$'s denote the polynomial weights in the $y$-direction.

Another natural extension of the method is its application to time evolution partial differential equations: we intend to address the question in future work [Be-Mi-Tr].

Acknowledgment. The authors wish to thank the anonymous referees whose comments have improved the present work.

REFERENCES


