OPTIMAL CONTOURS FOR HIGH-ORDER DERIVATIVES

FOLKMAR BORNEMANN AND GEORG WECHSLBERGER

Abstract. As a model of more general contour integration problems we consider the numerical calculation of high-order derivatives using Cauchy’s integral formula. Bornemann (2011) showed that the condition number of the Cauchy integral strongly depends on the chosen contour and solved the problem of minimizing the condition number for circular contours. In this paper we minimize the condition number within the class of rectangular paths of step size $h$ using Provan’s algorithm for finding a shortest enclosing walk in weighted graphs embedded in the plane. Numerical examples show that optimal rectangular paths yield small condition numbers in cases where circular contours are known to be of not much use, such as for functions with branch-cut singularities.

1. Introduction

To escape from the ill-conditioning of difference schemes for the numerical calculation of high-order derivatives, numerical quadrature applied to Cauchy’s integral formula has on various occasions been suggested as a remedy (for a survey of the literature, see Bornemann 2011). To be specific, we consider a function $f$ holomorphic on a domain $D \ni 0$; Cauchy’s formula gives

$$f^{(n)}(0) = \frac{n!}{2\pi i} \int_{\Gamma} z^{-n-1} f(z) \, dz$$

for each cycle $\Gamma \subset D$ that has winding number $\text{ind}(\Gamma; 0) = 1$. If $\Gamma$ is not carefully chosen, however, the integrand tend to oscillate at a frequency of order $O(n^{-1})$ with very large amplitude (Bornemann 2011, Fig. 4). Hence, in general, there is much cancelation in the evaluation of the integral and ill-conditioning enters again. The condition number of the integral

$$\kappa(\Gamma, n) = \frac{\int_{\Gamma} |z|^{-n-1} |f(z)| \, d|z|}{\int_{\Gamma} z^{-n-1} |f(z)| \, dz}$$

and $\Gamma$ should be chosen as to make that number as small as possible. Since the denominator is, by Cauchy’s theorem, independent of $\Gamma$, we have to minimize

$$d(\Gamma) = \int_{\Gamma} |z|^{-n-1} |f(z)| \, d|z|.$$  \hspace{1cm} (1)

2010 Mathematics Subject Classification. 65E05, 65D25; 68R10, 05C38.

1Without loss of generality we evaluate derivatives at $z = 0$.

2Given an accurate and stable quadrature method, the condition number actually estimates by

# loss of significant digits $\approx \log_{10} \kappa(\Gamma, n)$

the result error, caused by round-off error in the last significant digit of the data (i.e., the function $f$).
FOLKMAR BORNEMANN AND GEORG WECHSLBERGER

Figure 1. Rectangular path of step size $h$ (taken from Lang 1999, Fig. IV.13).

Bornemann (2011) considered circular contours of radius $r$; he found that there is a unique $r = r(n)$ solving the minimization problem and the corresponding condition number $\kappa_*(n)$ satisfies (as $n \to \infty$), e.g.:

- $\kappa_*(n) \to \infty$, if $f$ is in the Hardy space $H^1$;
- $\limsup_{n \to \infty} \kappa_*(n) \leq m$, if $f$ is an entire function of completely regular growth, satisfying a non-resonance condition of the zeros, and having $m$ maxima of the Phragmén–Lindelöf indicator.

Hence, these (and similar) results solve (for all practical purposes) the problem of choosing the proper contour for entire functions, but not for functions in $H^1$. Moreover, the restriction to circles lacks an algorithmic flavor that could point to more general problems depending on the right choice of contours, such as the numerical solution of highly-oscillatory Riemann–Hilbert problems (Olver 2011).

In this paper, we solve the contour optimization problem within the more general class of rectangular paths of step size $h$ (see Fig. 1) as they are known from Artin’s proof of the general, homological version of Cauchy’s integral theorem (Lang 1999, IV.3). Such paths are composed from horizontal and vertical edges taken from a (bounded) rectangular grid $\Omega_h \subset D$ of step size $h$. Now, the weight function (1) turns the grid $\Omega_h$ into an edge-weighted graph and an optimal contour $W_*$ turns out to be a shortest enclosing walk (SEW); “enclosing” because we have to match the winding number condition $\text{ind}(W_*;0) = 1$. An efficient solution of the SEW problem for embedded graphs was found by Provan (1989), which serves as a starting point for our method.

Outline of the Paper. In Section 2 we discuss general embedded graphs in which we look for an optimal contour for the Cauchy integral; we discuss the problem of finding a shortest enclosing walk and recall Provan’s algorithm. In Section 3 we discuss some implementation details and optimizations for the problem at hand. Finally, in Section 4 we give some numerical examples and compare with the optimal circles obtained by Bornemann (2011).

2. Contour Graphs and Shortest Enclosing Walks

By generalizing the grid $\Omega_h$, we consider a finite graph $G = (V, E)$ built from vertices $V \subset D$ and edges $E$ that are smooth curves connecting the vertices within
the domain $D$. We write $uv$ for the edge connecting the vertices $u$ and $v$; by (1), its weight is given by

$$d(uv) = \int_{uv} |z|^{-(n+1)}|f(z)|\,|d|z|.$$  \hfill (2)

A walk $W$ in the graph $G$ is a closed path built from a sequence of adjacent edges, written as (where $+$ denotes joining of paths)

$$W = v_1v_2 + v_2v_3 + \cdots + v_nv_1;$$

it is called enclosing the obstacle 0 if the winding number is $\text{ind}(W;0) = 1$. The set of all possible enclosing walks is denoted by $\Pi$. As discussed in §1, the condition number is optimized by the shortest enclosing walk

$$W_s = \arg\min_{W \in \Pi} d(W)$$

where, with $W = v_1v_2 + v_2v_3 + \cdots + v_nv_1$ and $v_{m+1} = v_1$, the total weight is

$$d(W) = \sum_{j=1}^{m} d(v_jv_{j+1}).$$

The problem of finding such a SEW was solved by Provan (1989): The idea is that with $P_{u,v}$ denoting a shortest path between $u$ and $v$, any shortest enclosing walk $W_s = w_1w_2 + w_2w_3 + \cdots + w_mw_1$ can be cast in the form (Provan 1989, Theorem 1)

$$W_s = P_{w_1,w_j} + w_jw_{j+1} + P_{w_{j+1},w_1}$$

for at least one $j$. Hence, any shortest enclosing walk $W_s$ is already specified by one of its vertices and one of its edges; therefore

$$W_s \in \Pi = \{P_{u,v} + uv + P_{w,v} : u \in V, vw \in E\}.$$ 

Provan’s algorithm finds $W_s$ by, first, creating $\tilde{\Pi}$; second, removing all walks from it that do not enclose $z = 0$; and third, selecting a walk from the remaining candidates that has the lowest total weight. Using Fredman and Tarjan’s (1987) implementation of Dijkstra’s algorithm to compute the shortest paths $P_{u,v}$, the run time of the algorithm is known to be (Provan 1989, Corollary 2)

$$O(|V||E| + |V|^2 \log |V|).$$

3. Implementation Details

3.1. Edge Weight Calculation. Using the edge weights $d(uv)$ requires to approximate the integral in (2). Since not much accuracy is needed, a simple trapezoidal rule with two nodes is generally sufficient:

$$d(uv) = \int_{uv} |z|^{-(n+1)}|f(z)|\,|d|z|$$

$$\approx \frac{|u - v|}{2} (d(u) + d(v)) = \tilde{d}(uv)$$

with the vertex weight

$$d(z) = |z|^{-(n+1)}|f(z)|.$$  \hfill (3)

Although $\tilde{d}(uv)$ will typically have an accuracy of not more than just a few bits, we have not encountered a single case, in which a more accurate quadrature formula would have resulted in a different SEW $W_s$. 

OPTIMAL CONTOURS FOR HIGH-ORDER DERIVATIVES
3.2. Reducing the size of $\tilde{\Pi}$. As described in Section 2 Provan’s algorithm starts by creating a walk for every pair $(v,e) \in V \times E$ and then proceeds by selecting the best enclosing one. A simple heuristics, which works well for all our test cases, helps to considerably reduce the number of walks to be processed: Let

$$v_*= \arg\min_{v \in V} d(v)$$

and define $W_{v_*}$ as a SEW subject to the constraint

$$W_{v_*} \in \tilde{\Pi}_{v_*} = \{ \mathcal{P}_{v_*,u} + uw + \mathcal{P}_{w,v_*} : uw \in E \}.$$ 

Obviously $W_*$ and $W_{v_*}$ do not need to agree in general, as $v_*$ does not have to be traversed by $W_*$. However, since $v_*$ is the vertex with lowest weight, both walks differ mainly in a region that has no, or very minor, influence on the total weight and, consequently, also no significant influence on the condition number. Actually, $W_*$ and $W_{v_*}$ yield the exact same total weight for all the functions that we have studied to test this heuristics. Using this heuristics, the run time of Provan’s algorithm improves to

$$O(|E| + |V| \log |V|),$$

because its main part reduces to applying Dijkstra’s shortest path algorithm just once. Fig. 2 compares $W_*$ and $W_{v_*}$ for a few examples.

3.3. Size of the Grid Domain. We restrict ourselves to graphs given by finite square grids of step size $h$, centered at $z = 0$ with all vertices and edges removed that do not belong to the domain $D$. The side length $l$ of the square has to be chosen large enough to contain a SEW that approximates an optimal general integration contour. For example, if $f$ is entire, we chose $l > 2r_*$, where $r_*$ is the radius of the optimal circular contour given in Bornemann (2011). In other cases we employ a simple search for a suitable value of $l$ by calculating $W_*$ for increasing values of $l$ until $d(W_*)$ does not decrease anymore. During this search each grid uses a fixed number of vertices.

3.4. Multilevel Refinement of the SEW. Choosing a proper value for the step size $h$ is not straightforward since we would like to balance a good approximation of a generally optimal integration contour with a reasonable amount of computing time. In principle, we construct a sequence of SEWs for smaller and smaller values of $h$ until the weight of $W_*$ does not decrease anymore. To avoid an unduly amount of computational work, we do not refine the grid everywhere but use a more adaptive refinement: the grid is only refined in a neighbourhood of the currently given SEW $W_*$. Such a local refinement process is shown in Fig. 3 and works as follows:

1: calculate $W_*$ within an initial grid;
2: subdivide each rectangle adjacent to $W_*$ into 4 rectangles;
3: remove all other rectangles;
4: calculate $W_*$ in the newly created graph.

As long as the total weight of $W_*$ decreases significantly, steps 2 to 4 are repeated. It is even possible to optimize this process further by not subdividing rectangles which just contain vertices or edges of $W_*$ having weights below a certain threshold.
Figure 2. $W_\ast$ (red) vs. $W_{v\ast}$ (blue): the color coding shows the size of log $d$; with red for large values and green for small values. The thin black lines are level curves of log $d$; the smallest level shown is the threshold, below of which the edges of $W_\ast$ do not contribute to the first couple of significant digits of the total weight. All plots show that $W_\ast$ and $W_{v\ast}$ differ just in a small region well below that threshold; consequently, both walks yield about the same condition number.
4. Numerical Results

Table 1 displays conditions numbers of SEWs $W_*$ on rectangular grids as compared to the optimal circles $C_r$ for a couple of functions; Fig. 4 shows some of the corresponding contours. For entire $f$ we observe that $W_*$, like the optimal circle $C_r$, automatically traverses the saddle points of $d(z)$. It was shown in Bornemann (2011, Theorem 10.1) that, for such $f$, the major contribution of the condition number comes from these saddle points and that circles are (asymptotically, as $n \to \infty$) paths of steepest decent. Since $W_*$ can cross a saddle point only in the horizontal or vertical direction, slightly larger condition numbers had to be expected. Nevertheless, the order of magnitude is precisely matched. This match
Table 1. Condition numbers for some functions: $r_*$ are the optimal radii given in Bornemann (2011). $W_*$ was calculated in all cases (except the last one) on a $70 \times 70$-grid with $l = 3r_*$. In the last example the method of §3.3 was used to find a suitable value of $l$. The orders of differentiation $n = 2006$ and $n = 10935$ are two of the examples that give exceptionally large condition numbers for $1/\Gamma(z)$ (Bornemann 2011, Table 5).

<table>
<thead>
<tr>
<th>$f(z)$</th>
<th>$n$</th>
<th>$\kappa(W_*, n)$</th>
<th>$\kappa(C_{r_*}, n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$e^z$</td>
<td>300</td>
<td>2.0</td>
<td>1.0</td>
</tr>
<tr>
<td>$\cos z$</td>
<td>300</td>
<td>1.1</td>
<td>1.0</td>
</tr>
<tr>
<td>$e^{-z^2}$</td>
<td>300</td>
<td>1.2</td>
<td>1.0</td>
</tr>
<tr>
<td>$\text{Ai}(z)$</td>
<td>300</td>
<td>1.6</td>
<td>1.2</td>
</tr>
<tr>
<td>$1/\Gamma(z)$</td>
<td>300</td>
<td>2.2</td>
<td>1.6</td>
</tr>
<tr>
<td>$1/\Gamma(z)$</td>
<td>2006</td>
<td>$7.8 \cdot 10^4$</td>
<td>$4.7 \cdot 10^4$</td>
</tr>
<tr>
<td>$1/\Gamma(z)$</td>
<td>10935</td>
<td>$1.6 \cdot 10^5$</td>
<td>$1.4 \cdot 10^5$</td>
</tr>
<tr>
<td>$(1 - z)^{11/2}$</td>
<td>10</td>
<td>7.7</td>
<td>$5.0 \cdot 10^5$</td>
</tr>
</tbody>
</table>

holds in cases where circles give condition numbers of approximately 1, as well as in cases with exceptionally large condition numbers, such as for $f(z) = 1/\Gamma(z)$ with an order of differentiation $n = 2006$ or $n = 10935$ (cf. Bornemann 2011, §10.4).

For some non-entire $f$, however, optimal circles can be far from optimal in general: Bornemann (2011, Theorem 4.7) shows that the optimal circle $C_{r_*}$ for functions $f$ from the Hardy space $H^1$ with boundary values in $C^k, \alpha$ yields a lower condition number bound of the form

$$\kappa(C_{r_*}, n) \geq cn^{k+\alpha};$$

for instance, $f(z) = (1 - z)^{11/2}$ gives $\kappa(C_{r_*}, n) \sim 0.16059 \cdot n^{13/2}$; the principal branch of that $f$ has a branch cut at $(1, \infty)$ and $W_*$ gives significantly better condition numbers than $C_{r_*}$ by automatically following that cut.

We conclude that optimal rectangular contours are a flexible tool covering different classes of functions in a completely algorithmic fashion; no deep theory is needed to let it run (the theory is only needed to explain large condition numbers if they cannot be avoided, such as for $1/\Gamma$ and certain orders $n$ of differentiation).

References

\[ f(z) = e^z, \quad n = 300 \]

\[ f(z) = e^{-z^2}, \quad n = 300 \]

\[ f(z) = \cos(z), \quad n = 300 \]

\[ f(z) = \text{Ai}(z), \quad n = 300 \]

\[ f(z) = \frac{1}{\Gamma(z)}, \quad n = 2006 \]

\[ f(z) = (1 - z)^{11/2}, \quad n = 10 \]

**Figure 4.** \( W_\ast \) (blue) vs. \( C_\ast \) (cyan) for some of the examples of Table 1: the color coding shows the size of \( \log d \); with red for large values and green for small values. The thin black lines are level curves of \( \log d \); the smallest level shown is the threshold, below of which the edges of \( W_\ast \) do not contribute to the first couple of significant digits of the total weight.