Numerical approximation of the logarithmic capacity

Dijkstra, W.; Hochstenbach, M.E.

Published: 01/01/2008

Document Version
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the author’s version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 21. Dec. 2018
Numerical approximation of the logarithmic capacity

W. Dijkstra*, M.E. Hochstenbach

Department of Mathematics and Computing Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

Abstract

The logarithmic capacity on compact sets in \( \mathbb{R}^2 \) plays an important role in various fields of applied mathematics. Its value can be computed analytically for a few simple sets. In this paper a new algorithm is presented that numerically approximates the logarithmic capacity for more involved sets. The algorithm is based on the solution of a Dirichlet problem with the help of boundary integral equations. We illustrate the performance of the algorithm for a number of sets and observe that the logarithmic capacity is approximated accurately.

Key words: logarithmic capacity, numerical approximation, collocation, Galerkin

1 Introduction

The logarithmic capacity is a real positive number that is a function of a compact set in \( \mathbb{R}^2 \). In general it is considered as a measure for the capability of a set to support a unit amount of charge. A formal definition of the logarithmic capacity depends on the field of mathematics it is used in, as it is an important concept in several fields of applied mathematics. In potential theory the logarithmic capacity is a measure of the size of a compact set in \( \mathbb{R}^2 \) [10]. It is also often called the transfinite diameter, introduced for the first time by Fekete in the 1930s [7–9]. The transfinite diameter is a key ingredient in number theory [3]. The logarithmic capacity also appears in polynomial approximation [2,11], where it is called Chebyshev constant. It can also be directly linked to the Robin constant, which plays an essential role in the field of conformal mappings [14,15]. The latter constant appears in the Green’s function, and by approximating this Green’s function, the Robin constant, i.e. the logarithmic capacity

* Corresponding author. http://www.win.tue.nl/~hochsten

Preprint submitted to Elsevier 16 March 2008
capacity can also be approximated. The logarithmic capacity is critical in the
field of boundary integral equations, where it determines whether the integral
equation for a Dirichlet problem is singular [5,4].

Despite its frequent appearance, direct evaluation of the logarithmic capac-
ity has received little attention. The logarithmic capacity can be computed
analytically only for a few simple sets, for instance ellipses and squares. For
slightly more complex sets it can be bounded, but accurate approximations are
rare [16,17]. In this paper we propose a new algorithm to numerically approx-
imate the logarithmic capacity of compact sets that are bounded by a finite
set of smooth Jordan curves. In the next section we introduce the logarithmic
capacity from a physical point of view. A numerical algorithm to approxi-
mate the logarithmic capacity is presented in the third section. We illustrate
the performance of the algorithm in Section 4 on a number of simple sets for
which the exact value of the logarithmic capacity is known. We compare our
algorithm with the algorithms developed in [16,17].

2 Logarithmic capacity

Let Ω be a simply-connected domain in \( \mathbb{R}^2 \). Imagine that Ω is a conducting
plate and let \( \mu \geq 0 \) be a unit charge distribution on the plate. The charge
distribution \( \mu \) generates a potential \( p_\mu \), given by

\[
p_\mu(x) := \int_\Omega \log \frac{1}{\|x - y\|} \, d\mu(y),
\]

which is called the logarithmic potential. The potential energy of the charge
distribution is

\[
I(\mu) := \int_\Omega p_\mu(x) \, d\mu(x) = \int_\Omega \int_\Omega \log \frac{1}{\|x - y\|} \, d\mu(y) \, d\mu(x).
\]

The charge distributes itself such that the energy is minimized. Let \( V \) be the
minimal energy,

\[
V := \inf_\mu I(\mu),
\]

then the logarithmic capacity is defined as

\[
C(\Omega) := e^{-V}.
\]
The charge obeys Faraday’s principle and resides at the boundary of the plate. Let $\Gamma$ denote the boundary of $\Omega$. If $\Gamma$ is bounded by a finite number of smooth Jordan curves, $d\mu(x)$ is equivalent to $\mu(x)dx$ [6]. In the sequel we assume that this is the case and therefore the logarithmic potential can be written as

$$p_\mu(x) = \int_{\Gamma} \log \frac{1}{||x-y||} \mu(y) d\Gamma_y,$$

and the energy can be written as

$$I(\mu) = \int_{\Gamma} \int_{\Gamma} \log \frac{1}{||x-y||} \mu(y)\mu(y) d\Gamma_y d\Gamma_x.$$

Correspondingly we consider the logarithmic capacity as a function of $\Gamma$.

The logarithmic capacity can be computed analytically for a few simple boundaries, such as ellipses and rectangles. Table 1 lists a number of boundaries for which the logarithmic capacity is analytically known [13]. Note that $\Gamma(\cdot)$ denotes the gamma-function. For boundaries for which there is no analytic expression known, the following properties can be used to bound or to estimate the logarithmic capacity [1,12].

<table>
<thead>
<tr>
<th>boundary $\Gamma$</th>
<th>logarithmic capacity $C(\Gamma)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>circle with radius $R$</td>
<td>$R$</td>
</tr>
<tr>
<td>square with side $L$</td>
<td>$\frac{\Gamma(\frac{1}{4})^2}{4\pi^{3/2}}L \approx 0.59 \cdot L$</td>
</tr>
<tr>
<td>ellipse with semi-axes $a$ and $b$</td>
<td>$(a+b)/2$</td>
</tr>
<tr>
<td>interval of length $a$</td>
<td>$\frac{1}{4}a$</td>
</tr>
<tr>
<td>isosceles right triangle side $l$</td>
<td>$\frac{3^{3/4}\Gamma(1/4)^2}{2^{7/2}\pi^{3/2}}l \approx 0.48 \cdot l$</td>
</tr>
</tbody>
</table>

Table 1

The logarithmic capacity of some simple boundaries.

(1) If $\Gamma$ is the outer boundary of a closed bounded domain $\Omega$, then $C(\Gamma) = C(\Omega)$. This agrees with Faraday’s principle, mentioned above.

(2) Denote by $d_\Gamma$ the Euclidean diameter of $\Omega$, then $C(\Gamma) \leq d_\Gamma$. Hence the radius of the smallest circle in which $\Gamma$ is contained is an upper bound for the logarithmic capacity of $\Gamma$.

(3) If $\Gamma = x + \alpha \Gamma_1$, then $C(\Gamma) = \alpha C(\Gamma_1)$. Hence the logarithmic capacity behaves linearly with respect to scaling and is invariant with respect to translation.

(4) If $\Omega_1 \subset \Omega_2$, then $C(\Omega_1) \leq C(\Omega_2)$.

(5) For a convex domain $\Omega$,
\begin{equation}
C(\Omega) \geq \left( \frac{\text{area}(\Omega)}{\pi} \right)^{1/2}.
\end{equation}

3 Algorithm

In this section we develop a new algorithm to numerically approximate the logarithmic capacity of an arbitrary simply connected domain in \( \mathbb{R}^2 \). To this end we introduce a boundary integral operator \( \mathcal{V} : \mu \to p_\mu \),

\[(\mathcal{V}_\mu)(x) := \int_\Gamma \log \frac{1}{\|x - y\|} \mu(y) d\Gamma_y, \quad x \in \Gamma.\]

This operator is called the single layer operator. Related to this operator is the following Dirichlet problem: find a distribution \( \mu \) such that

\[
\mathcal{V}_\mu = f,
\]

where \( f \) represents Dirichlet data on the boundary \( \Gamma \). It is well known that the classical Dirichlet problem in differential form has a unique solution. For the boundary integral formulation of the Dirichlet problem this is not always the case. This can be observed with the following result [12,18].

Theorem 1 For each boundary \( \Gamma \) there exists a unique distribution \( \mu_E \) which minimizes the potential energy \( I(\mu) \) and satisfies

\[
(\mathcal{V}_\mu_E)(x) \equiv -\log C(\Gamma), \quad \text{for all } x \in \Gamma.
\]

The distribution that minimizes \( I(\mu) \) is called the equilibrium distribution. When the logarithmic capacity of \( \Gamma \) is equal to one, the right-hand side of (2) becomes zero, and the operator has a zero eigenvalue. Hence the homogeneous equation has a non-trivial solution, which can be added to the solution of the non-homogeneous Dirichlet problem. Therefore the boundary integral equation for the Dirichlet problem does not have a unique solution in this particular case.

Now we develop a strategy to find the logarithmic capacity of \( \Gamma \). Assume that the operator does not have a zero eigenvalue. This is no loss of generality, since if there is a zero eigenvalue, the logarithmic capacity is equal to one, and no further computations are required. If we rescale the boundary \( \Gamma \) by a factor \( a \), it can be verified that the corresponding single layer operator \( \mathcal{V}^a \) reads...
\[ \mathcal{V}^a \mu := a \mathcal{V} \mu - a \log a. \]

Here we made use of the fact that \( \mu \) is a unit charge distribution, i.e., has a unit contour integral. Also for the new operator \( \mathcal{V}^a \) and the rescaled boundary \( a \Gamma \) Theorem 1 holds, i.e., there exists a distribution \( \mu_E \) such that \( \mathcal{V}^a \mu_E = -\log C(a \Gamma) \). Assume that \( a \) is chosen equal to a certain critical value \( a_\ast \) such that \( \log C(a_\ast \Gamma) = 0 \). Such a value \( a_\ast \) can always be found since the logarithmic capacity behaves linearly with respect to scaling. For this particular scaling parameter \( a_\ast \), \( \mathcal{V}^a \) has a zero eigenvalue, and we find

\[ \mathcal{V}^a \mu_E = -\log a_\ast - \log C(\Gamma) = 0. \]

Hence the logarithmic capacity of the original boundary is equal to the inverse of \( a_\ast \),

\[ C(\Gamma) = \frac{1}{a_\ast}. \]

In this way the logarithmic capacity of \( \Gamma \) is related to the scaling parameter \( a_\ast \) for which the operator \( \mathcal{V}^a \) has a zero eigenvalue.

For most boundaries it is not possible to find the critical scaling parameter \( a_\ast \) for which \( \mathcal{V}^a \) has a zero eigenvalue analytically. Therefore we construct a numerical approximation of the operator, i.e., a matrix, and determine for which value \( a_\ast \) this matrix has a zero eigenvalue. To this end we discretize the boundary \( \Gamma \) and the equation (1). We choose \( N \) points \( x_k, k = 1, \ldots, N \) on the boundary \( \Gamma \). Two consecutive points \( x_k \) and \( x_{k+1} \) are connected by a straight line element \( \Gamma_k \). We assume that at each element \( \Gamma_k \) the distribution \( \mu \) is approximated by a constant value \( \mu_k \). The integral over \( \Gamma \) can be approximated by a sum of integrals over the elements \( \Gamma_k \), yielding

\[ \sum_{k=1}^{N} \mu_k \int_{\Gamma_k} \log \frac{1}{||x - y||} d\Gamma_y = f(x), \; x \in \Gamma. \tag{3} \]

There are two ways to transform the latter equation into a linear system that are commonly used in the field of boundary elements.

### 3.1 Collocation approach

The collocation approach is the most commonly used approach to discretize boundary integral equations such as (3). It yields a matrix whose elements are computed by a single evaluation of an integral. The drawback of the collocation approach is that it yields asymmetric matrices in general. We choose
collocation nodes $x^p$ at the center of each element, i.e., $x^p = (x_p + x_{p+1})/2$. We substitute $x = x^p$ in equation (3) and obtain

$$\sum_{k=1}^{N} \mu_k \int_{\Gamma_k} \frac{1}{\|x^p - y\|} d\Gamma_y = f(x^p) =: f_p, \quad p = 1, \ldots, N. \tag{4}$$

We introduce the matrix $A$ and the vectors $\mu$ and $f$ by

$$A_{pk} := \int_{\Gamma_k} \log \frac{1}{\|x^p - y\|} d\Gamma_y, \quad \mu := [\mu_1, \ldots, \mu_N]^T, \quad f := [f_1, \ldots, f_N]^T, \tag{5}$$

and write (4) in matrix-vector notation

$$A\mu = f.$$

The matrix $A$ is the discrete counterpart of the single layer operator $V$. Let $\mathbf{1}$ denote the constant vector $[1, \ldots, 1]^T$ of length $N$ and let $l$ denote the vector with the lengths of the boundary elements, $l = [|\Gamma_1|, \ldots, |\Gamma_N|]$, with $|\Gamma_k|$ the length of the $k$-th element. It can be verified that, after rescaling of the boundary by a factor $a$, the new matrix $A^a$ is given by

$$A^a = aA - a \log a \mathbf{1} l^T.$$

Hence by rescaling the domain a rank-one matrix is added to the original matrix $A$.

### 3.2 Galerkin approach

Another well-known approach to discretize boundary integral equations is the Galerkin approach. The matrices that occur in this approach are symmetric. The computation of a matrix element however, requires the evaluation of a double integral. Define the shape function $\phi_i$ by

$$\phi_i(x) = \begin{cases} 1 \text{ at } \Gamma_i, \\ 0 \text{ elsewhere,} \end{cases}$$

for $i = 1, \ldots, N$. We multiply (3) by $\phi_i$ and integrate over $\Gamma$, to get
\[
\sum_{j=1}^{N} \sum_{k=1}^{N} \mu_k \int_{\Gamma_j} \phi_i(x) \int_{\Gamma_k} \log \frac{1}{\|x - y\|} d\Gamma_y d\Gamma_x = \sum_{j=1}^{N} \int_{\Gamma_j} f(x) d\Gamma_j,
\]
for \(i = 1, \ldots, N\). Assume that \(f = f_i := f(x^i)\) at \(\Gamma_i\). As \(\phi_i\) is zero at \(\Gamma_j\) for \(j \neq i\), we obtain

\[
\sum_{k=1}^{N} \mu_k \int_{\Gamma_i} \int_{\Gamma_k} \log \frac{1}{\|x - y\|} d\Gamma_y d\Gamma_x = f_i|_{\Gamma_i}, \ i = 1, \ldots, N.
\]  

We introduce the matrix \(B\) and the vector \(g\) by

\[
B_{ik} := \int_{\Gamma_i} \int_{\Gamma_k} \log \frac{1}{\|x - y\|} d\Gamma_y d\Gamma_x \quad \text{and} \quad g := [f_1|_{\Gamma_1}, \ldots, f_N|_{\Gamma_N}]^T,
\]

and write (6) in matrix-vector form

\[
B \mu = g.
\]

Rescaling of the boundary by a factor \(a\) yields a new matrix \(B^a\), given by

\[
B^a = a^2B - a^2 \log a \ l l^T.
\]

Again, by rescaling the boundary, a rank-one matrix is added to the original matrix \(B\).

To determine the value \(a^*\) for which \(A^a\) or \(B^a\) have a zero eigenvalue, for both the collocation and Galerkin approach we realize the following. Given a matrix \(C\) and two vectors \(u\) and \(v\), the rank-one modification \(C_{\gamma,a}\) of \(C\) reads

\[
C_{\gamma,a} := \gamma C - \gamma \log a \ uv^T, \ \gamma \in \mathbb{R}.
\]

Assume for the moment that \(C\) is invertible. The determinant of the matrix \(C_{\gamma,a}\) is computed with

\[
\det(C_{\gamma,a}) = \det \left( \gamma C - \gamma \log a \ uv^T \right) = \gamma^N \det(C) \det \left( I - \log a \ C^{-1}uv^T \right) = \gamma^N \det(C) \left( 1 - \log a \ v^T C^{-1}u \right).
\]

This result implies that \(C_{\gamma,a}\) has a zero eigenvalue when
\[ a = \exp \left( \frac{1}{\sqrt{T}C^{-1}u} \right). \]

When \( C \) is not invertible, \( C^{\gamma,a} \) has a zero eigenvalue when \( a = 1 \). For the collocation approach this means that the matrix \( A^\alpha \) has a zero eigenvalue if

\[ a = a_1 := \exp \left( \frac{1}{l^TA^{-1}1} \right), \tag{8} \]

when \( A \) is invertible, and \( a = 1 \) when \( A \) is not invertible. For the Galerkin approach this means that the matrix \( B^\alpha \) has a zero eigenvalue if

\[ a = a_2 := \exp \left( \frac{1}{l^TB^{-1}l} \right), \tag{9} \]

when \( B \) is invertible, and \( a = 1 \) when \( B \) is not invertible.

In Algorithm 1 we summarize the two algorithms to approximate the logarithmic capacity.

**Algorithm 1** A boundary element method for approximating the logarithmic capacity.

**Input:** A set of nodes describing a boundary \( \Gamma \).

**Output:** An approximation of the logarithmic capacity.

1. (Collocation) Compute matrix \( A \) with (5).
2. Construct the vector \( 1 \) with ones of length \( N \), \( 1 = [1, \ldots, 1]^T \).
3. Construct the vector \( l \) containing the lengths of the boundary elements, \( l = [\|\Gamma_1\|, \ldots, \|\Gamma_N\|]^T \).
4. Compute \( C = \exp \left(-\frac{1}{l^T(A\setminus 1)}\right) \).
5. (Galerkin) Compute matrix \( B \) with (7).
6. Compute \( C = \exp \left(-\frac{1}{l^T(B\setminus l)}\right) \).

**4 Results**

We illustrate the algorithm to numerically determine the logarithmic capacity for a number of simple boundaries for which we know the exact values of the logarithmic capacity. All test are performed in Matlab on a Pentium D 3.0 GHz with 1.0 GB RAM. We compare the performance of our algorithm with the performance of the algorithm developed by Rostand in 1997 [17], which we will refer to as the Rostand algorithm. This algorithm is based on a relation between the logarithmic capacity and the Green’s function together with a
Table 2  
Error and computation time for approximating the logarithmic capacity of an ellipse with aspect ratio 3/2.

<table>
<thead>
<tr>
<th>N</th>
<th>abs. error</th>
<th>time (s)</th>
<th>abs. error</th>
<th>time (s)</th>
<th>abs. error</th>
<th>time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>$4.8 \cdot 10^{-2}$</td>
<td>0.1</td>
<td>$7.8 \cdot 10^{-2}$</td>
<td>0.3</td>
<td>$9.6 \cdot 10^{-2}$</td>
<td>0.2</td>
</tr>
<tr>
<td>20</td>
<td>$4.0 \cdot 10^{-5}$</td>
<td>0.1</td>
<td>$2.0 \cdot 10^{-2}$</td>
<td>0.3</td>
<td>$2.5 \cdot 10^{-2}$</td>
<td>0.2</td>
</tr>
<tr>
<td>30</td>
<td>$4.7 \cdot 10^{-8}$</td>
<td>0.1</td>
<td>$9.0 \cdot 10^{-3}$</td>
<td>0.4</td>
<td>$1.1 \cdot 10^{-2}$</td>
<td>0.2</td>
</tr>
<tr>
<td>40</td>
<td>$3.6 \cdot 10^{-11}$</td>
<td>0.1</td>
<td>$5.1 \cdot 10^{-3}$</td>
<td>0.5</td>
<td>$6.3 \cdot 10^{-3}$</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Table 3  
Error and computation time for approximating the logarithmic capacity of a triangle.

<table>
<thead>
<tr>
<th>N</th>
<th>abs. error</th>
<th>time (s)</th>
<th>abs. error</th>
<th>time (s)</th>
<th>abs. error</th>
<th>time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>$2.8 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$7.6 \cdot 10^{-3}$</td>
<td>0.4</td>
<td>$1.4 \cdot 10^{-2}$</td>
<td>0.3</td>
</tr>
<tr>
<td>30</td>
<td>$3.8 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$3.4 \cdot 10^{-3}$</td>
<td>0.5</td>
<td>$6.1 \cdot 10^{-3}$</td>
<td>0.3</td>
</tr>
<tr>
<td>60</td>
<td>$3.9 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$1.5 \cdot 10^{-3}$</td>
<td>0.7</td>
<td>$2.7 \cdot 10^{-3}$</td>
<td>0.4</td>
</tr>
<tr>
<td>90</td>
<td>$3.9 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$9.1 \cdot 10^{-4}$</td>
<td>1.2</td>
<td>$1.6 \cdot 10^{-3}$</td>
<td>0.6</td>
</tr>
</tbody>
</table>

uniform harmonic approximation theorem. In the algorithm the boundary is also approximated with $N$ linear elements, which allows us to get a good comparison.

The first boundary that we consider is an ellipse whose semi-axes have lengths 3 and 2. Table 1 shows that the logarithmic capacity of this boundary is equal to 2.5. As described in the previous section, we approximate the boundary of the ellipse with $N$ linear elements and compute the matrices $A$ and $B$. The critical scaling parameters $a_1$ and $a_2$ are found with (8) and (9) and the logarithmic capacity is equal to the inverse of these scaling parameters. We repeat this process for several values of $N$. In Table 2 we show the error between the approximation and the exact value of the logarithmic capacity and the cpu time.

We clearly see that the Rostand algorithm outperforms both the Galerkin and collocation algorithm. The Rostand algorithm obtains a high accuracy with a very small number of boundary elements. The Galerkin and collocation algorithm do converge however, though with a much slower rate.
The second boundary for which we approximate the logarithmic capacity is an equilateral triangle. For this boundary the logarithmic capacity can be computed analytically, and is given by

\[ C(\Gamma) = \frac{3\Gamma \left(\frac{1}{3}\right)^3}{8\pi^2} \approx 0.7305. \]

In Table 3 we show the accuracy and the cpu time for approximating the logarithmic capacity for the triangle. We observe that the Rostand algorithm does not converge to the exact value, while the Galerkin and collocation algorithm do converge. In [17] a number of modifications to the standard Rostand algorithm is introduced to obtain convergence for the triangle. Though convergence is reached, it still requires many boundary elements \((N = 600)\) to get comparable convergence to our Galerkin and BEM algorithm. It turns out that the Rostand algorithm performs perfectly for smooth boundaries, such as the ellipse, but performs poorly on boundaries that have corners, such as the triangle. Our algorithm has comparable accuracies on both smooth and non-smooth boundaries.

This trend is also observed when approximating the logarithmic capacity for the unit square. Also in this case an analytical expression for the logarithmic capacity is known, and we find

\[ C(\Gamma) = \frac{\Gamma \left(\frac{1}{4}\right)^2}{4\pi^{3/2}} \approx 0.5902. \]

Table 4 shows the accuracy and cpu time for the approximation of the logarithmic capacity for the square. Again the Rostand algorithm does not give convergence, while the Galerkin and collocation algorithm do give convergence.

For the next example we turn back to the ellipse. We keep the number of boundary elements fixed, \(N = 48\), and vary the aspect ratio of the ellipse. For each ellipse we approximate the logarithmic capacity. In Table 5 we give the error and the cpu time for these approximations. We see that the Rostand algorithm performs very well for ellipses with large aspect ratios. For slender ellipses however, the error becomes much larger. The Galerkin and collocation algorithm have the same accuracy for all ellipses. The Galerkin algorithm even seems to provide more accurate results when the aspect ratio gets smaller.

Our algorithm can also approximate the logarithmic capacity for sets that are not connected, for instance the union of several non-intersecting subsets. In Figure 1(a) we show a compact set that consists of the union of a circle and two semi-circles. This example has also been presented by Ransford and Rostand [16] and we will refer to their algorithm as the Ransford algorithm. The
Table 4
Error and computation time for approximating the logarithmic capacity of a square.

<table>
<thead>
<tr>
<th>$N$</th>
<th>Rostand [17]</th>
<th>Galerkin</th>
<th>collocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs. error</td>
<td>time (s)</td>
<td>abs. error</td>
<td>time (s)</td>
</tr>
<tr>
<td>20</td>
<td>$1.0 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$4.8 \cdot 10^{-3}$</td>
</tr>
<tr>
<td>40</td>
<td>$1.9 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$2.0 \cdot 10^{-3}$</td>
</tr>
<tr>
<td>60</td>
<td>$2.0 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$1.1 \cdot 10^{-3}$</td>
</tr>
<tr>
<td>80</td>
<td>$2.1 \cdot 10^{-2}$</td>
<td>0.2</td>
<td>$7.8 \cdot 10^{-4}$</td>
</tr>
</tbody>
</table>

Table 5
Error and computation time for approximating the logarithmic capacity of an ellipse with varying aspect ratio. The number of boundary elements is $N = 48$.

<table>
<thead>
<tr>
<th>aspect ratio</th>
<th>Rostand [17]</th>
<th>Galerkin</th>
<th>collocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs. error</td>
<td>time (s)</td>
<td>abs. error</td>
<td>time (s)</td>
</tr>
<tr>
<td>0.5</td>
<td>$3.5 \cdot 10^{-9}$</td>
<td>0.1</td>
<td>$1.1 \cdot 10^{-3}$</td>
</tr>
<tr>
<td>0.4</td>
<td>$7.5 \cdot 10^{-7}$</td>
<td>0.1</td>
<td>$9.9 \cdot 10^{-4}$</td>
</tr>
<tr>
<td>0.3</td>
<td>$6.4 \cdot 10^{-5}$</td>
<td>0.1</td>
<td>$9.2 \cdot 10^{-4}$</td>
</tr>
<tr>
<td>0.2</td>
<td>$2.0 \cdot 10^{-3}$</td>
<td>0.1</td>
<td>$8.5 \cdot 10^{-4}$</td>
</tr>
<tr>
<td>0.1</td>
<td>$8.2 \cdot 10^{-2}$</td>
<td>0.1</td>
<td>$7.7 \cdot 10^{-4}$</td>
</tr>
</tbody>
</table>

Fig. 1. The domain consists of a circle and two semi-circles. The logarithmic capacity is approximated with the collocation and Galerkin approach. The estimate of Ransford & Rostand and the corresponding error bounds are represented by horizontal lines.

algorithm determines an upper and a lower bound for the logarithmic capacity. We discretize the boundaries of the three subsets and use the Galerkin and
collocation approach to approximate the logarithmic capacity. In Figure 1(b) we plot the approximations as a function of the number of boundary elements \( N \). We observe that the approximation of the collocation approach and the approximation of the Galerkin approach converge to a value close to 2.196. The dashed horizontal lines represent an upper and lower bound for the logarithmic capacity as determined in the Ransford algorithm. We see that our approximations approach this interval.

5 Concluding remarks

We developed an algorithm to numerically approximate the logarithmic capacity of compact sets in \( \mathbb{R}^2 \). Our algorithm has approximately the same quality for sets with smooth and non-smooth boundaries, which implies that it is generally better for the latter category than the algorithm in [17]. The algorithm also works on sets that are the union of several non-overlapping subsets. Although the idea behind our algorithm is conceptually relatively simple, the results are quite encouraging. The cpu time needed to approximate the logarithmic capacity is low, and even a small number of boundary elements already produces a satisfactory accuracy.

6 Acknowledgements

We thank Dr. Timo Betcke of the University of Manchester and Prof. Dr. R.M.M. Mattheij of the Eindhoven University of Technology for helpful comments.

References


