Advances in Engineering Software 43 (2012) 96–103

Contents lists available at SciVerse ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Boundary element modeling with variable nodal and collocation point locations

T.V. Hromadka II *, D. Zillmer [1]

United States Military Academy, West Point, NY 10996, United States

ARTICLE INFO

Article history:
Received 16 August 2010
Received in revised form 12 May 2011
Accepted 30 June 2011
Available online 29 September 2011

Keywords:
Boundary elements
Complex Variable Boundary Element Method (CVBEM)
Nodal point locations
Collocation point locations
Hilbert space
Complex variables

ABSTRACT

In both the real variable and Complex Variable Boundary Element Methods (CVBEM), nodal points are typically located on the problem boundary and then various techniques are used to fit boundary condition values at the nodal point locations such as collocation (equating approximation function to boundary condition values at a discrete set of locations on the boundary) or least squares minimization on the boundary, among others. In this paper, the CVBEM is used to examine the significant improvement in approximation accuracy achieved by using as additional approximation variables the actual nodal point locations (both on the problem boundary as well as exterior of the problem domain union boundary), and to also use as additional approximation variables the locations where boundary conditions are fitted (i.e. collocation points). The developed concepts also apply directly to the more commonly used real variable boundary element technique. Our results show that significant improvement in modeling accuracy is achieved by including the nodal point coordinates and also the collocation point coordinates as additional variables to be optimized.

Published by Elsevier Ltd.

1. Introduction

The real variable Boundary Element Method (BEM) and also the Complex Variable Boundary Element Method (CVBEM) are well documented and described in the literature (for example, for the CVBEM see Hromadka and Whitley [1] and Hromadka [2], and for the BEM see Brebbia [3]), and so the reader is referred to those publications, among others, for detailed background information into these numerical methods.

In the current work, the CVBEM is focused upon assessing the advantages achieved by using a new approach towards improving approximation accuracy. This new approach, described for the CVBEM, would also apply to the well-known real variable BEM. Therefore, discussions regarding the CVBEM applications would similarly apply to the case of the BEM.

The new approach being presented is an extension of Dean and Hromadka [4]. In Dean and Hromadka [4], nodal point locations used in the CVBEM approximation function are included as variables to be optimized in minimizing approximation error in matching problem boundary conditions at collocation points that are located on the problem boundary. In other words, the general approach in boundary element methods is to fix nodal point locations on the problem boundary and then to estimate coefficients (which may be the nodal point values themselves) such that boundary conditions are better fitted by the resulting approximation function. Common to the BEM and CVBEM is the specification of these nodes to lie on the problem boundary. However, in Dean and Hromadka [4], the CVBEM node locations themselves are included as additional variables to optimize in the approximation effort in better fitting the problem boundary conditions. That is, node locations are considered variable on the problem boundary and also exterior of the problem domain union boundary.

The CVBEM is a two-dimensional and three-dimensional (and higher dimension, see Hromadka [2]) approximation technique that utilizes basis functions that exactly solve the governing partial differential equations (PDE) of the Laplace and Poisson type, among other equations. Therefore, a linear combination of such CVBEM basis functions also exactly solves such PDE, which is a property afforded by the CVBEM that is not achieved by the usual domain numerical techniques such as finite difference and finite element methods. The various coefficients used in the linear combination of CVBEM basis functions are typically determined by fitting boundary condition values under specific error reduction strategies such as collocation or least squares error minimization. The usual CVBEM basis functions involves products of complex variable polynomials with complex logarithms, which in turn depend on the defined location of the modeling nodal points. In Dean and Hromadka [4], nodal point locations were included as variables to be optimized.

In the current work, the above concept is extended by not only including the nodal point locations as variables to be optimized,

* Corresponding author. Address: Department of Mathematical Sciences, United States Military Academy, West Point, NY 10996, United States. Tel.: +1 845 938 2036; fax: +1 949 709 4632.

E-mail addresses: Ted@PHDPHDPHD.com (T.V. Hromadka II), Devon.Zillmer@GMAIL.com (D. Zillmer).

[1] Tel.: +1 870 350 6072.

0965-9978/$ - see front matter Published by Elsevier Ltd.
doi:10.1016/j.advengsoft.2011.07.003

but also by including the locations where boundary conditions are fitted (i.e. the collocation point locations) as additional variables to be optimized. It will be shown that by optimizing the boundary condition fit locations, in addition to optimizing nodal point locations, further improvement in modeling accuracy is achieved over the approach of locating nodes and collocation points arbitrarily (such as evenly space locations, for example) on the problem boundary.

As a result, significant modeling accuracy improvement is achieved with fewer nodal points and corresponding reduction in computational demand in solving problems. The CVBEM optimization algorithms were developed for use on program Mathematica, the source code of which is included as an Appendix.

2. Optimizing the CVBEM nodal point and collocation point locations

2.1. Nodal point location optimization

In this work, nodal point locations are included as variables to increase the accuracy of the CVBEM. The algorithm input required includes the number of nodes, the collocation point locations, and the problem boundary conditions.uses Collocation is used as the strategy for fitting boundary conditions.

Using the given collocation points and boundary conditions, a CVBEM approximation function is developed. The problem boundary conditions are then compared to the CVBEM approximation functions (as evaluated on the problem boundary) and used to improve the accuracy by varying node point locations, including moving nodes to be located exterior of the problem domain and boundary.

2.2. Collocation point location optimization and algorithm

In the current work, the boundary condition fitting error reduction scheme used is to equate the CVBEM approximation function to specified boundary condition values. The optimization techniques described below can extend to other boundary condition fitting schemes such as least squares error minimization, among others.

The current work's algorithm requires a set of initial collocation point locations, boundary conditions, the nodal point locations (or an additional node-optimization module can be used), and the spatial increment desired between successive tests for collocation point location movement. The final result is an optimized set of collocation point locations and their error measure. The current paper only focuses on the optimization of collocation point locations, although nodal point location optimization is contained in the Mathematica code of Appendix A.

Several different methods of calculating modeling error in matching boundary conditions were considered. Using numerical integration proved to be most efficient. First, an error function is created by taking the absolute value of the difference between boundary conditions and the approximation function evaluated on the problem boundary. This error function is then integrated along the entire problem boundary.

The bulk of the collocation point location optimization work in Appendix A is accomplished within two iterative loops: the outer loop cycles through each collocation point location, while the inner loop moves a point along the problem boundary in steps of the desired spatial increment. The movement loop sets the first collocation point location, and uses the other collocation points to generate an approximation function, then determines the error associated with using that particular point. Then the inner loop moves the first collocation point an increment (specified by the user) along the boundary and rebuilds an approximation function, determines the error, and compares it to the previous error, storing the better approximation function's collocation point location.

The above process repeats for every collocation point along the boundary until the target variable collocation point returns to its starting location, and the collocation point location that yielded the lowest measure of error is determined. The program outer loop then steps to the next collocation point, and repeats the above process for the second point, saving the collocation point location that generated the best approximation function, and so on. When each collocation point's "best" location is determined, the final distribution of collocation points is used to generate the CVBEM approximation function, and the Mathematica program (see Appendix A) outputs the collocation point locations and the associated error measure generated by that approximation function. (In this work, it is assumed that nodal point placement may occur at the rate of one node at a time, with optimization occurring accordingly. Further research is needed to assess optimization characteristics related to different schemes of placing nodal points.)

Because the CVBEM exactly solves the governing PDE within the problem domain, the modeling error is only in the matching of boundary conditions. Because the PDE considered is the Laplace or Poisson equation the maximum magnitude of modeling error occurs on the problem boundary. Therefore the modeling goal becomes developing an approximation function that matches boundary conditions.

3. Mathematica computer code

Computer program Mathematica (published by Wolfram Research) is the program that was used for the work described above. Its advantages include good quality graphics and high accuracy for complex calculations, such as complex variable mathematics. It also has several built-in functions that can make programming simpler, and some of these features were used in this work.

The main aspects of Mathematica used for this work are new function definitions, numerical integration, standard computer logic (for loops), and complex variable (not just real variable) calculations. See Appendix A for the Mathematica code.

In order to build the CVBEM approximation functions, some of the Mathematica code from Dean and Hromadka [4] is used. The current program is divided into four sections: an input and declaration section, an optional node location optimization section, a section that optimizes collocation point locations, and a plotting section that generates the relevant graphics. The inputs are found in the first section, and consist of the problem boundary conditions, initial node and collocation point locations, and the collocation point spatial increment movement desired (for collocation point movement or node location optimization trials).

4. Example problem

Consider a two-dimensional problem domain located in the first quadrant and bounded by a circle (used solely for the ease of graphical generation) of radius 1, centered at coordinate (2,2), and the complex function of $w(z) = z^2 + \frac{1}{z^2}$. The function $w(z)$ is the complex variable mathematical model describing ideal fluid flow (or other potential flow) around a unit radius cylinder in a right angle bend. (It is noted that a circular domain is selected only for convenience; irregularly-shaped domains are similarly analyzed.)

A plot of the considered ideal fluid flow problem potentials and corresponding orthogonal streamlines is shown in Fig. 1. Because the analytic solution for the presented example problem is known a priori, its complete flownet can be plotted for comparison

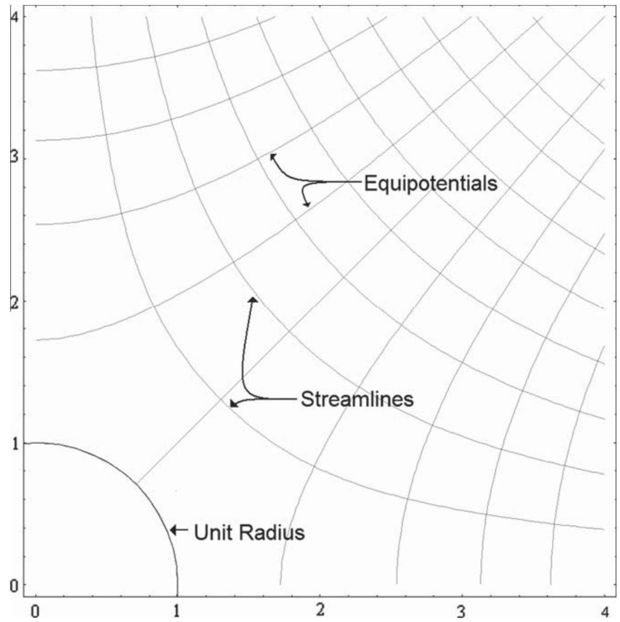

Fig. 1. Flownet corresponding to ideal fluid flow over a cylinder in right angle flow.

purposes with respect to the CVBEM approximation functions (The selected problem of ideal fluid flow is used to demonstrate the discussed procedures, while taking advantage of modeling a problem where the exact solution is known for comparison purposes.). In application, one works only with the boundary condition values known as continuous values on the problem boundary. Fig. 2 is again the flownet of the known analytic solution to the problem, but includes the problem domain and boundary used in the CVBEM analysis along with the initial set of CVBEM approximation function node and collocation point locations. In Fig. 2, the problem boundary is shown as dotted black lines, the nodal point location is shown as an "X", and collocation points are shown as small circles. (The model problem boundary is the circular region shown in Fig. 2. Boundary conditions are therefore the values from the real part of the complex function solution stated above. Boundary condition values are specified at the model collocation points also shown in Fig. 2.)

Although the provided computer program (in Mathematica, see Appendix A) includes algorithms to optimize the locations of the CVBEM nodal points (which are restricted to lie outside of the interior of the problem domain) and also the locations of the collocation points (which must be located on the problem boundary), the provided algorithms must be initialized with user-defined locations for both the nodes and collocation points. Once these various locations are initialized, the algorithms vary the node locations and the collocation point locations so as to reduce the error measure in matching boundary condition values (which is the numerical integration of the absolute value of the difference between CVBEM approximation function values on the problem boundary and the given boundary conditions). The algorithms vary the several locations one point at a time (i.e., the nodes, and once optimized, the collocation points). For example, when optimizing a particular target node location, once the error measure is optimized, the target point nodal location is held fixed and then the next nodal point is examined for its optimal location. When all node locations are optimized, the collocation point locations are optimized similarly.

In order to demonstrate the significance in optimizing node and collocation point locations as part of the approximation effort, a second CVBEM model (or "baseline" model) is built for comparison purposes, modeling the same ideal fluid flow problem presented previously. The baseline CVBEM model has a single CVBEM node and the associated five collocation points. The baseline model uses evenly spaced collocation points as shown in Fig. 2. All these various points are located on the problem boundary as is the case for both the CVBEM as well as the real variable BEM such a placement of nodes and collocation points on the problem boundary is typical of how one applies a CVBEM or real variable BEM modeling technique. Using the error measure discussed, the resulting error

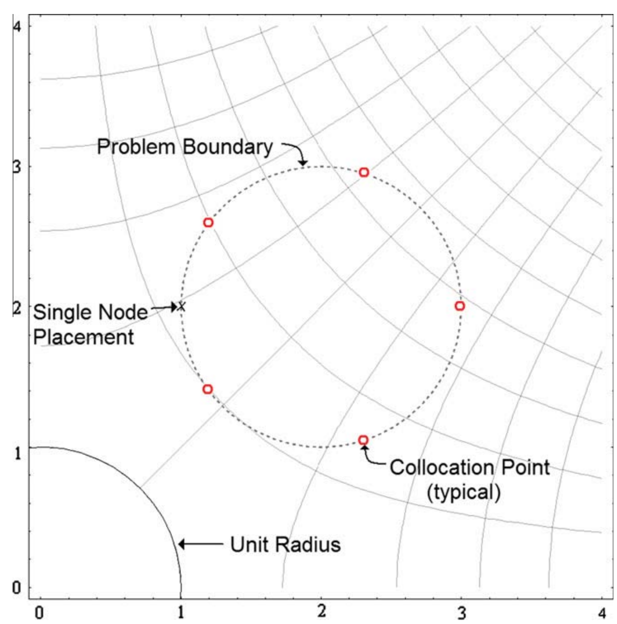

Fig. 2. Test problem domain situated in ideal fluid flow regime of Fig. 1 – diagram includes initial node and collocation point locations.

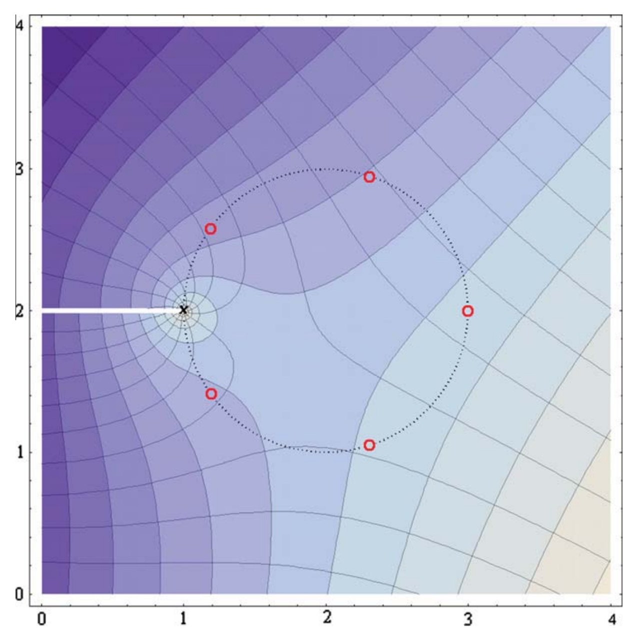

Fig. 3. Flownet from baseline CVBEM model of Fig. 2 (note branchcut emanating from nodal point).

measure for this particular baseline model is 8.30457. A plot of potentials and streamlines from this baseline model is shown in Fig. 3. Other baseline model setups are possible and other corresponding error measures would result. (It is noted that in Fig. 3, the resulting CVBEM approximation function (developed over the circular problem domain shown in Fig. 2) is defined throughout the interior and also the exterior of the circular problem domain. Therefore, the cylinder region located near the origin as shown in Fig. 1 does not apply in Fig. 3.)

For the new algorithm approach, it is recalled that the algorithm starts with optimization of the nodal point locations, and then proceeds to optimizing the corresponding collocation points.

In this example problem, using only one nodal point, the optimized location for the single node CVBEM model is shown in Fig. 4. A closer view of the resulting flownet produced from the optimized model in the vicinity of the problem domain is seen in Fig. 5. Somewhat surprisingly, the optimized location of the single CVBEM node is found to be positioned to the far upper right of the problem domain, some four diameters distant from the problem domain itself. The relevant optimized locations for collocation points are also seen in Fig. 5 and are concentrated along the northerly hemisphere of the problem boundary. The corresponding error measure for this particular single node CVBEM model is .07246. The CVBEM modeling improvement in using the optimized single node CVBEM model in comparison to the baseline model is a ratio of error measures of 8.30457/.07246 or approximately 114.6. It is noted that the above improvement in modeling accuracy is accomplished in using a single node CVBEM model (with associated five collocation points) with optimized node and collocation point locations versus a typically set up CVBEM model with one node and five approximately evenly spaced collocation points. It is also noted that by comparing the flownets of Fig. 5 (optimized single node model) to Fig. 2 (analytic solution), it is seen that a close approximation of the analytic solution is achieved on the problem domain by the presented optimization procedure.

The provided algorithm functions similarly with multiple nodes and the corresponding collocation points. Holding all other nodal point locations fixed, the program moves a single node until the optimized approximation function's error is minimized, locks that node in place, and then optimizes the next node. The program follows a similar pattern for the collocation points. In this particular problem, the two-node CVBEM model error measure is .04559.

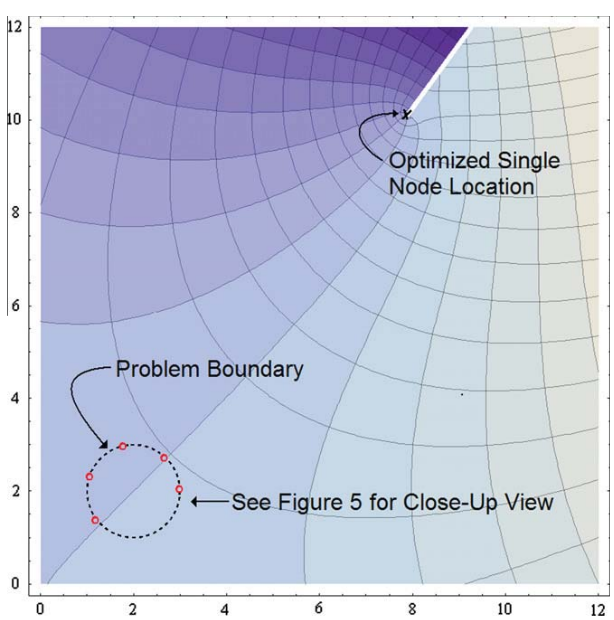

Fig. 4. Optimized node location for single-node CVBEM model of test problem.

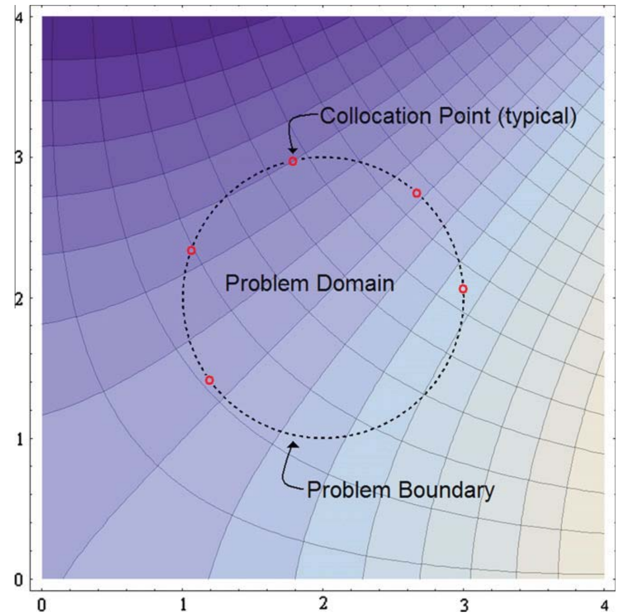

Fig. 5. Flownet from CVBEM single-node model (with optimized node location) within problem domain of Fig. 2.

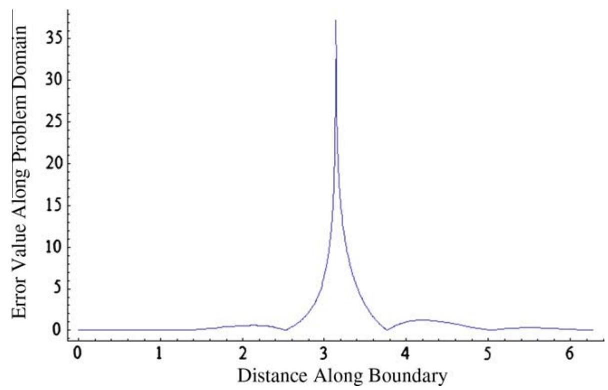

Fig. 6. CVBEM model error in matching boundary conditions for situation of Fig. 3.

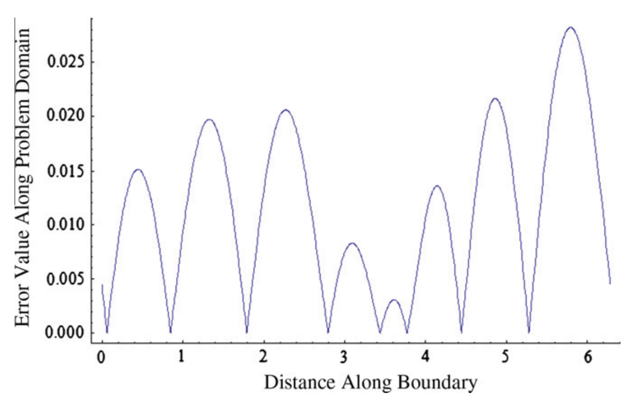

Fig. 7. CVBEM model error in matching boundary conditions for situation of Fig. 5.

Figs. 6 and 7 are plots of the error measure along the problem boundary. The Maximum Modulus Theorem (see applications in Hromadka and Whitley [1]) provides that the maximum value of the error interior of the problem domain occurs on the problem boundary. Fig. 6 corresponds to modeling error in matching boundary conditions for the case of Fig. 3; Fig. 7 corresponds to the case of Fig. 5.

5. Discussion of results

Although the example problems presented are straightforward to consider, their use aids in fully disclosing the advantages afforded by extending boundary element method mods (both CVBEM and BEM) to include both the nodal point locations and collocation point locations as additional variables to be optimized. The modeling goal would include locating nodes (both on and off the boundary) and collocation points (on the boundary) such as to reduce modeling error in matching boundary conditions. It was found that even the straightforward problems, optimizing these node and collocation point locations provides considerable improvement in modeling accuracy.

6. Conclusions

In this paper, an algorithm is presented that optimizes both the nodal point locations and also the collocation point locations used in a Complex Variable Boundary Element Method (CVBEM) model. The algorithm examines various locations for nodal points that are positioned not only on the problem boundary but also exterior of the problem domain union boundary. Starting with an initial set of five collocation point locations on the problem boundary, corresponding to a single node CVBEM model, the first considered CVBEM nodal point is used in a single node CVBEM model (which requires five collocation point for a collocation type fit to problem boundary conditions, where for a Dirichlet problem, it is assumed that the conjugate stream function has value at another collocation point), and various test locations are considered for the placement of the first node. For each test location, the error in matching boundary conditions is computed on the total problem boundary. The location for the first node, that results in the least error in matching problem boundary conditions, is then fixed as the optimized location for the first node. The algorithm proceeds to then considering adjustment of the five collocation point locations in order to further improve CVBEM approximation results. Once optimized, the algorithm holds the optimized locations for the first node and associated five collocation points as fixed. Next, the algorithm introduces the second node and an additional two collocation points. With the first node held as fixed and the first five collocation locations held fixed, the algorithm searches for the optimized locations for the second node and also the next two associated collocation points. Once optimized locations are determined, the CVBEM model has two node locations and seven collocation points locations determined. These various locations are now held fixed for the introduction of the third node and an additional two collocation points. The algorithm continues stepwise in this fashion. Computational results show significant improvement in modeling accuracy by optimizing both the nodal point and collocation point locations, where nodes are allowed to be positioned on the problem boundary or outside and away from the problem boundary (collocation points are necessarily positioned on the problem boundary). For the current paper, the described algorithm is implemented on computer program Mathematica. Source code for the subject program is provided.

Acknowledgements

The authors would like to acknowledge the Department of Mathematical Sciences at the United States Military Academy, West Point, NY, for their unending support, including but by no means limited to the Department Head COL Michael Phillips, Professor LTC Archie Wilmer III, and Professor MAJ Anthony Johnson. Acknowledgements are also due to cadets Thomas Dean, Thomas Kendall, Iain Cruickshank, Paul Falcone, and James Bailey.

References

[1] Hromadka II TV, Whitley RJ. Advances in the complex variable boundary element method. London: Spring-Verlag; 1998.
[2] Hromadka II TV. A multi-dimensional complex variable boundary element method. Southampton, England: WIT Press; 2002.
[3] Brebbia CA. The boundary element method for engineers. London: Pentech Press; 1978.
[4] Dean TR, Hromadka II TV. A collocation CVBEM using program mathematica. Eng Anal Bound Elements 2010;34(4):417–22.

Appendix A: Mathematica Optimization Code

```
(*Below are the inputs required of the user, with some defaults already in*)
(* Section 1 *)
Φ[u_, v_] := u^2 − v^2 + u^2/(u^2+v^2)^2 − u^2/(u^2+v^2)^2 ; (* This is the "solution" function *)
ζ = {{1,2}}; (* Initial Node Location(s) *)
λ = {{3,2}, {2 + Cos[(2 π)/5], 2 + Sin[(2 π)/5]}, {2 + Cos[(4 π)/5], 2 + Sin[(4 π)/5]}, {2 + Cos[(6 π)/5], 2 + Sin[(6 π)/5]}, {2 + Cos[(8 π)/5], 2 + Sin[(8 π)/5]}}; (* Initial collocation point distribution *)
α = π/20; (* Increment for each change in node location point, in radians *)
β = (2 π)/15;(* Collocation point configuration rotation increment, in radians *)
RDist = {1,1.1,1.2,1.3,1.5,2,3,4,5,10} ;(* Set of radii that will be attempted for nodal point optimization *)
δ = π/100;(* Increment for each change in collocation points, in radians *)

(* From this point on, no input or additional effort is required of the user. Below is the generation of the collocation point values (Γ) and other assorted declarations *)
n = Length[ζ];
Δθ = ConstantArray[0, n];
 Do[Δθ[[i]] = −(π + ArcTan[ζ[[i]][[1]] − 2, ζ[[i]][[1]] − 2]), {i, n}];
Γ = {};For[i = 1, i ≤ Length[λ], i + +, Γ = Join[Γ, {Φ[λ[[i]][[1]], λ[[i]][[2]]]}]];
Off[NIntegrate::slwcon];Off[Unset::norep];Off[N::meprec]; Off[NIntegrate::ncvb];
(* Node Testing *)
RNum = (2 π)/α − 1; N2Coord = e^π; N2Error = e^π π^e;
 (* Collocation Point Testing *)
MovC = (2 π)/δ; M2Error = e^π π^e;

(* Below should not have to be altered by the user. *)
(* Section 2 *)

For[NCount = 1, NCount ≤ n, NCount + +,
 Print["NODE ", NCount];
 For[rad = 1, rad ≤ Length[RDist], rad + +,
  Print["Testing a radius of ", RDist[[rad]], "..."];
  For[ang = 0, ang ≤ RNum, ang + +,
   ζ[[NCount]] = {2 + RDis[[rad]] Cos[ang α], 2 + RDist[[rad]] Sin[ang α]}//N;
   n = Length[ζ];
   Δθ[[NCount]] = −(π + ArcTan[ζ[[NCount]][[1]] − 2, ζ[[NCount]][2] − 2])//N;
   test = 0; For[i = 1, i ≤ Length[λ], i + +, If[ζ[[NCount]] == λ[[i]], test = 1]];
   For[j = 1, j ≤ (Length[ζ] − 1), j + +, If [ζ[[NCount]] == ζ [[Mod[NCount − 1 + j, Length[ζ]] + 1]] , test = 1]];
```

```
If[test == 0, b0 = 0;
φ[x_, y_] = ComplexExpand[Re[(a0 + i b0) + (a1 + i b1)(x + i y) + ∑_{j=1}^{n}(aj+1 +
i bj+1) (((x Cos[Δθ[[j]]] − y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] + x Sin[Δθ[[j]]])) −
(ζ[[j]][[1]]Cos[Δθ[[j]]] − ζ[[j]][[2]] Sin[Δθ[[j]]] +
i (ζ[[j]][[2]]Cos[Δθ[[j]]] + ζ[[j]][[1]]Sin[Δθ[[j]]])) Log[((x Cos[Δθ[[j]]] −
y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] + x Sin[Δθ[[j]]]) + i (ζ[[j]][[2]]Cos[Δθ[[j]]] +
ζ[[j]][[1]]Sin[Δθ[[j]]]))]]];
ψ[x_, y_] = ComplexExpand[Im[(a0 + i b0) + (a1 + i b1)(x + i y) + ∑_{j=1}^{n}(aj+1 +
i bj+1) (((x Cos[Δθ[[j]]] − y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] + x Sin[Δθ[[j]]])) −
(ζ[[j]][[1]]Cos[Δθ[[j]]] − ζ[[j]][[2]] Sin[Δθ[[j]]] +
i (ζ[[j]][[2]]Cos[Δθ[[j]]] + ζ[[j]][[1]]Sin[Δθ[[j]]])) Log[((x Cos[Δθ[[j]]] −
y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] + x Sin[Δθ[[j]]]) + i (ζ[[j]][[2]]Cos[Δθ[[j]]] +
ζ[[j]][[1]]Sin[Δθ[[j]]]))]]];
A = Table[φ[λ[[i]][[1]], λ[[i]][[2]], {i, 1,2 n + 3}]//N;
{k, m} = CoefficientArrays[A, Join[Table[aj, {j, 0, n + 1,1}], Table[bj{j, 1, n + 1,1}]]];
ω = LinearSolve[m, Γ]; Do[aj−1 = ω[[j]], {j, n + 2}]; Do[bj = ω[[j + n + 2]], {j, n + 1}];
ErrFun[x_, y_] := Abs[φ[x, y] − Φ[x, y]];
Error = NIntegrate[ErrFun[2 + Cos[t], 2 + Sin[t]], {t, 0,2 π}, WorkingPrecision → 10. ];
If[Error < N2Error, N2Error = Error; N2Coord = ζ[[NCount]]];
Do[aj−1 =., {j, n + 2]; Do[bj−1 =., {j, n + 2}]
Print[ζ[[NCount]], " is not a valid node location; it will be skipped. "]] (∗ IF ∗)
](* angle *)
](* radius *)
ζ[[NCount]] = N2Coord//N;
Δθ[[NCount]] = −(π + ArcTan[ζ[[NCount]][[1]] − 2, ζ[[NCount]][[1]] − 2])//N;
f[NCount == n, If[n == 1, Print["Final node location is ", ζ, ", with an error rating of ",
N2Error],  Print["Final node locations are ", ζ, ", with an error rating of ", N2Error]],
Print["New node location is ", ζ[[NCount]], ", with an error rating of ", N2Error]]; ] (*
node *)

(* Section 3 *)
Print["Beginning collocation point movement ... " ];
For[pointNo = 1, pointNo ≤ Length[λ], pointNo + +,
```

```
Print["Point ", pointNo, " ... out of ", Length[λ]];
For[rot = 0, rot ≤ (MovC − 1), rot + +,
λ[[pointNo]] = {2 + Cos[rot δ], 2 + Sin[rot δ]}//N;
Γ[[pointNo]] = Φ[λ[[pointNo]][[1]], λ[[pointNo]][[2]]] //N;
test = 0;
For[j = 1, j ≤ (Length[λ] − 1), j + +, If[λ[[pointNo]] == λ[[Mod[pointNo − 1 +
j, Length[λ]] + 1]], test = 1]];
For[i = 1, i ≤ Length[ζ], i + +, If[λ[[pointNo]] == ζ[[i]], test = 1]];
 If[test == 0, b_0 = 0;
  φ[x_, y_] = ComplexExpand[Re[(a_0 + i b_0) + (a_1 + i b_1)(x + i y) + ∑_{j=1}^{n}(a_{j+1} +
i b_{j+1}) (((x Cos[Δθ[[j]]] − y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] + x Sin[Δθ[[j]]])) −
(ζ[[j]][[1]]Cos[Δθ[[j]]] − ζ[[j]][[2]] Sin[Δθ[[j]]] + i (ζ[[j]][[2]]Cos[Δθ[[j]]] +
ζ[[j]][[1]]Sin[Δθ[[j]]])) Log[((x Cos[Δθ[[j]]] − y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] +
x Sin[Δθ[[j]]]) + i (ζ[[j]][[2]]Cos[Δθ[[j]]] + ζ[[j]][[1]]Sin[Δθ[[j]]]))]]];
  ψ[x_, y_] = ComplexExpand[Im[(a_0 + i b_0) + (a_1 + i b_1)(x + i y) + ∑_{j=1}^{n}(a_{j+1} +
i b_{j+1}) (((x Cos[Δθ[[j]]] − y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] + x Sin[Δθ[[j]]])) −
(ζ[[j]][[1]]Cos[Δθ[[j]]] − ζ[[j]][[2]] Sin[Δθ[[j]]] + i (ζ[[j]][[2]]Cos[Δθ[[j]]] +
ζ[[j]][[1]]Sin[Δθ[[j]]])) Log[((x Cos[Δθ[[j]]] − y Sin[Δθ[[j]]]) + i (y Cos[Δθ[[j]]] +
x Sin[Δθ[[j]]]) + i (ζ[[j]][[2]]Cos[Δθ[[j]]] + ζ[[j]][[1]]Sin[Δθ[[j]]]))]]];
  A = Table[φ[λ[[i]][[1]], λ[[i]][[2]], {i, 1,2 n + 3}]//N;
  {k, m} = CoefficientArrays[A, Join[Table[a_j, {j, 0, n + 1,1}], Table[b_j{j, 1, n + 1,1}]]];
  ω = LinearSolve[m, Γ]; Do[a_{j−1} = ω[[j]], {j, n + 2}]; Do[b_j = ω[[j + n + 2]], {j, n + 1}];
  ErrFun[x_, y_] ≔ Abs[φ[x, y] − Φ[x, y]];
  Error = NIntegrate[ErrFun[2 + Cos[t], 2 + Sin[t]], {t, 0,2 π}, WorkingPrecision → 10. ];
  If[Error < M2Error, M2Error = Error; MCoord = λ];
  Do[a_{j−1} =., {j, n + 2]; Do[b_{j−1} =., {j, n + 2}]] (* If *)
 ]; (* Angle *)
λ = MCoord;
Do[Γ[[i]] = Φ[λ[[i]][[1]], λ[[i]][[2]], {i, Length[λ]}]
]; (* Point 1,2,3,… *)
   Print["Optimal collocation point movement yields the points ", λ, ", with a final error
of ", M2Error, ". "];
```