LBL-22084 c.2

Lawrence Berkeley Laboratory

UNIVERSITY OF CALIFORNIA

Materials & Molecular Research Division

RECEIVED
LAWRENCE
BERKELEY LABORATORY

NOV 18 1986

LIBRARY AND
DOCUMENTS SECTION

STUDIES ON CURRENT DISTRIBUTION IN ELECTROCHEMICAL CELLS

J.O. Dukovic* and C.W. Tobias
(*Ph.D. Thesis)

August 1986

TWO-WEEK LOAN COPY
This is a Library Circulating Copy
which may be borrowed for two weeks.

LBL-22084 c.2

Prepared for the U.S. Department of Energy under Contract DE-AC03-76SF00098

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

Studies on Current Distribution in Electrochemical Cells

John Owen Dukovic

Ph.D. Thesis

Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

August 1986

The United States Department of Energy has the right to use this thesis for any purpose whatsoever including the right to reproduce all or any part thereof.

Studies on Current Distribution in Electrochemical Cells

© 1986

John Owen Dukovic

Studies on Current Distribution in Electrochemical Cells

John Owen Dukovic

ABSTRACT

Three studies of electrochemical current distribution have been performed using potential-theory models and the boundary-element method (BEM).

1. The steady-state behavior of cells with nonuniform current density over a passivating anode is investigated. Current distributions calculated for a test cell, using the measured kinetic behavior of nickel in acid-nickel-sulfate solution, are compared to estimates from earlier models. Although current-density profiles determined by weight loss on a segmented rotating cylinder agreed satisfactorily with model calculations, the measured length of the passive zone exceeds the theoretical value. The model's applicability to anodic protection is demonstrated for a stainless-steel sulfuric-acid holding tank.

2. A model is established to describe the effects of attached bubbles on the potential drop at gas-evolving electrodes including: 1) ohmic obstruction within the electrolyte; 2) area masking on the electrode surface, which raises surface overpotential; and 3) decreased local supersaturation, which lowers the concentration overpotential. The model, based on pseudosteady-state diffusion of dissolved gas within a concentration boundary layer, is applied to an example of hydrogen evolution in KOH solution. Under Tafel kinetics, the current distribution is nearly uniform over the unmasked electrode, and the increase in surface overpotential is the dominant effect. Outside the Tafel regime, the current density is strongly enhanced near the bubble, and the lowering of concentration overpotential is a major voltage effect.

3. A model of electrodeposition in the presence of diffusion-controlled leveling agents is developed. The evolution of groove profile during deposition of nickel from a Watts-type bath containing coumarin is predicted and compared to measurements reported in the literature.

To Jane and Jim, doctors of parenthood

Acknowledgments

I am grateful to many fine people who have helped me along the way:

Professor Charles Tobias has been an extraordinary man to work for. I have benefited immensely from his spirited leadership, humor, compassion, and faith in the "good stuff" that will emerge when a student is encouraged to develop in his own way.

Professor John Newman's teaching and problem-solving approach have (hopefully) rubbed off on me, and I appreciate his attention to my work. Professor Dennis Hess has given me generous guidance in exploring a new field.

I am grateful for inspiration from my earlier teachers, R. Burliegh, G. R. McMillan, J. Angus and R. Harris.

I received a lot of help from students ahead of me, Dennis Dees, Phil Russell, Mike Matlosz, Paul Sides and Geoff Prentice.

I treasure the friendship of Gina Whitney and Mark Verbrugge, who have never declined to help me with any problem — technical, personal or recreational.

It was a joy to work alongside Larry Galovich, whose excellent work, at once creative and meticulous, has been an inspiration. It will be hard to set foot in a machine shop without thinking of him.

Thanks to Mark Buehler and Brian Tobey for help with the experiments.

Finally, these people have made my time at Berkeley terrific:

Rick G, Tim O, Dawn B, Vicki E, Paul B, Rod T, John H,
Tarric E, Ron C, Grace C, Bob D, Laurie S, Marshall H,
Kathy S, Mark P, Rick H, Steve E, Ken J, Sholeh H,
Rob M, Alan H, Corinne C, Dan F, Dave K,
Philippe C, Marco K, Sandy U, Andy K,
Kay E, Christina O, Dolores N,
Brian H, Will H, Karen R,
Jennifer I, Stuart H,
and Linda.

This work was supported by the Assistant Secretary of Conservation and Renewable Energy, Office of Energy Systems Research, Energy Storage Division of the U.S. Department of Energy under Contract DE-AC03-76SF00098.

Contents

Abstract		
Dedication		i
Acknowedgment		ii
Introduction		1
Chapter 1	Current Distribution in Electrochemical Cells with Passivating Kinetics	6
	Part A. Numerical Evaluation	7
	Part B. Experimental Determination	43
Chapter 2	The Influence of Attached Bubbles on Potential Drop and Current Distribution at Gas-Evolving Electrodes	90
Chapter 3	Simulation of Leveling in Electrodeposition	146
Appendix A-1	Boundary-Element Code for Active-Passive Model	176
Appendix A-2	Boundary-Element Code for Bubble Model	236
Appendix A-3	Boundary-Element Code for Leveling Model	277
Appendix B-1	Comparison of Finite-Element and Boundary-Element Methods	327
Appendix B-2	Finite-Element Code for Active-Passive Model	335

Introduction

Each chapter of this dissertation describes a separate study of current distribution in electrochemical cells. While these pertain to different areas of electrochemical technology, and one project also involves an experimental investigation, the unifying thread of this work is the numerical solution of potential- and concentration- field problems.

The first chapter, "Current Distribution in Electrochemical Cells with Passivating Kinetics" pertains to corrosion control. Chapter 2, "The Influence of Attached Bubbles...", takes a close view of bubbles in gas-evolving electrolysis. The final chapter, "Simulation of Leveling...", deals with the evolution of surface roughness during electrodeposition. Each chapter can be viewed as an independent study, with separate abstract, symbol table, and list of references. The Fortran computer programs used in each study are listed in Appendix A.

Part of the aim of this research has been to examine the various numerical methods available for solving current-distribution problems and to compare their features. My direct predecessor on this project, Geoffrey Prentice (1), worked with the finite-difference method (FDM). I have written several codes based on the finite-element method (FEM) and used them in early for Chapters 1 and 2. However, all of the calculations reported in the body of the dissertation were performed with the boundary-element method (BEM). A relative newcomer among numerical methods, BEM is described in a number of publications, notably the monographs by Brebbia, *et al.* (2,3); a short description relevant to the present work is given Chapter 2.

A widely reaching review of numerical methods and solutions of current-distribution problems was presented by Prentice in 1982 (4). The period since this survey has been characterized by the continued use of FDM, the entrenchment of FEM,

and the introduction of BEM. Significant contributions are named below.

The finite-difference work of Prentice and Tobias is documented in three papers (5,6,7). Some earlier FDM work by Riggs, Muller and Tobias, applied to electrochemical machining, has been published (8). Menon and Landau have employed FDM in a model that incorporates diffusion, migration and unsteady-state effects in stagnant electrolytes (9).

Landolt and coworkers have continued to apply the finite-element method, simulating electrode shape change in anodic leveling (10,11,12). Alkire and Reiser have applied FEM to simulate electrodeposition on an array of parallel strips (13). A FEM program general to both electrodeposition and etching has been developed by Peskin, working with Sani (14). Finally, FEM was applied to the calculation of ohmic drop near a Luggin probe by Tokuda, *et al.* (15)

The studies using the boundary-element method have all been published since the Prentice review. Deconinck, *et al.* have applied BEM (with linear basis functions) to simulations of both electrodeposition and electrochemical machining (16). Hume, *et al.* have used BEM (with quadratic basis functions) to model electrodeposition through polymeric masks (17). In a later paper (18), the same authors have given a detailed comparison between BEM and FEM for potential problems with moving boundaries. A brief comparison of these two methods appeared earlier in a publication by Parn, *et al.* on jig design in electroplating baths (19). A three-dimensional BEM model (using constant basis functions) has been developed for cathodic protection systems with time-dependent boundary conditions (20). Finally, a boundary-integral-equation method (related to BEM) based on an iterative solution technique and promising improved computational efficiency has been presented by Cahan, *et al.* (21).

Comparison of one method to another has frequently been the subject of lively discussion. Each method has its proponents, and there are few thorough studies that consider them on an equal basis. The study by Hume, *et al.* is a rare and excellent contribution. These authors compare three formulations of the BEM to a FEM code on a moving-boundary potential problem for which the exact solution is known. For a bounded domain, the quadratic BEM was found superior to the linear BEM in efficiency and accuracy. However, the biquadratic FEM proved more efficient for a given accuracy than either. For an unbounded domain, a special formulation of the BEM was found superior.

The distinguishing characteristic of the boundary-element method is the reduction in the number of dimensions: the problem is stated and solved on the boundaries of the domain, and the advantages of BEM over the domain methods (FEM and FDM) stem from this feature. Only the boundary must be discretized (sometimes only portions of the boundary), requiring fewer nodes and resulting in a smaller matrix problem. However, since the matrix is fully populated rather than sparse, there is no significant advantage in computational efficiency. In moving-boundary problems, the repositioning of nodes is less cumbersome with the BEM. Finally, it is easier to formulate and enter the nodal representation of the problem geometry. From my experience, this advantage has been underrated by many. Although most finite-element codes provide for automatic grid generation, there are some geometric configurations for which this is not practicable; an example is given in Appendix B-1. Compared to BEM, FEM requires the extra work of specifying the coordinates of internal nodes, judiciously numbering the nodes, and specifying the element topology. This requires considerable time and attention from an analyst, and it is easy to make costly mistakes. As the cost of computing power continues to drop rapidly, this consideration could soon

overwhelm the others.

There is another, perhaps more important consideration, also related to the increasing availability of computing power. Numerical modeling is already extending to physical problems of great complexity, for example, electrochemical systems involving concentrated-solution transport or convective diffusion with complicated fluid flow. It is quite significant that boundary-element methods are restricted to a subset of linear differential equations. While BEM may occupy important niches, such as potential theory, the domain methods are likely to dominate in general practice; expertise acquired in FEM might be of greater lasting value to the scientist or engineer.

References

1. G. A. Prentice, Ph.D. Dissertation, University of California, Berkeley, Lawrence Berkeley Laboratory, LBL-11694 (December 1980).

2. C. A. Brebbia and S. Walker, "Boundary Element Techniques in Engineering," Newnes-Butterworths, Boston (1980).

3. C. A. Brebbia, "The Boundary Element Method for Engineers," Wiley & Sons, New York (1978).

4. G. A. Prentice and C. W. Tobias, *J. Electrochem. Soc.*, **129**, 72 (1982).

5. G. A. Prentice and C. W. Tobias, *AIChE Journal*, **28**, 486 (1982).

6. G. A. Prentice and C. W. Tobias, *J. Electrochem. Soc.*, **129**, 78 (1982).

7. G. A. Prentice and C. W. Tobias, *J. Electrochem. Soc.*, **129**, 316 (1982).

8. J. B. Riggs, R. H. Muller and C. W. Tobias, *Electrochimica Acta*, **26**, 961 (1981).

9. M. M. Menon and U. Landau, submitted to *J. Electrochem. Soc.*, 1986. (?)

10. R. Sautebin, H. Froidevaux, and D. Landolt, *J. Electrochem. Soc.*, **127**, 1096 (1980).

11. R. Sautebin and D. Landolt, *J. Electrochem. Soc.*, **129**, 946 (1982).

12. C. Clerc and D. Landolt, *Electrochimica Acta,* **29**, 787 (1984).

13. R. C. Alkire and D. B. Reiser, *Electrochimica Acta,* **28**, 1309 (1983).

14. A. P. Peskin, Ph.D. Dissertation, University of Colorado, Boulder, Colorado, 1985.

15. K. Tokuda, T. Gueshi, K. Aoki, and H. Matsuda, *J. Electrochem. Soc.*, **132**, 2390 (1985).

16. J. Deconinck, G. Maggetto, and J. Vereecken, *J. Electrochem. Soc.*, **132**, 2961 (1985).

17. E. C. Hume, W. M. Deen, and R. A. Brown, *J. Electrochem. Soc.*, **131**, 1251 (1984).

18. E. C. Hume, R. A. Brown, and W. M. Deen, *International Journal for Numerical Methods in Engineering,* **21**, 1295 (1985).

19. S. Y. Parn, M. McCormick, D. Howe, and J. A. Naismith, *Transactions of the Institute of Metal Finishing,* **59**, 61 (1981).

20. P. O. Gartland and R. Johnsen, Corrosion 85, Boston, Massachusetts, 1985, paper 319, National Association of Corrosion Engineers, Houston, Texas.

21. B. D. Cahan, D. Scherson, and M. A. Reid, Extended Abstracts, Electrochemical Society Meeting, October 1985, Electrochemical Society, Pennington, New Jersey.

Chapter 1

Current Distribution in Electrochemical Cells with Passivating Kinetics

This chapter is dedicated to the memory of Larry Galovich, who built the segmented-electrode apparatus for this project.

Current Distribution in Electrochemical Cells with Passivating Kinetics

A. Numerical Evaluation

ABSTRACT

A numerical model is developed for the evaluation of current distribution in electrochemical systems that exhibit passivity. The boundary-element model, based on potential theory, employs a detailed fit of the active-passive polarization curve as the anode boundary condition. The model is used to investigate the steady-state behavior of active-passive systems with nonuniform current density, in particular the condition of coexisting active and passive regions on the anode surface. Calculations are performed for a test cell of high aspect ratio using overpotential data determined on Nickel-200 in 2M H_2SO_4, 0.5M $NiSO_4$ solution. The results are compared to the predictions of simple, one-dimensional models reported in the literature. Application of the model to anodic protection is demonstrated in a series of calculations for a stainless-steel sulfuric-acid holding tank.

Introduction

Many electrochemical current-distribution problems of practical significance have been addressed in recent years as computing power has become more available and as numerical methods have improved. Surprisingly, relatively little attention has been devoted to the class of electrochemical systems involving passivating kinetics. Passivity is a critical phenomenon in many corrosion-control schemes. One technique, anodic protection, in which a structure is systematically polarized into the passive state, requires strict attention to "throwing power," the distance range that the protective passive film can be projected. Here, one may want to know the current distribution over an irregularly shaped electrode that exhibits complicated active-passive kinetic behavior.

Our aim has been to develop a numerical model for calculating current distribution that would: 1) apply to irregular, two-dimensional and axisymmetric configurations, and 2) incorporate the detailed active-passive polarization behavior of the anode.

Prior efforts at predicting active-passive current distribution can be divided into two groups. The first is a series of treatments of an iron rotating-disk electrode by Newman and various coworkers (1,2,3). In these, potential and current density at the disk surface are obtained as a global combination of Legendre polynomials (2), and a multidimensional Newton-Raphson scheme is used to iterate on the nonlinear boundary condition.

Vahdat and Newman (1) treat a freely corroding, iron rotating disk in salt water, and calculate the nonuniformity in current density that results from nonuniform transport of dissolved oxygen. The anodic overpotential neglects mass transfer, and con-

sists of a continuous analytic function representing the active-passive kinetic characteristic. Calculated current distributions include cases wherein the outer portion of the disk is passive while the inner portion is actively corroding.

Law and Newman (3) calculate the current distribution at an iron rotating-disk anode in sulfuric acid. Again the coexistence of active and passive regions is predicted. Their active-passive polarization curve consists of a discontinuous function: constant, low current density in the passive range and, in the active range, a modified Butler-Volmer expression corrected for mass-transfer limitations. In another paper (4), the same authors approach the problem of free corrosion of an iron rotating disk in oxygenated salt solution. They improve on the treatment of Vahdat and Newman with a thorough treatment of hydrodynamics and mass transfer. Again, they employ a kinetic expression that is discontinuous between active and passive states. In both treatments by Law and Newman, it was necessary to perform an external iteration on the location of the active-passive transition point.

The foregoing analyses are restricted to the relatively simple disk configuration. The treatments are somewhat more sophisticated than our own in that they include mass transfer, but in many passivating systems, especially in anodic protection, mass transfer does not play a significant role.[1]

The other arena of activity in active-passive current-distribution calculation is the field of anodic protection. An excellent and comprehensive review of the theory and

---

[1] In the case of iron in H_2SO_4 or NaCl solution studied by Newman, *et al.*, the peak active current density i_{cr} is relatively high, on the order of 1000 mA/cm^2; thus, even under strong convection, mass transfer *limits* the dissolution rate over a substantial potential range and must be accounted for in any model of value. In contrast, many passivating systems (such as 316 stainless steel in 67-percent H_2SO_4 at 25 °C with i_{cr} on the order of 1 mA/cm^2) never approach the limiting current, and the influence of mass-transfer is slight.

practice of this anticorrosion technique is given by Riggs and Locke (5). An updated overview is also available (6).

Edeleanu and Gibson (7) give a very clear discussion of the important considerations in throwing power for anodic protection. They derive a simple formula for predicting L_P^{∞}, the distance along the inside of an infinitely long[2] metal tube (or a wire inside an insulating tube) that the passive zone can be "thrown" from a cathode positioned at one end:

$$L_P^{\infty} = \frac{R\ E_P}{2\sqrt{E_A\ \bar{i}\ \rho\ r}}, \tag{1}$$

where E_A is the potential range of the active segment of the polarization curve, as illustrated in Figure 1A-1; E_P is the potential range spanned in the passive regime and depends on the applied voltage V_{APP} according to

$$E_P = V_{APP} - E_A\,; \tag{2}$$

$\bar{i}$ is the arithmetic mean current density in the active loop of the polarization curve; ρ is the electrolyte resistivity; R is the tube radius; and r is the radius of the wire ($r = R$ for a metal tube). Of particular interest is the limit E_P^{max}, the full potential range between the active and transpassive states (also shown on Figure 1A-1). This corresponds to the maximum extent of passive coverage, L_P^{max}, on the anode surface. The authors affirm Equation 2 by experiments on 18-8 stainless steel wire in 30-percent H_2SO_4. For anodes of finite length, a clear distinction is made between the distance over which one can *reestablish* passivity and the distance over which passivity can be *maintained*; the latter is acknowleged to greatly exceed the former in most cases.

Incidentally, Edeleanu and Gibson state that the transition from active to passive

[2] The superscript ∞ is our designation that the superscripted variable applies to systems of infinite as opposed to finite length.

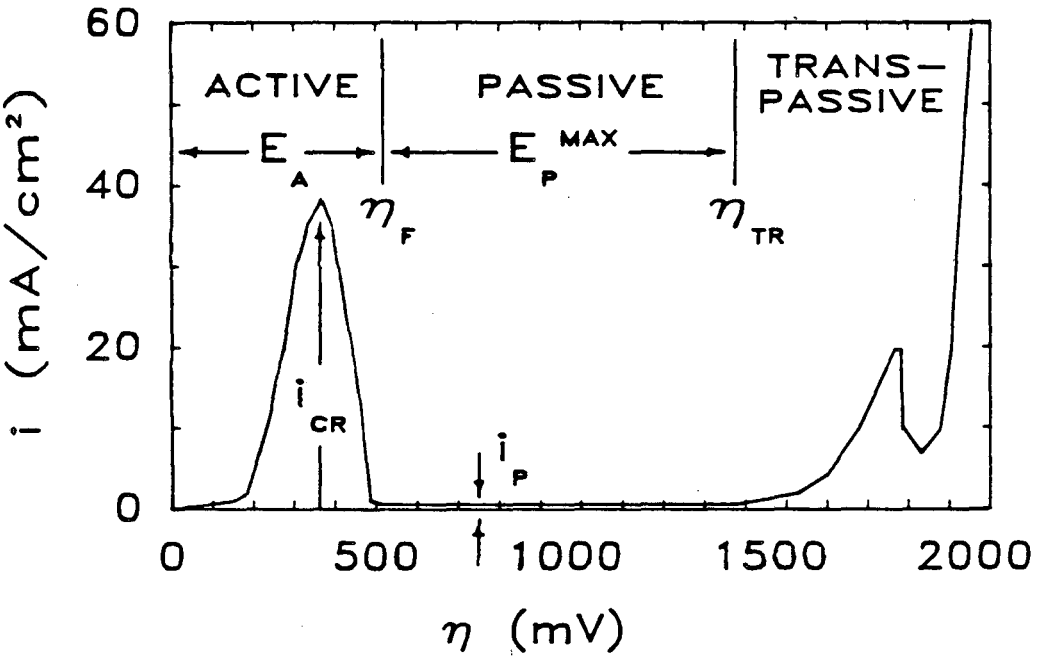


Figure 1A-1. Piecewise-linear fit of an anodic polarization curve for Nickel 200 in 2M H_2SO_4, 0.5M $NiSO_4$, illustrating potential ranges and parameter definitions.

is generally gradual with potential, and they give a mechanistic basis for this. Only systems with substantial ohmic potential drop display an abrupt transition. The misrepresentation of the iron polarization curve as a discontinuous function is perhaps a consequence of the impossibility of experimentally distinguishing ohmic drop from kinetic overpotenital for this system because active current densities are so high for iron.

Mueller (8) presents a similar but more detailed one-dimensional treatment. A simple, two-piece analytic expression approximates the polarization curve. Potential and position are related by an ordinary differential equation, which is solved analytically, producing relatively simple expressions for the passive length of a pipe of finite or infinite length. He also derives a simple expression for the length of a tube that can be maintained entirely in the passive state:

$$L_M = \left(\frac{R \; E_P^{\max}}{\rho \; i_p} \right)^{\frac{1}{2}}, \tag{3}$$

where i_p is the current density in the passive regime, assumed independent of potential. (For simplicity, we adhere to the nomenclature used by Edeleanu and Gibson.) By comparing the two cases (Eqs. 1 and 3) Mueller concludes that

$$\frac{L_M}{L_P^{\infty}} \approx \left(\frac{i_{cr}}{i_p} \right)^{\frac{1}{2}}. \tag{4}$$

As $\frac{i_{cr}}{i_p}$ often exceeds 10^4, $\frac{L_M}{L_P^{\infty}}$ can be greater than 100.

Fokin and Timonin (9), using a similar analysis, also derive an approximate length for the passive section of an infinitely long tube:

$$L_P^{\infty} = E_P \left(\frac{R}{2\, i_{cr}\, E_A\, \rho} \right)^{\frac{1}{2}}. \tag{5}$$

Their analogous expression for a cylindrical anode within an insulating annulus is

$$L_P^{\infty} = E_P \left(\frac{R^2 - r^2}{i_{cr}\, E_A\, \rho r} \right)^{\frac{1}{2}}. \tag{6}$$

These authors also derive an approximate length for the active portion of an infinitely long tube:

$$L_A^{\infty} = R \left(\frac{2E_A}{i_{cr}\, \rho\, r} \right)^{\frac{1}{2}}, \tag{7}$$

where i_{cr} is the peak current density on the polarization curve. They claim that a finite tube of length $L_P^{\infty} + L_A^{\infty}$ can be driven completely passive.

Makarov, *et al.* (10) derive similar one-dimensional formulae with minor modifications. They allow for cases in which the passive current density is not negligible (such as 1020 mild steel in 97-percent sulfuric acid at 50 °C (11) or nickel in 64-percent H_2SO_4 at 25 °C (12)). They also provide a more detailed description of the polarization curve by using a piecewise-exponential expression for current density. Both Meuller and Makarov, *et al.* state that the passive zone is longer in tubes of finite length than in infinite tubes, but no quantitative measure of this difference is given.

Timonin and Fokin (13) return to the issue of the longest tube that can be passivated directly. They experiment on wire electrodes of various lengths and confirm Eq. 5. For engineering calculations, they suggest an effective permissible length,

$$L^{ef} \leq L_P^{\infty} + \frac{1}{2} L_A^{\infty}. \tag{8}$$

The above treatments (7,8,9,10,13) reflect the evolution in understanding of distributed active-passive systems relevant to anodic protection. Compared to our model, they

rely on many simplifying assumptions; most importantly, they apply to one-dimensional rather than multidimensional cell configurations. However, they are convenient to use when applicable, the compact equations illustrate important trends, and they provide a valuable point of departure for models of greater generality.

Mathematical Model

As in most current-distribution calculations (14), our starting point is potential theory: Laplace's equation holds for the potential within the electrolyte,

$$\nabla^2\phi = 0. \tag{9}$$

This is valid when concentration gradients can be ignored so that the electrolyte conductivity κ is nearly uniform. Equation 9 follows from Ohm's law

$$i = -\kappa \nabla\phi, \tag{10}$$

and the steady-state charge balance

$$\nabla \cdot i = 0. \tag{11}$$

The domain Ω of the field problem is shown schematically in Figure 1A-2. The conditions on ϕ at the boundaries of this region are as follows. Since no current may cross an insulating wall, Γ_2, or a symmetry axis, Γ_4, there can be no normal derivative of potential at either:

$$\frac{\partial\phi}{\partial\mathbf{n}} = 0. \tag{12}$$

At the cathode surface, Γ_3, we ignore overpotential for simplicity and set the potential to the arbitrarily chosen value of zero. In current-distribution problems, the voltage of interest is the compontent of the cell voltage,

$$V = \phi_{\mathrm{M}}^{\mathrm{a}} - \phi_{\mathrm{M}}^{\mathrm{c}}, \tag{13}$$

that drives the irreversible process of current flow: we define V_{APP} as the voltage

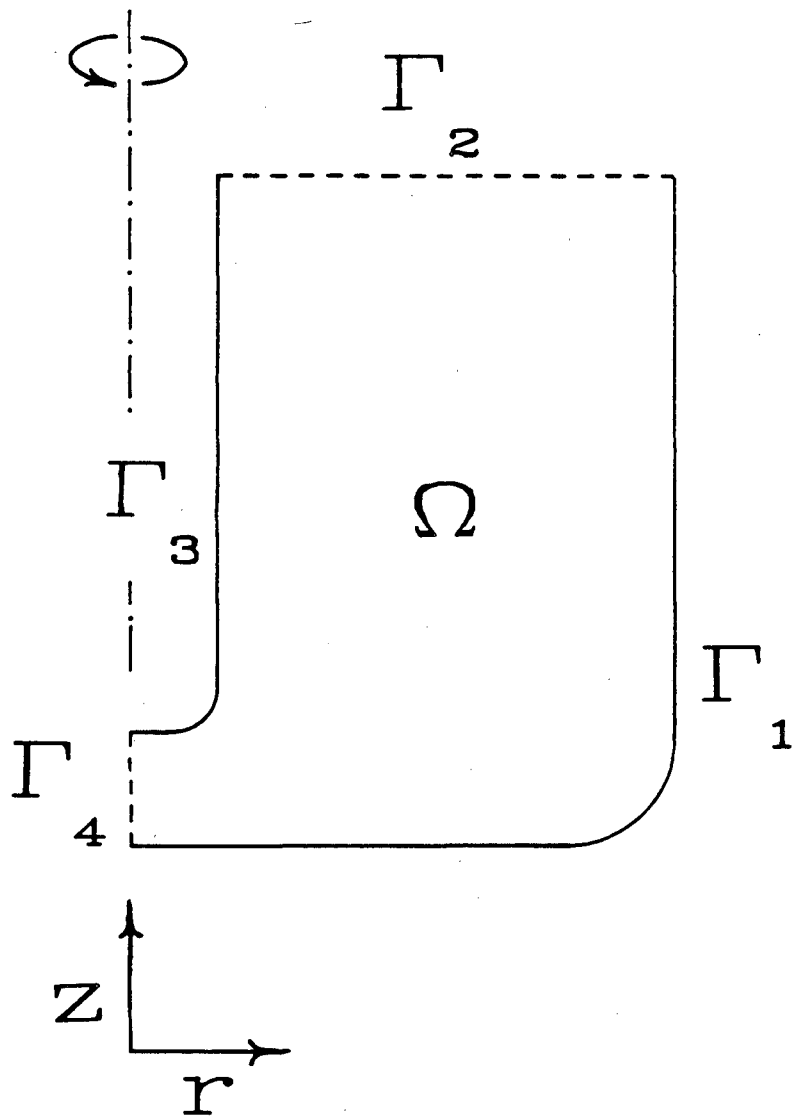


Figure 1A-2. Schematic of the domain and boundaries for the potential-field problem.

applied to the cell minus the open-circuit voltage,[3]

$$V_{APP} = V - V^{oc}. \tag{14}$$

Accordingly, we can simplify the problem by subtracting the open-circuit potential from the problem and considering the anode metal to be at the potential V_{APP} relative to the cathode.

The potential in solution adjacent to the anode surface, Γ_1, differs from ϕ_M^a by a surface overpotential, η, which depends on the local current density i:

$$\phi = V_{APP} - \eta(i). \tag{15}$$

This provides the final boundary condition for the elliptic field problem. The above boundary-value problem has been solved by numerous workers for various geometric configurations with the function $\eta(i)$ given by a linear, logarithmic, or Butler-Volmer relations. In the present study, we focus on passivating electrodes, for which overpotential and current density are related in a more complicated way. An example of this dependence is shown in Figure 1A-1, in which the regions of active dissolution, passivity, and transpassivity are labeled. Clearly, η is not a single-valued function of i, and this leads to multiple solutions and difficulty with convergence, as discussed later.

Numerical Method

We use the boundary-element method with quadratic elements. Details of our implementation are given in another communication (15). As the anode boundary condition is nonlinear, it is necessary to iterate to a solution. The starting point of the cycle is to supply an estimate of $\underline{i}$, the vector of current-density values at the

[3] Except in the case of a corrosion couple, where the open-circuit potential is not dictated purely by thermodynamics, V_{APP} can be thought of as the *irreversible* component of the cell voltage.

electrode-surface nodes. The Laplace equation is solved with this condition imposed, and the resulting overpotential profile is converted to current density according to the overpotential expression, $i(\eta)$ (*e.g.* Fig. 1A-1). (This direction for the cycle is preferred because, in the opposite direction, one would need to evaluate the function $\eta(i)$, which is not single valued. The difference between the supplied estimate of $\underline{i}$ and the newly calculated value is the residual, which is sought to vanish. The scheme we generally use for revising the starting iterate is the multivariable Newton-Raphson method. The strongly nonlinear and multivalued character of the the anodic boundary condition poses a problem with convergence. In the expamples presented in this chapter, the method often failed to converge for high values of V_{APP}. This problem was solved by recourse to *predictor-corrector continuation* (16), with V_{APP} as the continuation parameter. The scheme is to begin by solving the field problem at a low value of V_{APP}, at which convergence is easy; then V_{APP} is successively raised, giving the converged solution at each step until the desired value is reached. Euler's method is used to "predict" the starting iterate for the Newton-Raphson algorithm at each new value of V_{APP}:

$$\underline{i}(V_{APP} + \Delta V_{APP}) = \underline{i}(V_{APP}) + \Delta V_{APP} \left(\frac{\partial \underline{i}}{\partial V_{APP}} \right)_{V_{APP}} \tag{16}$$

The derivative is evaluated numerically using a small perturbation in V_{APP}. The value of ΔV_{APP} is adjusted at each continuation step according to the number of Newton-Raphson iterations used on the previous step. A "backtracking" feature guarantees that ΔV_{APP} will always be small enough to lead to a converged solution. On the approach to any "limit point" on the solution curve, where the Jacobian used in the Newton-Raphson algorithm becomes singular, ΔV_{APP} becomes successively smaller, and the continuation algorithm stalls.

As an example, the boundary-element nodal structure of Figure 1A-3B contains 113 nodes, 69 of these on the anode surface, with some refinement in the region where the active-passive transition is expected to fall. Dual nodes are employed at corners and at the flush boundary between anode and insulator (point **A** on Figure 1A-3A). As an example, the problem corresponding to the curve labeled $V_{APP} = 1200$ mV in Figure 1A-4 required 29 continuation steps from $V_{APP} = 5$ to $V_{APP} = 1200$ mV, requiring a total of 98 Newton-Raphson iterations. Thus the solution required 6860 solutions of Laplace's equation (a single forward-reduction step followed by 6860 back-substitution steps on the 113×113 boundary-element matrix), and 98 inversions of the 69×69 Jacobian matrix. The calculation required 866 c.p.u. seconds at double precision on a VAX 8600 computer.

Demonstration of the Model for a Laboratory-Scale Cell

We apply the model to the cell diagrammed in Figure 1A-3A. The electrolyte occupies an annular volume between the long anode and an insulating outer wall. The annulus is bordered at one end by an insulator and at the other by the cathode. There is a 0.64-cm section of insulator on the inner cylinder separating the two electrodes. The configuration and dimensions were chosen to be workable in the laboratory; the corresponding program of experiments is described in Part B of this chapter (22). Another feature of this cell is its high aspect ratio: since the flow of current is predominately axial, we approach the one-dimensional case studied by the aforementioned authors (7,8,9,10,13). The boundary-element nodal structure for this configuration is shown in Figure 1A-3B.

The particular passivating system we shall consider is Nickel-200 in 2M H_2SO_4 and $\frac{1}{2}$M $NiSO_4$. Figure 1A-1 is an approximation of a polarization curve for this system

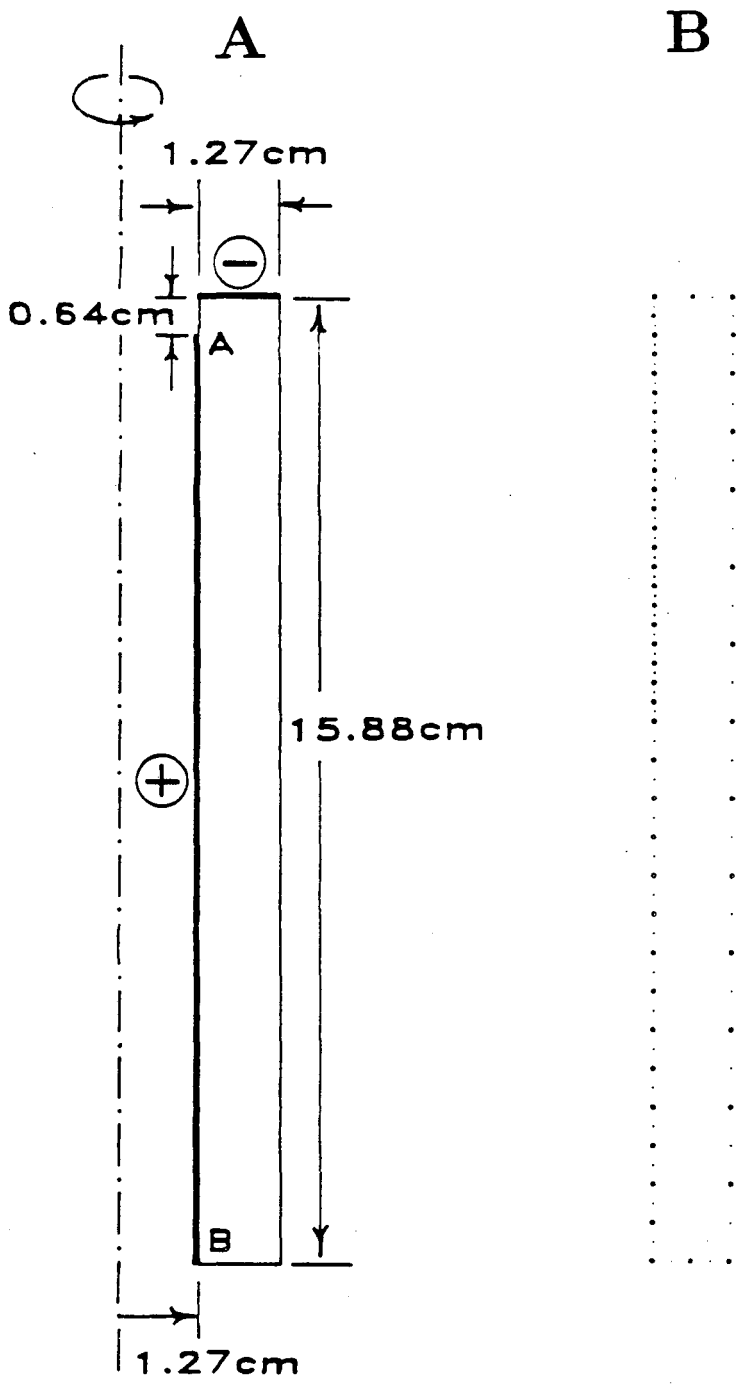


Figure 1A-3. Electrode configuration and dimensions (A), and Boundary-element nodal structure (B), of a laboratory-scale cell.

formed by joining linear segments. The polarization measurement is discribed in detail elsewhere (22). A low voltammetric scan rate was used, 8.3 millivolts per minute, in an attempt to determine the steady-state polarization behavior.[4] We have used 30 straight-line segments in this representation; greater detail could be easily achieved with no noticeable increase in computing time by fitting more segments.

The predicted current distribution is shown in Figure 1A-4 for a series of applied voltages. For V_{APP} below about 500 mV, the current density is highest at the anode edge closest to the cathode, and decreases monotonically with distance, as would be expected for a nonpassivating metal. For V_{APP} above about 600 mV, we predict that some portion of the anode near the cathode is passivated. Farther from the cathode, current density rises steeply, attains a maximum, and then declines in the manner seen at lower V_{APP}. The maximum corresponds to the peak anodic current from the overpotential curve of Figure 1A-1 and is denoted on Figure 1A-4 by the dotted line. One can consider that we have effectively "mapped" the $i-\eta$ characteristic onto the anode surface. Another view is the following: in Figure 1A-1, the polarization behavior of the anode is resolved *temporally*, *i.e.* determined by recording one $i-\eta$ pair at a time on a microelectrode; in Figure 1A-4, the i-η characteristic is resolved *spatially* over a macroelectrode.

As the V_{APP} is raised (*e.g.* 900 through 1688 mV), the length L_P of the passive zone increases, while the active portion of the current-density profile approximately

[4] It is questionable whether voltammetry on the time scale of hours can truly determine steady-state active-passive behavior. Current densities at fixed potential have been observed to drift downward for weeks after the establishment of passivity (5). Changes in the condition of the anode surface over such an interval can be significant and difficult to characterize. However, for obvious reasons, the passive portion of the polarization curve has far less impact on the current distribution than does the active portion. Transients in the active range are shorter-lived: in studies of the dependence of polarization data on potentiodynamic scan rate, on nickel in 1N H_2SO_4 (17) and on two steels in sufuric acid (18), i_{cr} was found to be roughly independent of scan rate below 30 mV/min.

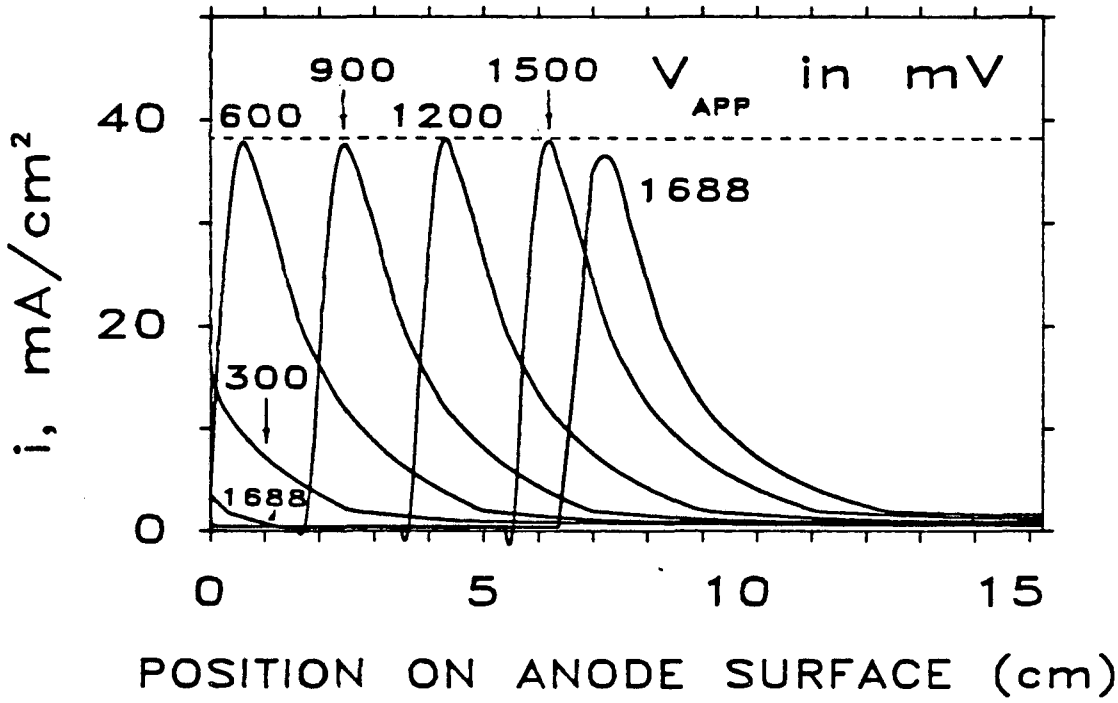


Figure 1A-4. Current density profiles predicted by the model for the laboratory-scale cell, for several values of the applied potential V_{APP}.

retains its shape. As explained earlier, the computational method progresses to higher values of V_{APP} using predictor-corrector continuation. The curve labeled $V_{APP} = 1688$ marks the point in this progression at which the continuation algorithm stalls; if V_{APP} is raised beyond this, the Newton-Raphson algorithm will diverge. However, if one changes the iteration procedure by conducting a series of relaxed-substitution steps (using a relaxation factor of 0.01 for 20 iterations) before the Newton-Raphson iteration, the system does converge beyond this limit point. The resulting solution is the condition of complete passivity. A reasonable physical interpretation is that a partially active condition cannot be maintained for V_{APP} higher than 1688 mV, and there is a transition to the fully passive condition.

Interestingly, although not contrary to expectation, one can use continuation to proceed downward in V_{APP} from this point, remaining in the regime of complete passivity, until another limit point is reached. As the passive current density is small so that the ohmic drop is slight, this limiting value, $V_{APP} = 637$ mV, is near the Flade potential, $\eta_F = 517$ mV.

Actually, in this example, for V_{APP} slightly exceeding 1688 mV, the anode is not entirely passive because the edge near the cathode displays the initial stage of transpassivity (see Fig. 1A-4). With further increases in V_{APP}, as shown in Figure 1A-5, the current distribution is described by a series of curves similar to those seen at low V_{APP} (*e.g.* the curve for $V_{APP} = 300$ mV in Fig. 1A-4) However, now the distribution is complicated slightly by the secondary-passivation effect (the minimum in the curve for $V_{APP} = 2000$ mV.)

The dotted line represents the peak anodic current from the polarization curve used in the model. If the numerical implementation of the model were perfect, the

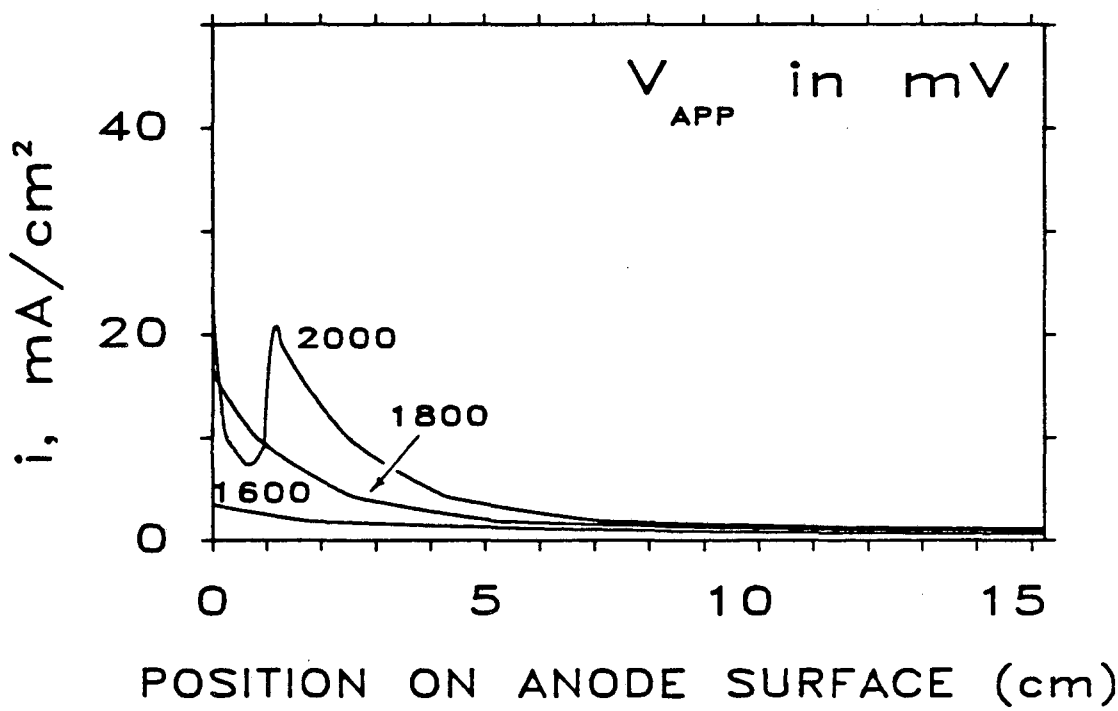


Figure 1A-5. Current density profiles reflecting transpassive dissolution, predicted by the model for the laboratory-scale cell at at high values of the applied potential V_{APP}.

peaks of the curves labeled 600, 900, 1200, 1500 and 1688 mV would all meet this line. Another manifestation of the imperfection of the numerical method, is the slight excursion below zero current density shown on some curves at the active-passive transition. Both artifacts are consequences of the limited ability of the quadratic basis function to represent the sharp variations as at current-density maxima and at the abrupt active-passive transition. The impact of these local imperfections on overall accuracy is probably quite small.

A better understanding of the behavior of a distributed active-passive system can be gained by plotting an aggregate property of the solution, *e.g.* the total current, I, as a function of the applied potential, V_{APP}. This is done for the present system in Figure 1A-6. The solid portions of the curve contain I – V_{APP} pairs representing converged solutions. The dotted portion is drawn arbitrarily to suggest how the two "stable branches" are likely to be connected. A method exists to solve for this dotted curve (19). We did not implement this technique, partly because this "unstable branch" of the solution curve is not of practical importance.[5] Based on the mathematical description, prior reported studies (20,21), and direct experimental investigations (22), our understanding is that, as V_{APP} is slowly raised, a real system will jump to complete passivity (downward arrow, Figure 1A-6) and on subsequent lowering of V_{APP}, the system will jump to partial (or total) activity (upward arrow, Figure 1A-6).

One parameter characterizing the current density profiles of Figure 1A-4 is L_P, defined earlier as the distance from the anode edge over which the passive region extends. This is of primary concern in anodic protection and is closely related to the

[5] It may be possible, however, to obtain portions of the unstable branch experimentally using the technique of Epelboin *et al.*(20), further demonstrated by Russell and Newman (21).

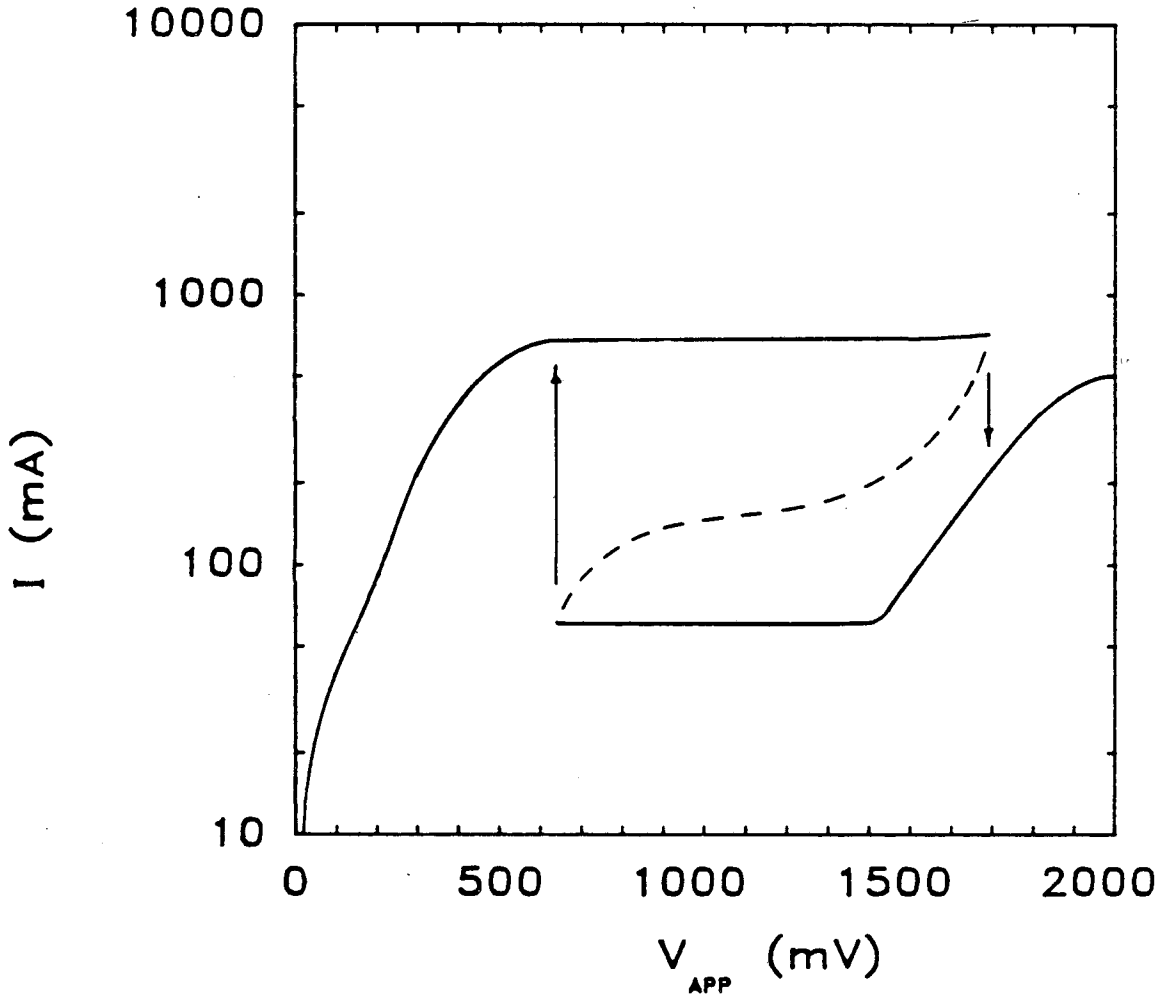


Figure 1A-6. Active-passive behavior of the laboratory-scale cell in terms of total cell current *vs.* applied potential.

"throwing power" of the system. L_P provides a convenient basis for comparing the model to the simpler one-dimensional models, as presented in Figure 1A-7. In this plot of L_P *versus* V_{APP}, the numerical model's prediction is represented by the heavy solid curve. The dashed vertical line denotes the transition to complete passivity at 1688 mV. Plotted for comparison are the predictions made using the models of Edeleanu and Gibson (7) (the fine solid line) and of Fokin and Timonin (13) (the dotted line). For all three models, the onset of transpassivity is marked by an open dot. The extension of the dotted line beyond this point represents the prediction that would be made *discounting* the transpassive portion of the polarization curve (*i.e.* if transpassivity did not occur as for Titanium). This is done to illustrate the point at which the transition to complete passivity would occur, according to the model of Fokin and Timonin. The two one-dimensional models agree remarkably well, considering their different forms[6] (*cf.* Eqs. 1 and 6). The numerical model agrees fairly well with both. The difference is probably due to the approximations inherent in the simpler treatments and to the departure of the cell configuration under study from a one-dimensional description.

It is instructive to observe how the current distribution predicted by the model changes when a different $i-\eta$ characteristic is entered as the boundary condition. Figure 1A-8 shows a piecewise-linear fit of a second polarization curve having a substantially lower value of i_{cr}. As the transpassive tail of the curve is not provided, we simply extrapolate to higher overpotentials with the dotted line at i_p. The resulting series of current-density profiles is shown in Figure 1A-9. Compared to those of Figure 1A-4, these profiles display lower peaks and greater axial extension: the active-passive transition is more gradual, and L_P is greater for a given V_{APP}. This last fact follows

[6] If one were to correct Edeleanu and Gibson's formula for the fact that, in the present example, the anode is not a negligibly thin wire, this estimate for L_P would be 13 percent lower.

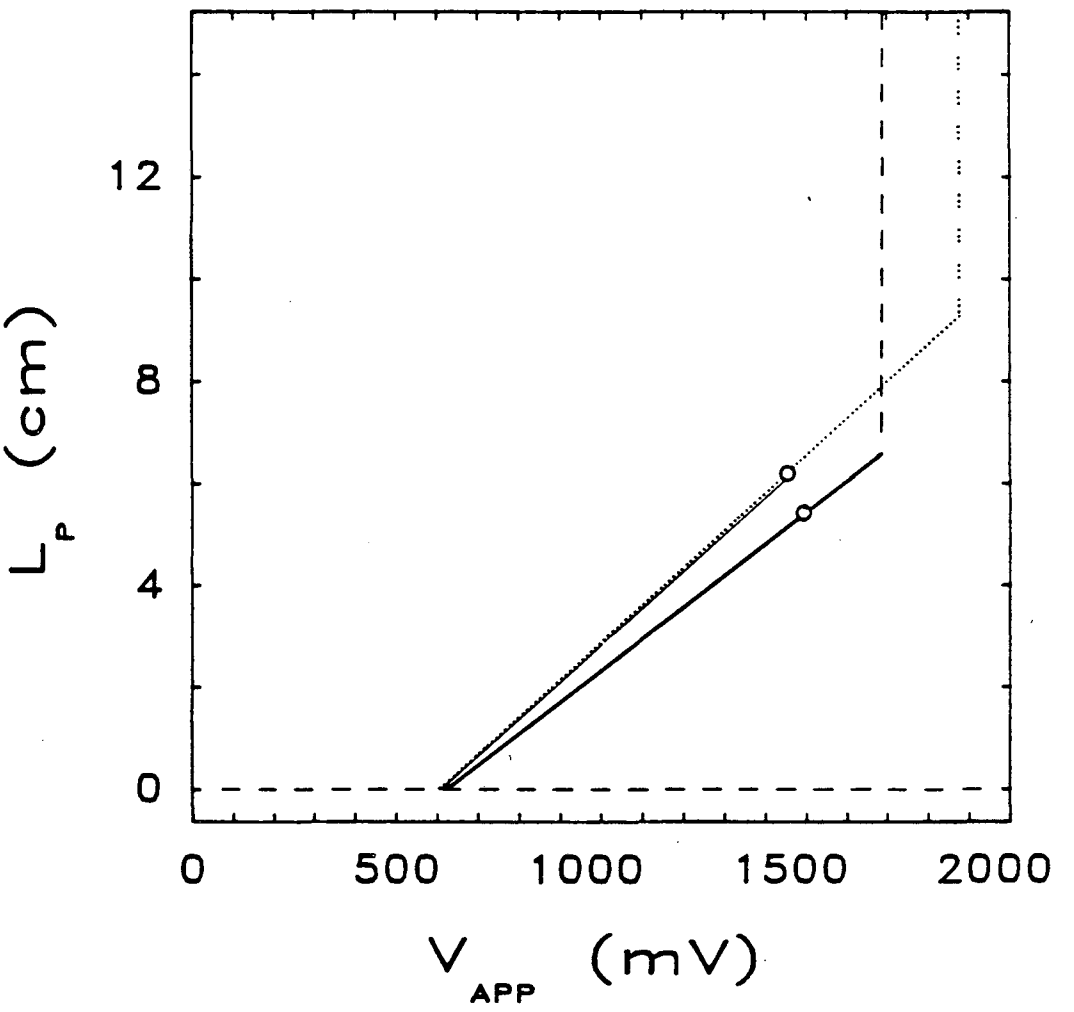


Figure 1A-7. The length of the passive portion of the anode, L_P, as a function of applied potential, as predicted by: 1—the numerical model (heavy solid curve), 2—the analytic model of Edeleanu and Gibson (7) (fine line), and 3—the formula of Fokin and Timonin (11) (dotted line).

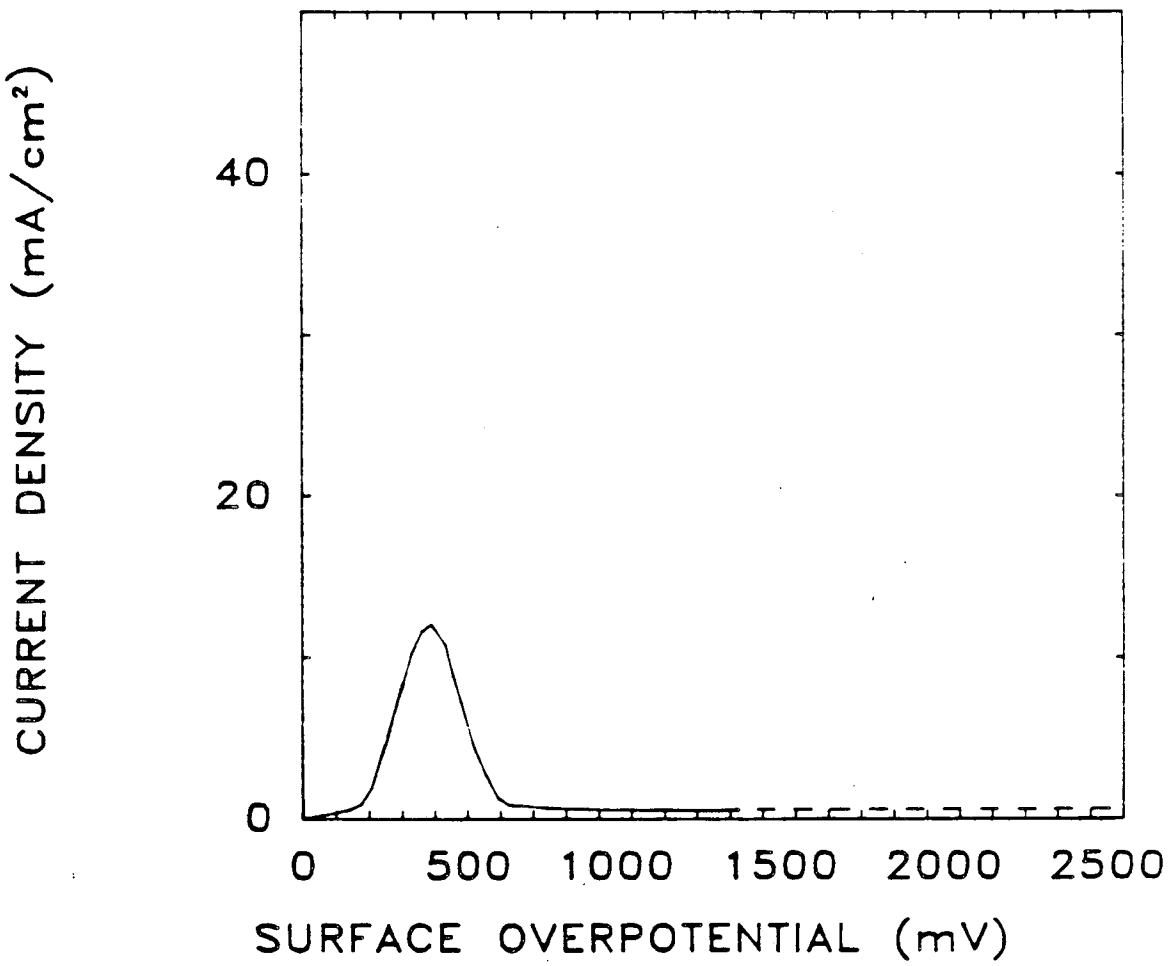


Figure 1A-8. Piecewise-linear fit of a second anodic polarization curve (*cf.* Fig. 1A-1) for Nickel 200 in 2M H_2SO_4, 0.5M $NiSO_4$.

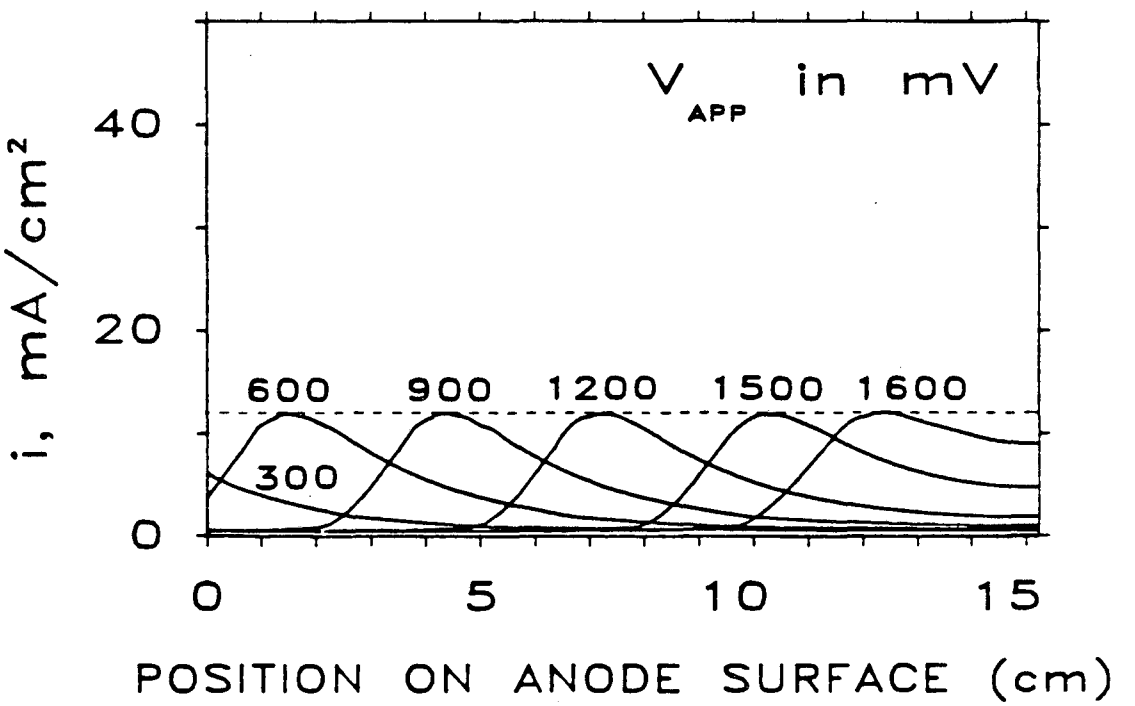


Figure 1A-9. Current density profiles predicted by the model for the laboratory-scale cell, for several values of the applied potential V_{APP}, using the polarization curve of Figure 1A-8.

expectation: since there is less current to be supplied to the active zone, the axial current along the passivated portion of the annulus is lower, and a greater distance can be traversed before the axial ohmic drop exceeds E_P. According to Equation 8, we should expect L_P to be inversely proportional the square root of i_{cr}. The two values of L_P at V_{APP} = 1500 mV taken from Figures 1A-4 and 1A-9 are consistant with a variation of L_P with i_{cr} to the power -0.28.

Demonstration of the Model in an Application to Anodic Protection

To illustrate how the model may be applied to systems of practical interest in anodic protection, we carry out a sample calculation for a sulfuric-acid holding tank. (Storage vessels for concentrated sulfuric acid are the most frequent application of anodic protection to date (6).) The tank contains 67-weight-percent sulfuric acid at 24.2 °C and is constructed of 316 stainless steel. Sudbury, Riggs and Shock (23) present anodic polarization data for this system. A 7-segment trapezoidal fit, which closely approximates the published curve, is given in Figure 1A-10. (We take the the open-circuit potential as the zero of η, and we neglect ohmic potential drop.) The temperature-corrected handbook value (24) for the electrolyte conductivity is 0.302 $ohm^{-1}cm^{-1}$.

We consider a 5800-liter holding tank, of the shape and dimensions shown in Figure 1A-11A. The cathode is a length of pipe entering through the roof and extending along the centerline to within 12 cm of the floor. Although an axisymmetric configuration is not often employed in practice, it does afford the simplicity of reducing the problem from three dimensions to two. Also for simplicity, we assume that the cathode boundary is an isopotential surface. While it would be more realistic to provide for a secondary current distribution here, we choose not to do so in this prelim-

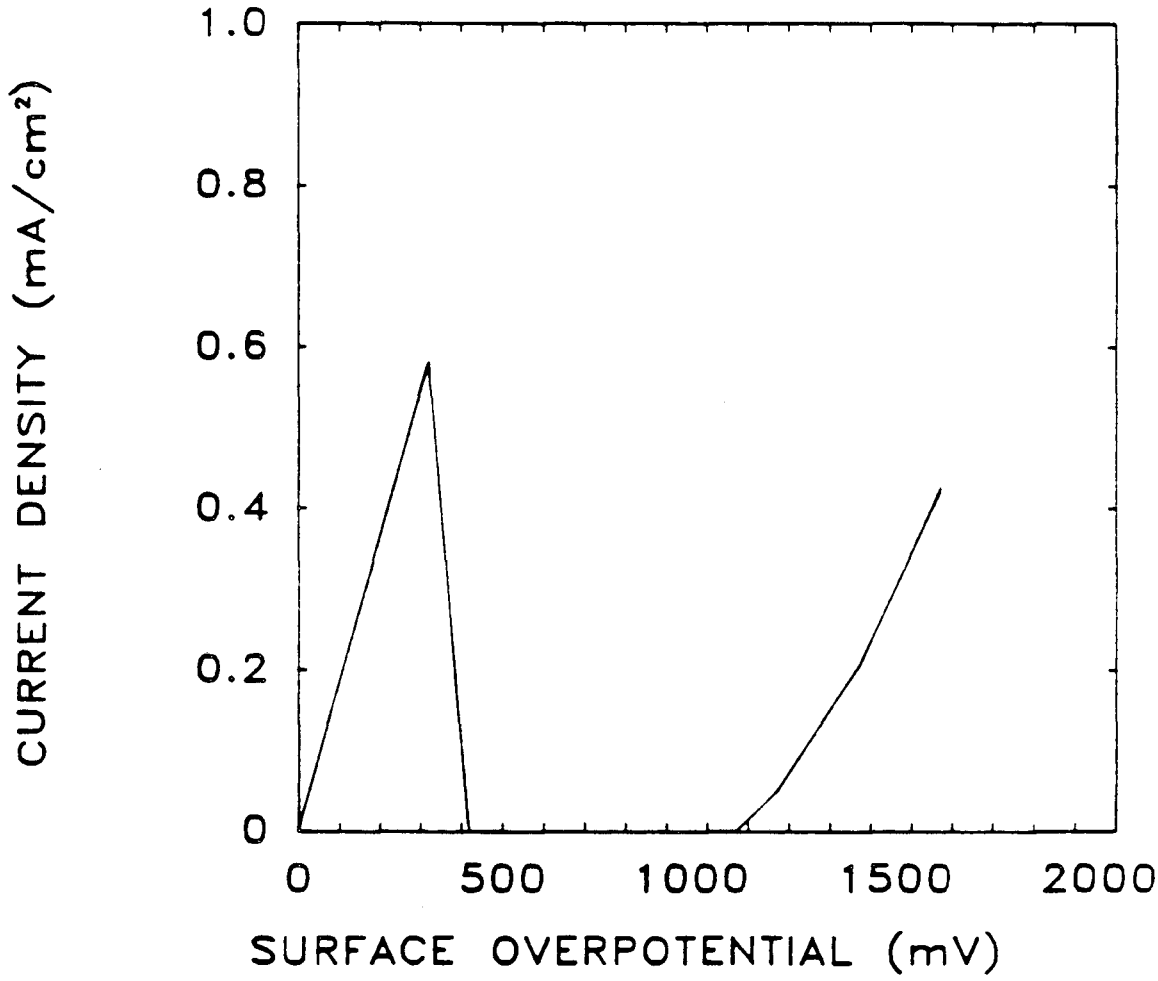


Figure 1A-10. Piecewise-linear curve fit of published anodic polarization data for 316 stainless steel in 67-percent sulfuric acid at 25 ° C (21).

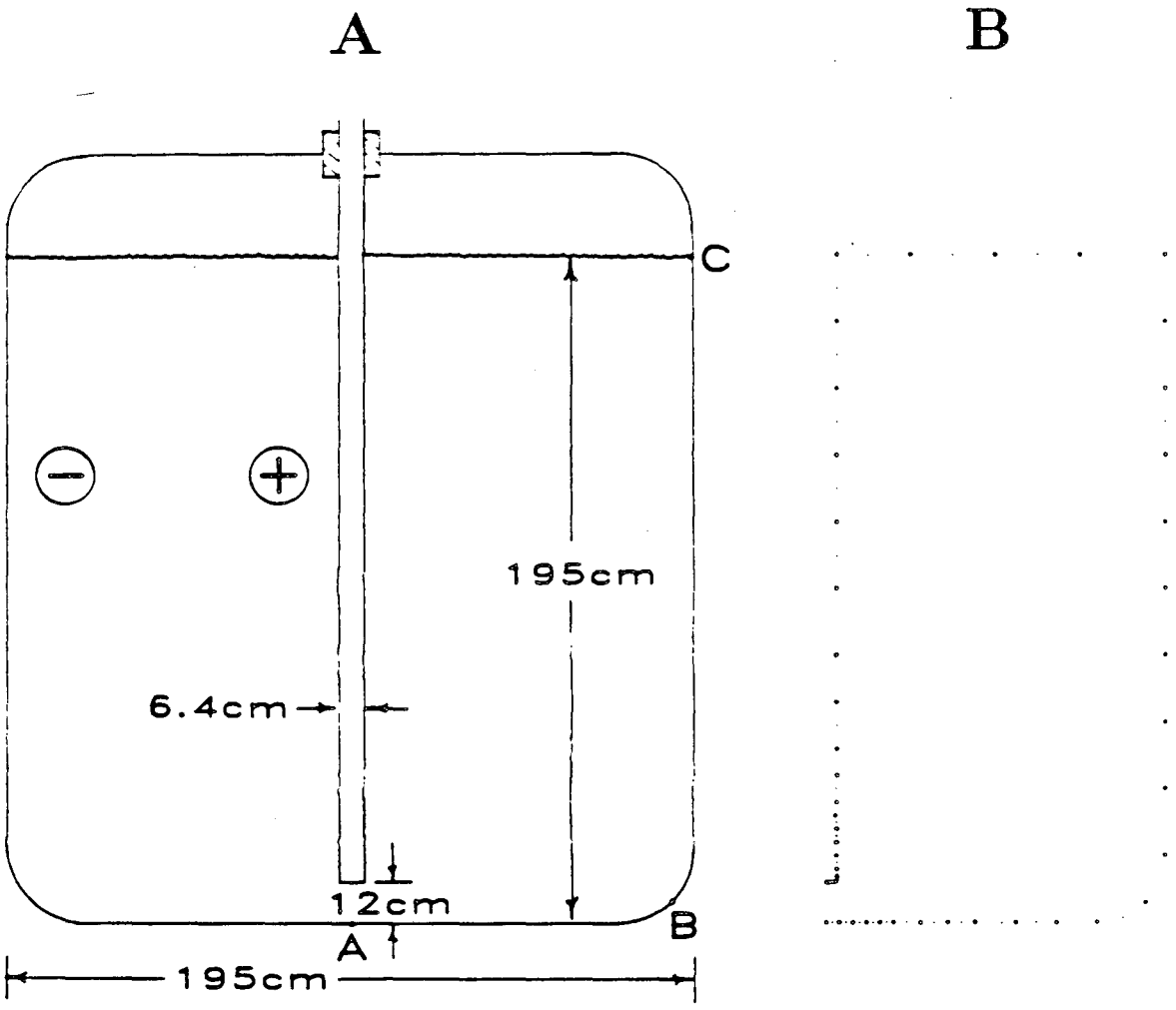


Figure 1A-11. Anodically protected holding tank: A—electrode configuration and dimensions; B—boundary-element nodal structure

inary analysis where we are focusing on the behavior at the anode. Neither do we define the position of the reference electrode, other than to say it is near the cathode, not interfering with the potential field.

The nodal structure for this configuration is shown in Figure 1A-11B. Ninety-three nodes represent the domain boundary in this quadratic boundary-element formulation: 45 of these on the anode (the tank wall) and 39 on the cathode. The axisymmetry line is not discretized. Heavy dots represent element borders; small dots are the internal nodes of the three-node quadratic elements.

The predicted current distribution on the tank wall is shown in Figure 1A-12 for a series of applied voltages, V_{APP}. The horizontal coordinate is distance from the center of the tank floor, measured along the inside surface. Points **A**, **B** and **C** correspond to the similarly labeled points on Figure 1A-11A. Point **B** is in the "bend" between floor and side wall. As before, we mark the peak anodic current density from the overpotential curve by a horizontal dotted line. No curve is shown for the condition $V_{APP} = 0$. Our model would predict uniformly zero current density. This will be unrealistic when, for example, the tank wall is nonuniformly accessible to dissolved oxygen, which is not accounted for in the model. Of course, at open circuit, the corrosion rate is not zero; rather there is a corrosion couple (steel dissolution and hydrogen evolution). The anodic component of this corrosion reaction must exceed i_p, or else anodic protection would be pointless. When the vessel wall is polarized anodically, we appropriately neglect the cathodic process and relate total current density to corrosion rate by Faraday's Law.

At $V_{APP} = 200$ mV we see that there is active anodic current over the entire tank wall, approximately 0.1 mA/cm^2, higher near the cathode tip than elsewhere, and

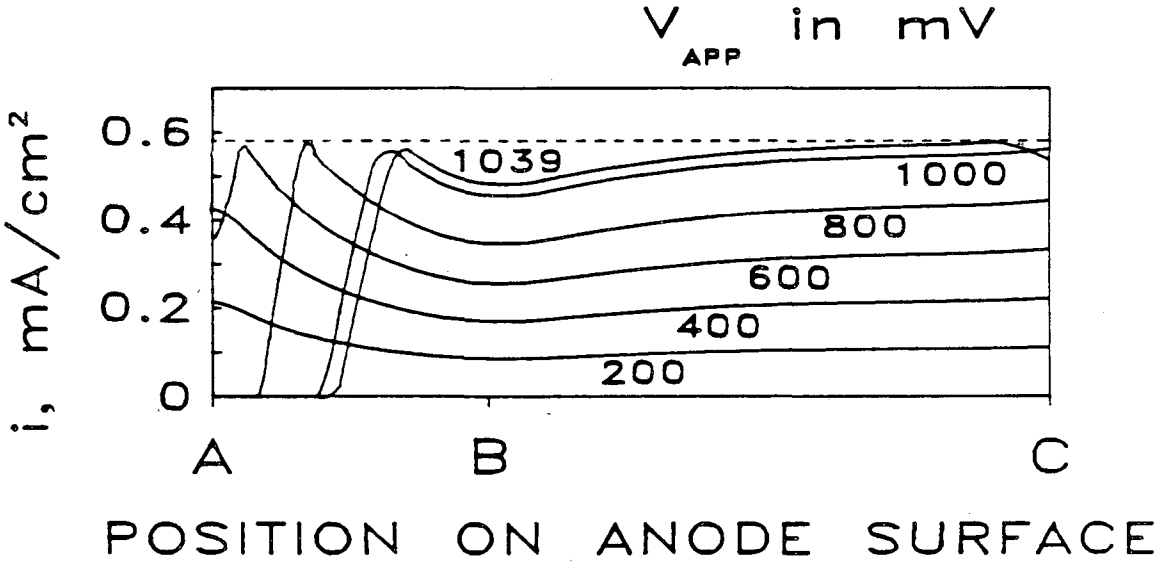


Figure 1A-12. Current-density profiles along the wall of a tank for different values of applied potential.

slightly depressed in the less accessible region where the side wall meets the floor. This behavior follows expectations. The current distribution for $V_{APP} = 400$ mV is qualitatively similar, while expectably of higher current magnitude. For an applied potential of 600 mV, the peak anodic current density, 0.58 mA/cm^2, appears on the tank bottom near the cathode tip. Toward the centerline, the current density falls to about 3.7 mA/cm^2. At $V_{APP} = 800$ mV a small region of the tank floor is passivated. The transition to the peak active current and beyond is gradual, and there remains a slight relative shortage of current to the corner where wall meets floor. When V_{APP} is increased to 1000 mV, the current-density profile shows a larger disk of passive protection centered on the floor (72 cm in diameter); a maximum, still on the floor, but farther out; and current densities at the side wall within 4 percent of the peak value. Slightly beyond this potential, at $V_{APP} = 1039$ mV, the trend reaches a limit: the peak anodic current density is obtained near the top of the wall.

The practical significance of the value $V_{APP} = 1039$ mV is that this must be exceeded in order to establish or to reestablish anodic protection on this system.[7] One might remark that 1039 mV is close to the overpotential of the onset of transpassivity, $\eta_{TR} = 1070$ mV. This means that the vessel under study is nearly the largest that could possibly be driven entirely passive without recourse to special techniques (5).[8]

In terms of total cell current *vs.* applied potential, the steady-state behavior of the tank system is shown in Figure 1A-13. While this shows many similarities to Figure 1A-6, the currents are much higher in magnitude, and there is a small potential range

[7] So far we continue to disregard transient behavior. Further, we know that, once complete passivity has been established, we need only keep V_{APP} above 420 mV to maintain protection.

[8] It is possible to have V_{APP} exceed η_{TR} without having η exceed η_{TR} anywhere on the anode, by relying on a nonzero ohmic potential drop, but this may not be practically advisable.

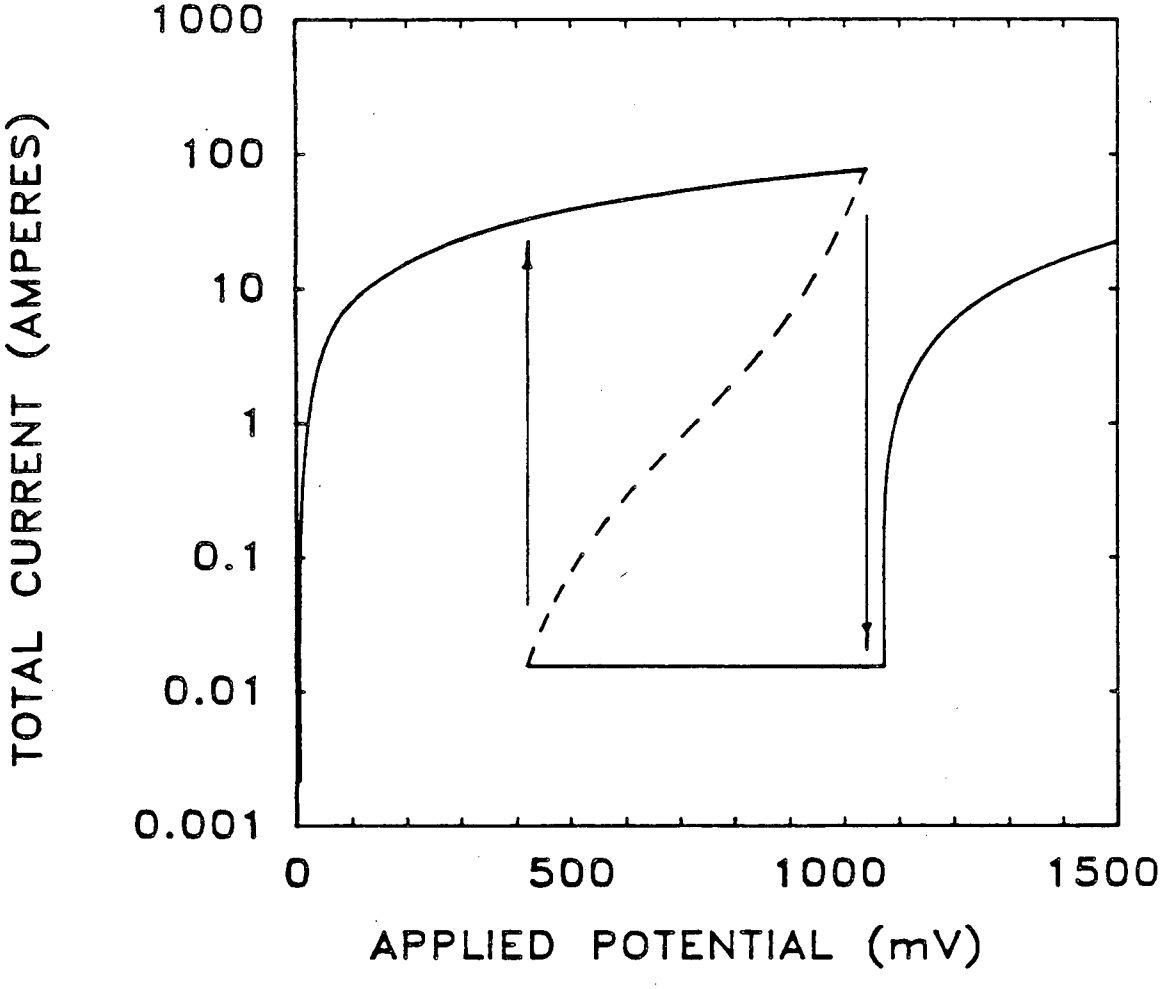


Figure 1A-13. Dependence of total current on applied potential for anodic protection of a holding tank.

where only the passive state is allowed.

A rigorous accounting for the establishment of passivity in an anodic protection installation would have to include transient behavior. The time to passivity might be much shorter than the achievement of steady-state polarization behavior. In general, the initial current exceeds the steady-state value at any given overpotential, and decreases asymptotically to the steady state with time (5,18). This means that our steady-state model would underestimate the total current required to establish passivity and overestimate the size of the largest vessel that could be passivated in a practical length of time. A fully satisfactory treatment of the system would require transient analysis. Such a model could be built on the foundation laid here. Since transients in surface overpotential are certainly much slower than adjustments in the potential field, a pseudosteady-state treatment with time stepping could be used. Computational expense would be significantly higher than for the present steady-state model. A transient analysis of anodic protection would also require good characterization of the unsteady kinetic behavior, either empirically or in terms of a model of the growing passive film. Time-varying polarization has been included in models of cathodic protection (25), which, moreover, employ the boundary-element method in three dimensions.

Conclusions

A numerical model has been devised for the calculation of current distribution in active-passive systems. In example computations, the method is shown to be convergent over an important range and moderate in cost. The model can be used to explore the behavior of distributed active-passive systems. In its present form (using predictor-corrector continuation) one can trace certain portions of the solution curve in

$I-V_{APP}$ space. The results are consistant with a hysteresis behavior similar to that expected for the simpler system of uniform current density at a passivating anode with ohmic resistence in series. We conjecture that the portions of the solution curve accessible by the present model correspond to stable physical states of the actual system, and that the unconverged regions correspond to physically unstable conditions.

In a nearly one-dimensional cell with an axial anode and the cathode positioned at one end, the position of the active-passive transition is a roughly linear function of applied potential. Reasonable agreement with analytic one-dimensional models in the anodic-protection literature is observed.

In a preliminary study, the model's application to anodic protection systems has been demonstrated. The results suggest that the model could ultimately be used to deterimine 1) a lower bound for the the total current required to achieve complete passivity on a given structure, 2) the size of the largest vessel that can be driven to complete passivity, and 3) the size of the largest vessel that, once passive, can be so maintained. The model could also aid in design decisions regarding vessel shape and placement of the counter electrode.

For the cell geometry of Figure 1A-3A, which is nearly one-dimensional in character, reasonable agreement is found with the one-dimensional, analytic models in the literature (7,13). The distinguishing feature of the new model is that it is generally applicable to two-dimensional and axisymmetric geometries such as the tank configuration of Figure 1A-11. Extension of the model to three dimensions could be done by implementing a three-dimensional boundary-element routine; these are increasingly common especially in problems of linear elastostatics (26), and have found application in current-distribution calculations for cathodic protection (25).

References

1. N. Vahdat and J. Newman, *J. Electrochem. Soc.*, **120**, 1682 (1973).

2. J. Newman, *J. Electrochem. Soc.*, **113**, 1235 (1966).

3. C. G. Law, Jr. and J. Newman, *J. Electrochem. Soc.*, **126**, 2150 (1979).

4. C. G. Law, Jr. and J. Newman, *J. Electrochem. Soc.*, **133**, 37 (1986).

5. O. L. Riggs and C. E. Locke, "Anodic Protection: Theory and Practice in the Prevention of Corrosion," Plenum Press, New York, 1981.

6. C. E. Locke, International Congress on Metallic Corrosion, Vol. 1, Toronto, Canada, June 3-7, 1984, National Research Council of Canada, Ottawa, Canada, 1984.

7. C. Edeleanu and J. G. Gibson, *Chemistry and Industry*, **10**, 301 (1961).

8. W. A. Mueller, *J. Electrochem. Soc.*, **110**, 698 (1963).

9. M. N. Fokin and V. A. Timonin, *Dokl. AN SSSR*, **164**, 150 (1965).

10. V. A. Makarov, Ya. M. Kolotyrkin, V. M. Knyazheva, and E. B. Mamin, *Zashchita Metallov*, **1**, 662 (1965).

11. W. P. Banks and J. D. Sudbury, *Corrosion*, **19**, 300t (1963).

12. J. R. Myers, F. H. Beck and M. G. Fontana, *Corrosion*, **21**, 277 (1965).

13. V. A. Timonin and M. N. Fokin, *Zashchita Metallov*, **2**, 307 (1966).

14. G. A. Prentice and C. W. Tobias, *J. Electrochem. Soc.*, **129**, 72 (1982).

15. J. Dukovic and C. W. Tobias, submitted to *J. Electrochem. Soc.*, May, 1986.

16. R. F. Heinemann, K. A. Overholser and G. W. Reddien, *Chemical Engineering Science*, **34**, 833 (1979).

17. C. J. Chatfield and L. L. Shreir, *Corrosion Science*, **12**, 563 (1972).

18. L. J. C. Urdaneta, "Time Dependence of Passivating Current for Anodic Protection," Masters Thesis, University of Oklahoma, Norman, 1981.

19. H. B. Keller, In "Applications of Bifurcation Theory," P. H. Rabinowitz, Ed., 359, Academic Press, New York, 1977.

20. I. Epelboin, C. Gabrielli, M. Keddam, J. C. Lestrade and H. Takenouti, *J. Electrochem. Soc.*, **126**, 1632 (1972).

21. P. P. Russell and J. Newman, *J. Electrochem. Soc.*, **130**, 547 (1983).

22. J. Dukovic and C. W. Tobias, (Part B of this series) submitted to *J. Electrochem. Soc.*, September, 1986.

23. J. D. Sudbury, O. L. Riggs, Jr., and D. A. Shock, *Corrosion*, **16**, 91 (1960).

24. Landolt-Bornstein, Sechste Auflage, II. Band, 7. Teil, Springer-Verlag, Berlin, 1960.

25. P. O. Gartland and R. Johnsen, Corrosion 85, Boston, Massachusetts, 1985, paper 319, National Association of Corrosion Engineers, Houston, Texas.

26. C. A. Brebbia and S. Walker, "Boundary Element Techniques in Engineering," pp. 25-53, Newnes-Butterworths, Boston (1980).

LIST OF SYMBOLS

E_A	potential range spanned in the active regime, mV
E_P	potential range spanned in the passive regime, mV
E_P^{max}	maximum potential range that can be spanned in the passive regime without incurring transpassivity, mV
F	Faraday's constant, 96,487 C/equiv
i	current density, mA/cm^2
$\bar{i}$	mean current density in the active loop of the polarization curve, mA/cm^2
$\underline{i}$	vector of current-density values at the electrode-surface nodes, mA/cm^2
i_{cr}	peak active current density on polarization curve, mA/cm^2
i_p	current density in passive potential range, mA/cm^2
I	cell current, mA
L_A^{∞}	length of the active zone on an infinitely long anode, cm
Lef	A tube of this length can be driven to passivity; cm
L_M	A tube of this length can be maintained in the passive state; cm
L_P	length of the passive zone, cm
L_P^{∞}	length of the passive zone on an infinitely long anode, cm
L_P^{max}	maximum possible length of the passive zone without incurring transpassivity, cm
$\mathbf{n}$	unit vector normal to domain boundary, dimensionless
r	radius of wire anode or metal tube, cm
R	tube radius, cm
V	cell voltage, $\phi_M^a - \phi_M^c$, mV
V_{APP}	voltage between anode and cathode in excess of the open-circuit value, mV
V^{oc}	cell voltage at open circuit, mV
κ	electrolyte conductivity, ohm^{-1}cm^{-1}

η	surface overpotential, mV
η_F	surface overpotential corresponding to the Flade potential, mV
η_{TR}	surface overpotential corresponding to the onset of transpassivity, mV
ϕ	potential, mV
ϕ_M^a, ϕ_M^c	potential of the metal in anode and cathode, mV
ρ	electrolyte resistivity, *ohmcm*

Current Distribution in Electrochemical Cells with Passivating Kinetics

B. Experimental Determination

ABSTRACT

An experimental study of current distribution in passivating systems is presented, with special attention to the steady-state coexistence of active and passive regions on the anode surface. Current-density profiles were measured by weight loss on a segmented rotating-cylinder anode of nickel in acid nickel sulfate electrolyte, with the cathode placed so as to produce a highly nonuniform potential field. Independent measurements of the steady-state, anodic current-overpotential characteristic were made on rotating-disk and rotating-cylinder electrodes. The measured current-density profiles were compared to predictions from a theoretical model developed for this purpose. The agreement is generally satisfactory, but the measured length of the passive zone exceeds the predicted value, implying that the model would supply a conservative estimate of throwing power in anodic protection.

Introduction

Passivity is a phenomenon of great importance in electrochemical technology, particularly in the field of Corrosion (1,2). The kinetics of passivating electrodes are complex, involving the formation and sustained existence of a thin oxide film. The behavior of an active-passive system is especially complicated when the current distribution is nonuniform, as it is in most practical electrochemical cells. A particularly interesting condition is the side-by-side coexistence of active and passive zones on the anode surface. An improved understanding of distributed active-passive systems would benefit the technology of anodic protection [1] (3) and assist in the interpretation of active-passive polarization measurements (for example on a large rotating disk with nonuniform current density).

While numerous experimental determinations of current distribution have been reported in the literature (*eg.* 4,5,6,7,8,9,10) measurements on active-passive systems have been quite limited. LaQue (11) observed regions of differing corrosive attack on rotating iron disks in seawater. The regions correspond to active and passive zones arising from uneven oxygen transport. The entire current-distribution could not be accurately measured by this technique, but the position of the active-passive boundary was recorded and shows fair agreement with a model by Newman and Law (12).

Epelboin, *et al.* (13) also observed the localized dissolution of an iron rotating-disk electrode at different currents and rotation speeds. They determined electrode profiles by depth-of-field measurement with an optical microscope. A sharp boundary between the active and passive regions was seen. Under certain conditions, the peripheral region of the disk was found to passivate while the center of the disk underwent active

[1] Anodic protection is a corrosion-control technique in which the passive state is electrochemically induced over the surface of the structure to be protected.

dissolution. In other cases this pattern was reversed, and in still other instances, there was a ring of active attack bordered both inside and outside by passive zones.

Russell and Newman (14) made an indirect experimental determination of the position of the active-passive transition on iron rotating disks in 1M H_2SO_4 by interpreting current-voltage curves in terms of a model developed by Law and Newman (15). Russell and Newman found that, with rising electrode potential, the average current density on the active inner portion of the disk increases as the size of the active area decreases.

In experiments conducted by researchers in the anodic-protection field (16,17), the anode was a thin wire stretched along the axis of a glass tube filled with electrolyte, and the cathode was positioned at one end. On the anodes used (18-8 steel in 30-percent H_2SO_4, and steel 3Kh13 in 3-percent H_2SO_4) the active region turns black; thus it is easy to locate the active-passive transition. However, other details of the current distribution are not available by this technique. The measured data were used to confirm various one-dimensional models (16,17,18,19,20).

The purpose of the present experimental study is to examine the steady-state behavior of active-passive systems with nonuniform current distribution. Our approach is to use a sectioned electrode to measure the steady-state current distribution at different values of applied potential. One of our goals is to validate the theoretical model presented in Part A.

Experimental Apparatus

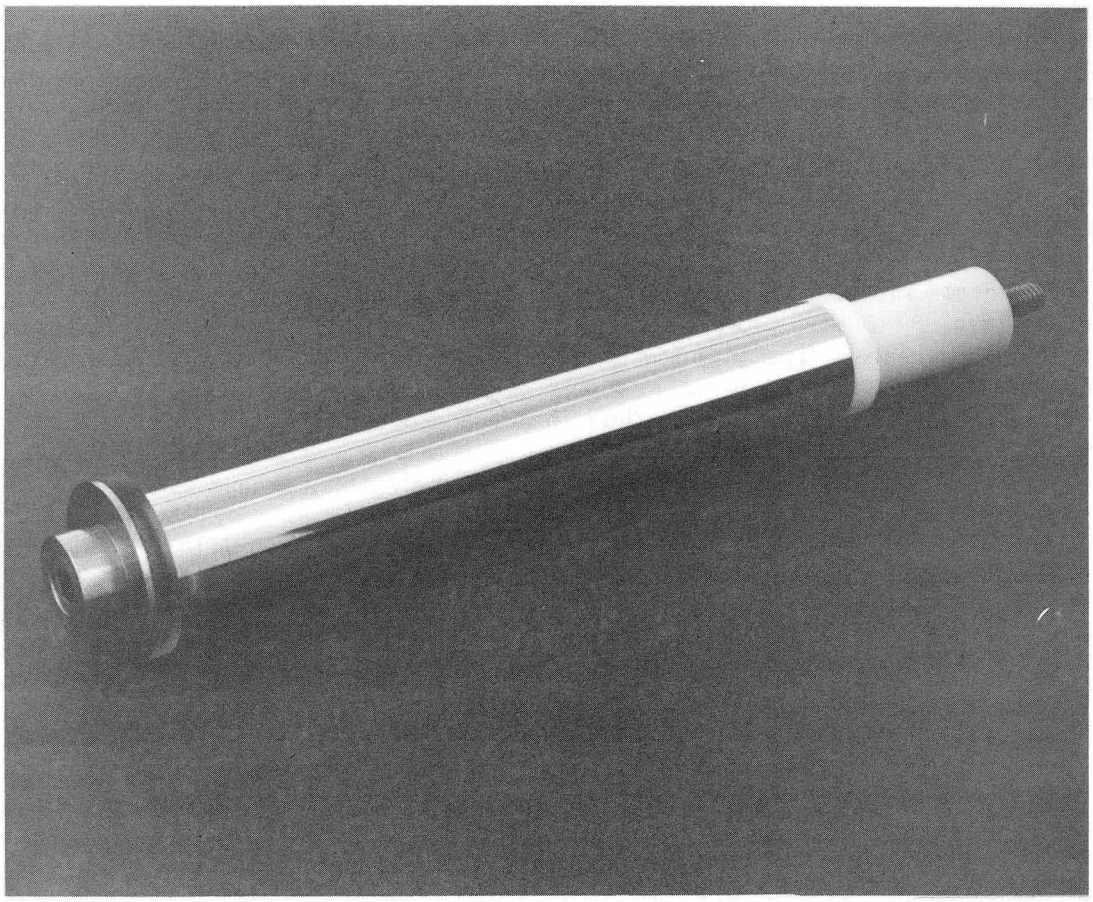
For the experimental determination of current distribution in active-passive systems, we designed and built a special laboratory cell. To promote a highly nonuniform current distribution (especially the coexistence of active and passive regions) we

employed a highly asymmetric electrode configuration.

A rotating-cylinder electrode (RCE) was chosen for several reasons. First, this system ensures a uniform, controlled mass-transfer condition over the electrode surface. While this criterion is also met with a rotating-disk electrode, the RCE is much easier to divide into segments. A long, segmented, rotating cylinder offers the possibility of good spatial resolution of current density. Finally, in contrast to planar electrodes in a channel-flow cell, the RCE has the advantage of axisymmetry: it can be machined and polished on the lathe, and there is less trouble with edge effects.

The anode chosen for this study was nickel in 2 M sulfuric acid, 0.5 M nickel sulfate. (The nickel salt was added to prevent large relative changes in concentration during an experiment.) Nickel was chosen for a number of reasons. The anodic dissolution of nickel in aqueous H_2SO_4 is relatively simple and well understood (21,22,23). There is a single oxidation state, and the current efficiency is essentially 100 percent. An acceptably pure grade of nickel, Nickel 200, is available in drawn-pipe form, suitable for the manufacture of rotating-cylinder electrodes. The material is well suited for machining and polishing and does not present a severe toxicity hazard. The peak active current density i_{cr} is of the proper magnitude for observing the active-passive transition in the laboratory on the convenient scale of 10 cm. (The throwing-power formula of Edeleanu and Gibson (16) suggests that i_{cr} is too high for this on iron and too low on many stainless steels.) Finally, the passive potential range is sufficiently broad for experimentation over a range of applied potentials.

Figure 1B-1 is a photograph of the anode spindle. The nickel surface is approximately 2.5 cm in diameter and 15 cm in length. From the photograph, as with the unaided eye, it is difficult to discern the boundaries between the ring segments that

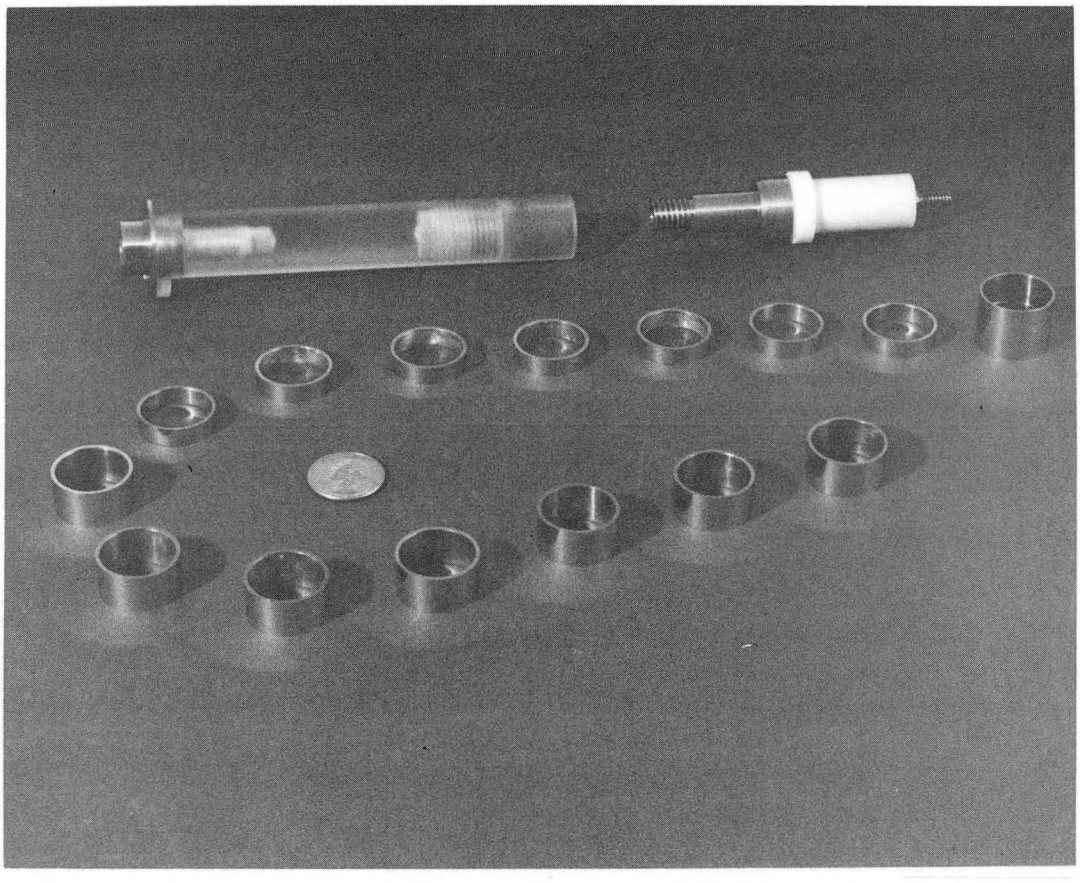


XBB 869-7734

Figure 1B-1. Assembled rotating-cylinder anode.

compose the nickel surface. Figure 1B-2 shows the spindle dissembled. Each ring slides snugly onto the Lucite shaft. The rings have an inside diameter of 2.29 cm and are either 1.27 or 0.64 cm in height. The shorter rings are positioned where the active-passive transition is expected to occur to provide greater resolving power in that region. The top ring is 1.91 cm long and contains a step on its inside surface. When the threaded nickel headpiece is screwed into the Lucite shaft, the headpiece presses axially against the inside step of the top ring, making electrical contact, and compressing all of the rings into a snug column of electrically-connected segments. The bottom ring butts against a collar of Lucite that is integral with the shaft and supported from below by a stainless-steel tailpiece. This tailpiece is electrically isolated from the nickel segments and inserts in a teflon bore in the stationary cell floor, keeping the end of the shaft on center. The tailpiece is attached to the Lucite shaft by a hexhead drawbolt tightened to a torque higher than that used to compress the rings or to attach the spindle assembly to the rotator shaft; accordingly, this hexhead can be used to tighten either of the latter joints. The nickel headpiece contains a steel stud, threaded at the top, which is used to attach the spindle assembly to the rotator shaft. The headpiece is protected from contact with electrolyte by a teflon sleeve, which is compressed when the spindle is joined to the shaft.

The rotating-cylinder cell is provided with axial flow so that the electrolyte could be exchanged with a large reservoir. Reasons for this provision are discussed later. The cell is depicted schematically in Figure 1B-3 and photographed in Figure 1B-4. The electrolyte volume is an annulus bounded by the anode spindle on the inside and a Lucite tube on the outside. Since the latter is transparent, one can observe the anode surface during the course of the experiment. A fiber-optic lamp is used to illuminate the subject. The electrolyte is pumped through the floor of the cell, a Lucite disk, 2.4



XBB 869-7735

Figure 1B-2. Disassembled rotating-cylinder anode, showing Lucite shaft, headpiece, and set of nickel rings.

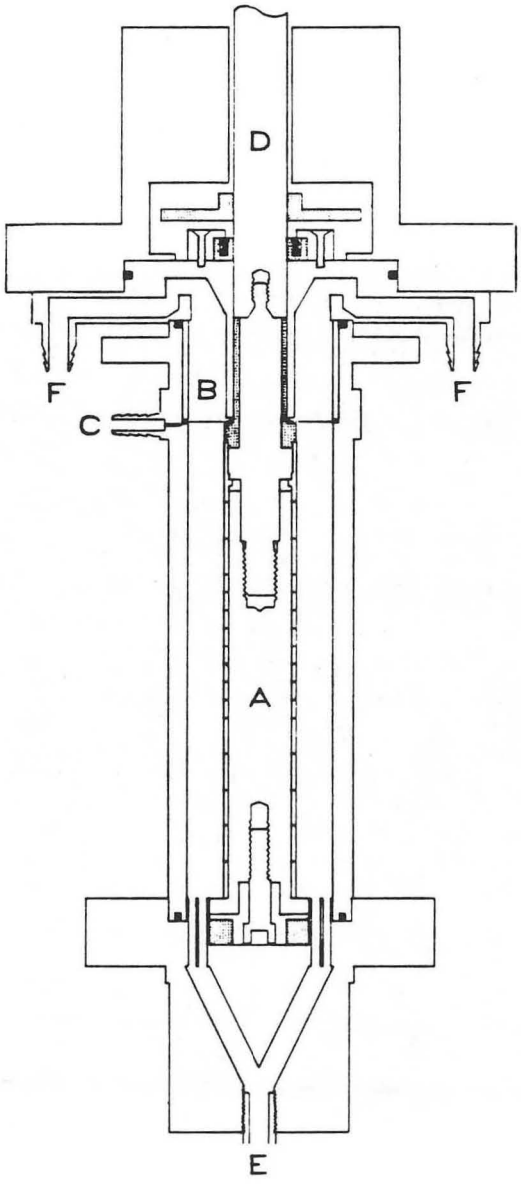
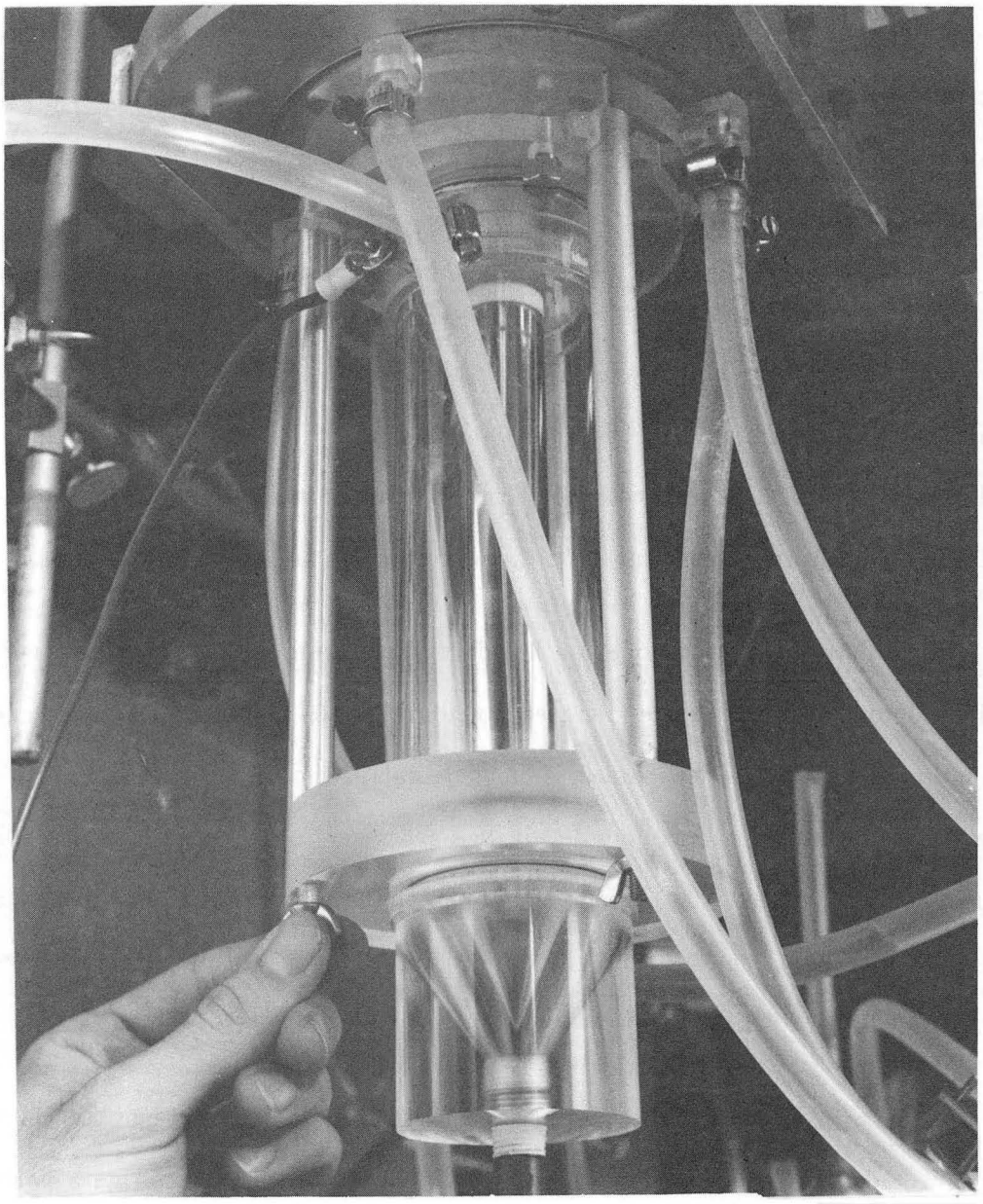


Figure 1B-3. Rotating-cylinder cell: **A**—anode spindle, Lucite; **B**—cathode, platinum mesh; **C**—Luggen-capillary port; **D**—rotator shaft; **E**—flow inlet; **F**—flow outlets. Teflon parts are shaded; o-rings are in black.



XBB 869-7736

Figure 1B-4. Rotating-cylinder cell, with anode in place and floor installed.

cm in thickness, with a dense, regular pattern of 0.32-cm holes. Below this manifold is a conical flow separator to promote uniform inlet flow. This cell floor is detachable for convenient installation of the anode spindle and access to the anode during pretreatment. At the top of the cell, there is an overflow chamber with three exit ports. There are also two ports for a nitrogen-gas blanket. Five centimeters above the top nickel ring on the rotator shaft, there is a teflon seal with a steel retaining spring.

The rotator shaft is belt driven by a 90-watt, variable-speed motor. The shaft is supported by three ball bearings and contains a bellows joint. Electrical contact is made to the shaft via four sets of "finger contacts," flexible brass sheets cut in a finger-like pattern. The resistance of the rotating contact was determined to be below 1 mΩ (by measuring from one set of fingers, through the shaft, to another set).

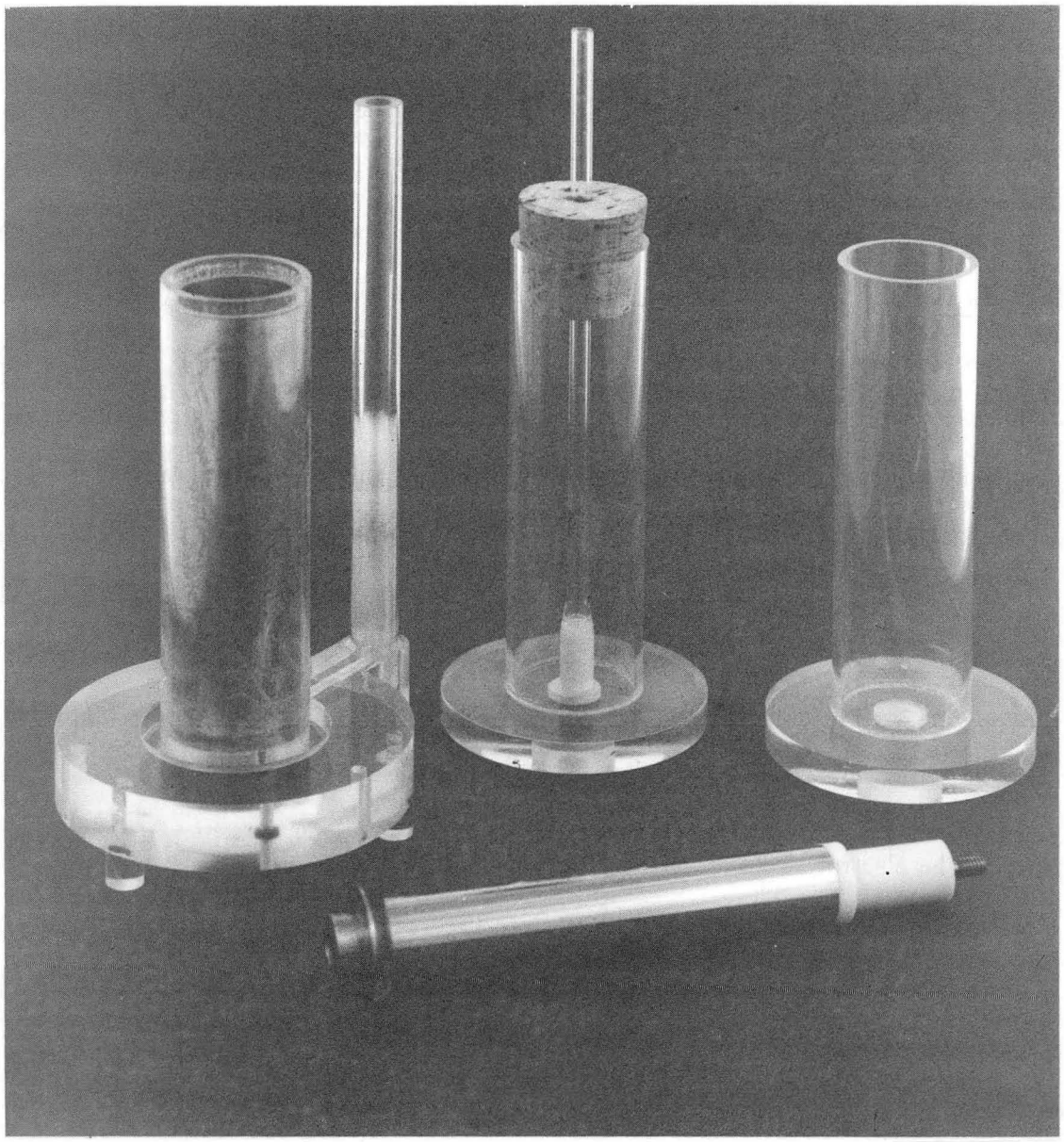
The flow-through cathode is a flat disk of platinum mesh (200-micron wires spaced 600 microns apart). This is mounted horizontally 0.64 cm above the top nickel ring on the anode spindle. There is a 2.54-cm hole cut in the mesh, and lined with platinum wire, to accommodate the spindle. It was necessary to drill an additional hole, 0.8 cm in diameter, to allow the passage of bubbles that would otherwise aggregate under the mesh. Electrical connection to the potentiostat is made with a platinum wire through the cell wall.

The reference-electrode chamber is a glass tube located outside the cell body and connected by Bev-a-line IV tubing (ethyl-vinyl-acetate tubing lined with polyethylene). The connection is made to a nipple on the cell wall, which leads to an aperture less than 0.05 cm in diameter at the inside surface of the Lucite wall. This aperture is located at the same vertical position as the cathode and serves as a flush-mounted luggin capillary. The glass reference chamber is immersed in a bath to keep the reference

electrode at nearly the same temperature as the flowing electrolyte. The reference electrode is a saturated mercurous-sulfate electrode, Radiometer America K601.

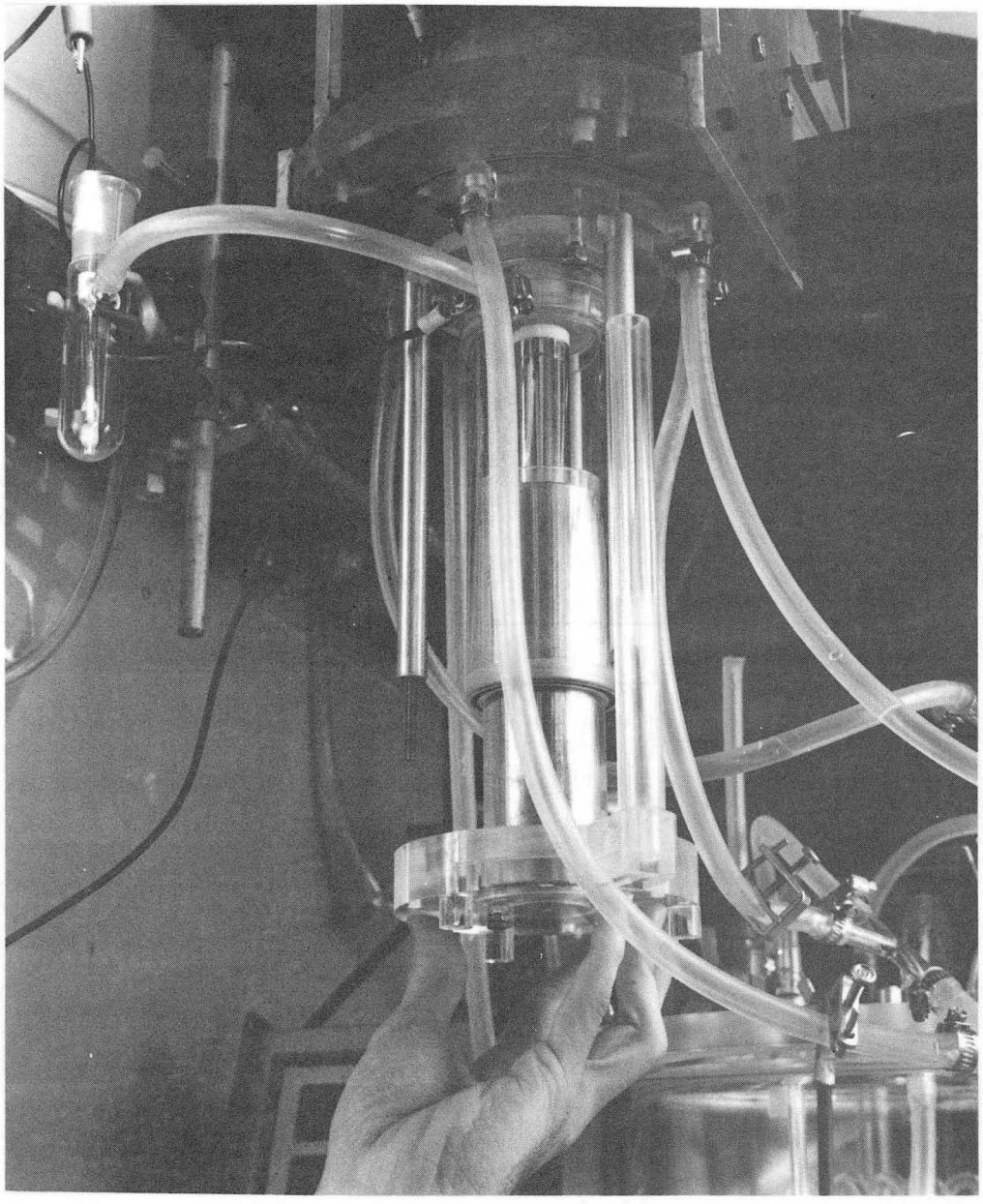
The cell is equipped with a pretreatment vessel and two rinse columns shown (behind the anode) in Figure 1B-5. The pretreatment vessel is a Lucite column fitted with a Nickel-200 sleeve, to which electrical contact is made through the vessel floor. With the cell floor dismounted, the vessel can be raised into the cell body, as shown in Figure 1B-6. The nickel sleeve forms a cylindrical counter electrode for the pretreatment electrolysis, keeping with the annular configuration necessary for uniform current distribution. Accordingly, the floor of the pretreatment vessel is a horizontal insulator, and the upper insulating boundary is formed by the free surface between the pretreatment electrolyte and the nitrogen blanket. To aid in adjusting the level of this interface, the pretreatment vessel is outfitted with a thin transparent side-column, which connects to the main column at its base. The Lucite rinse columns are of proper size to be raised into the cell body, immersing the anode spindle in rinse water.

The electrolyte flows vertically upward through the annular cell and returns to a reservoir. The total volume of electrolyte used in the system is 7 liters, sufficient to prevent relative concentration changes greater than 1 percent over the course of a typical experiment. The flow system is diagrammed in Figure 1B-7 and pictured in Figure 1B-8. An 8-liter desiccator jar with a Lucite top serves as the electrolyte reservoir. The top is clamped to the jar and seals to the ground-glass lip of the jar with an o-ring; the airtight chamber is kept under a nitrogen blanket prior to and during each run. A glass down tube leading to an 8-cm fritted-glass disk is provided for nitrogen sparging. Temperature control is actuated with a 300-watt, Sethco TH-300, quartz-tube immersion heater. For runs below room temperature, a stream of -10 °C methanol from a Lauda Ultra UK60D cryostat is passed through a glass cooling coil



XBB 869-7737

Figure 1B-5. Electrode-pretreatment vessel (left), rinse columns (center and right), with anode in foreground.



XBB 869-7738

Figure 1B-6. Pretreatment vessel being raised into position.

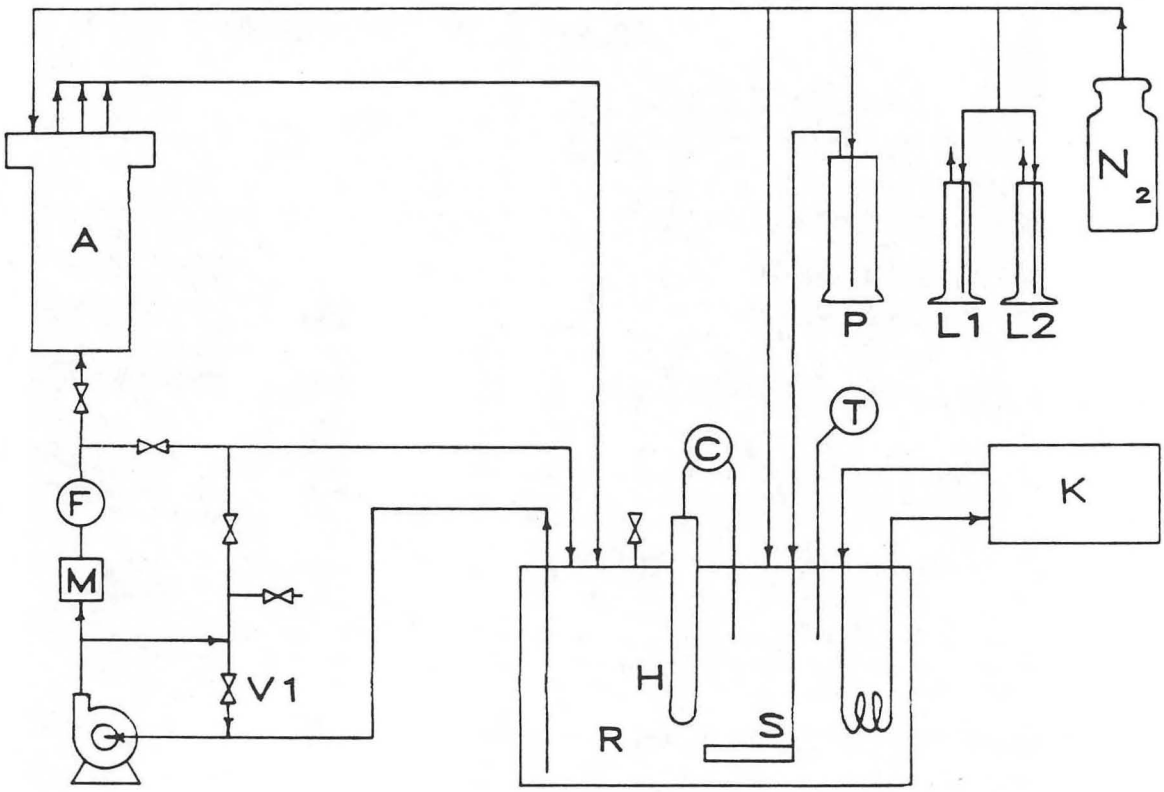
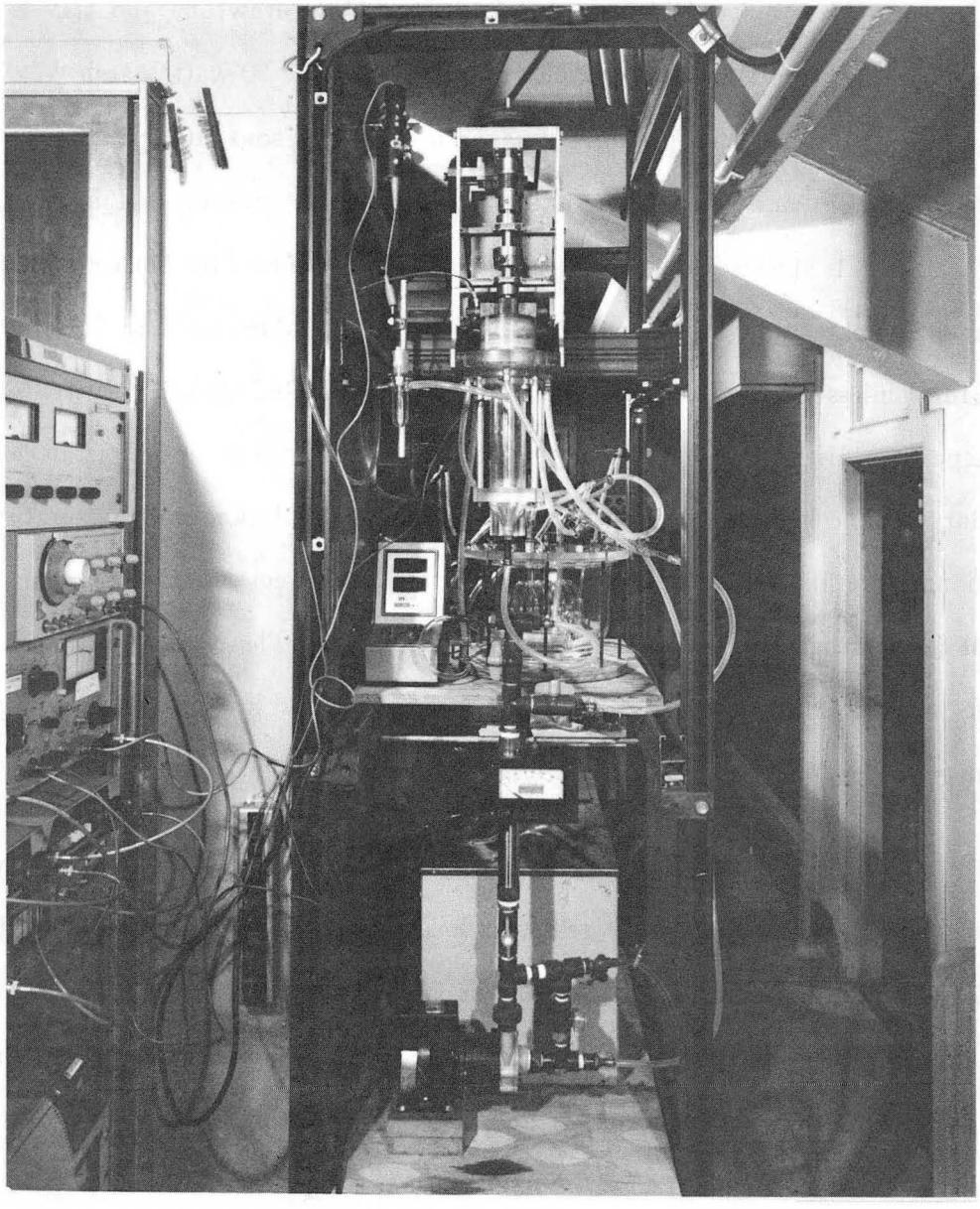


Figure 1B-7. Flow system: **A**—cell, **F**—filter, **M**—flowmeter, **V1**—recycle valve, **R**—reservoir, **H**—immersion heater, **C**—temperature controller, **S**—sparging tube, **T**—temperature probe, **K**—cryostat, **P**—presparger, **L1** & **L2**—rinse columns



XBB 869-7739

Figure 1B-8. Rotating-cylinder apparatus.

immersed in the electrolyte reservoir. Temperature is monitored in this reservoir, in the cryostat reservoir and in the reference electrode chamber.

Electrolyte exits the reservoir through a Lucite drawtube and travels through 1-cm-ID Bev-a-line IV tubing to the pump intake. The 90-watt March AC-5C-MD centrifugal pump is equipped with magnetic coupling so that the fluid contacts only polypropylene and Viton. There is an immediate recycle loop at the pump, controlled by valve V1, shown in Figure 1B-6. The plumbing and valves are 1.8-cm-ID PVC pipe, and all connecting tubes are Bev-a-line IV. A variable-area dial flow indicator with 316-stainless-steel interior is used (Fischer-Porter 10A2227A). A fritted-glass phase-separator/filter is used to prevent particles or bubbles from entering the cell. A secondary recycle line to the reservoir is provided for circulation and temperature stabilization before filling the cell. Cell effluent is returned directly to the reservoir, where most of the hydrogen bubbles separate from the solution. The number of materials brought in contact with the electrolyte has been kept as low as possible: glass, PVC, polyethylene, polypropylene, stainless steel, teflon, Lucite, quartz and Viton.

Nitrogen gas (Liquid Carbonic "Hi Pure") is carried through 0.6-cm Tygon tubing to the reservoir sparger, the reservoir head, the cell head (particularly when the cell floor is dismounted, to flush the cell volume and exclude oxygen), and to sparging tubes for the two pretreatment rinse columns. The nitrogen used to sparge the reservoir passes first through a glass sparging column of identical electrolyte to prevent concentration changes in the reservoir upon sparging.

The cell is powered with a Princeton Applied Research 371 Potentiostat/Galvanostat, capable of delivering 7 Amperes. During the potential ramp to steady state, the potentiostat is programmed by a Hewlett-Packard 3325A

Synthesizer/Function Generator. Total cell current and the Anode potential are monitored on a Nicolet 4094 Digital Oscilloscope. After each run, these data are dumped to a Hewleltt-Packard 9825A desktop computer and stored on a floppy disk. The 12 Amperes of pretreatment current are provided by a Hewlett-Packard 6259B DC Power Supply. The current is measured as voltage drop across a 5-milliohm shunt using a Kiethly 173A Digital Multimeter. The rotor speed is controlled by a Minarik Digi-Lock speed controller, and calibrated with a handheld digital tachometer. The temperature controller is a Versatherm 2156 proportional controller using YSI 400 Thermistor probes. Temperature is measured on a Digitec HT 5810 digital thermometer with YSI 700 thermistor probes.

Experimental Procedure

Each experiment uses seven liters of fresh electrolyte, 2.00 M H_2SO_4 / 0.50 M $NiSO_4$, prepared from crystalline nickel-sulphate hexahydrate and 95-percent sulfuric acid (both Mallinkrodt, Analytical-Reagent grade) and dionized, distilled water.

The set of nickel rings to be used for a single experiment is mounted on a brass polishing spud; the rings slide snugly onto one section of the spud and are compressed axially when the other section is screwed into place. This assembly is mounted on the lathe with less than 100 microns radial runout using a six-jaw chuck and a live-center tailstock. If the surface is initially very rough, such as after a previous run wherein part of the surface has been severely pitted, it is necessary to make a finish cut with a carbide tool. The first stages of polishing are done with dry, emery-cloth strips. One 100-cm strip of each grade, 180, 240, 400 and 500, is used for roughly ten minutes at 1800 rpm. After this, the surface is polished with 9-micron diamond paste using a strip of felt polishing cloth. Finally, the same is done with 1-micron diamond paste. The

surface is buffed with clean polishing cloth and then with cotton.

After polishing, the rings are dismounted from the polishing spud and cleaned by serial immersion in 1) technical-grade hexane, 2) acetone, 3) Alconox solution (followed by tap-water rinse), 4) deionized water and 5) reagent-grade acetone. Paint brushes are used in baths 1 and 3 to clean the entire surface of each ring. Taking care to preserve the order and orientation in which the rings were polished, they are stacked vertically on a wooded dowel to dry. At this time the remaining peices of the anode assembly are thoroughly washed in hexane and deionized water and allowed to dry.

Using a cotton glove, each ring is weighed to within 0.1 mg on an analytical balance and transferred to a second wooden dowel, again preserving order and orientation. The rings are carefully slid onto the Lucite shaft, and the headpiece is threaded into place, snugly compressing the column of rings. No attention is paid to the angular orientation of the rings.

All possible preparations to the flow system are made before the electrolytic pretreatment step. The flow system is cleaned by filling the reservoir with deionized water and circulating to all parts of the system. Each time the system is drained, the lines are detached at the pump inlet and at the filter, and a suction hose is used to remove the last fluid in the reservoir. The rinse and drain steps are repeated. Two hours before run time, the cryostat is turned on and the methanol coolant is brought to -10 °C. An hour before the run, the reservoir is charged with electrolyte and this is circulated through the recycle loops of the flow system. Coolant flow is started, the temperature-control system is turned on, and the system is allowed to come to temperature. The floor is detached from the cell body and a stream of nitrogen is supplied to flush out the cell.

For the purpose of rendering the anode surface in a uniform, reproducible condition before the experiment, an electrolytic pretreatment step is performed. Effort is made to minimize the time interval between this pretreatment and the beginning of the run, and, during this stage, to protect the surface from exposure to ambient oxygen. The pretreatment vessel is charged to the fill line with 60-percent sulfuric acid. Electrical connections are made to the high-current power supply. Both rinse columns are filled with deionized water and sparged with nitrogen for 15 minutes before pretreatment. Using a special positioning jig, the anode spindle is raised into position in the cell and threaded into the rotator shaft. The motor is turned on and the shaft is spun at 30 rpm to ensure circumferential uniformity of current density.

When all preparations have been made, the pretreatment vessel is raised into position and the electrolyte level is adjusted. (If the starting level of acid in the pretreatment vessel has been judicious, no adjustment is necessary.) This is done by adding or removing fluid in the clear side column until the meniscus coincides with a mark level with the top edge of the anode surface. If the level is too low, part of the anode does not receive pretreatment. If the level is too high, there is an excessive pathway for current to the upper edge of the anode; the local current density will be higher there, and any weight change of the anode can no longer be assumed to be uniformly distributed in the axial direction. (It is impossible to weigh the rings after installation, so it is important to be able to accurately estimate the weight change of each ring during pretreatment.) The power supply is turned on for the preset time interval, which is measured to within one second. In most runs, an anodic pretreatment was performed at 100 mA/cm^2 for 60 seconds. In other instances an equivalent cathodic current was passed, either for 60 or 10 seconds. Once the current has been turned off, the pretreatment vessel is lowered and quickly replaced with the first rinse column. There is a

sufficient back pressure of nitrogen maintained at the cell head that whenever the cell bottom is open (*i.e.* not occupied by the pretreat vessel or a rinse column) the nitrogen efflux prevents the influx of air. The spindle is rinsed for two minutes at 30 rpm before the second rinse column is exchanged rapidly with the first. Upon removing the second column, the cell floor is quickly replaced and fastened. This done, the electrolyte, which has been circulating, is shunted through the cell. The total time between pretreatment electrolysis and cell filling is usually about five minutes.

In every run, the electrolyte throughput was 55 $\pm$ 5 cm^3/s, corresponding to an axial superficial velocity of 3.6 cm/s in the cell. Once there is electrolyte in the cell, the reference-electrode chamber is also filled as soon as possible. This is done by placing a ground-glass fitting into the tube and pulling a vacuum on the sealed chamber; this draws electrolyte from the cell along the connecting line into the reference chamber. Once the reference electrode is in place and all connections are made between the electrodes and the potentiostat, the cell potential is allowed to reach a steady value. The oscilloscope sweep is triggered at this point to record the approach to equilibrium, which usually takes from 10 to 30 minutes. The sweep duration must be long enough to record the cell current and potential during the entire run. When the potential ceases to drift, the potentiostat is switched from the direct-measurement mode to the null mode and the appropriate potential difference needed to null the cell is applied. Since this open-circuit potential is the mixed potential of a corroding system, it depends on mass transfer and hence on rotation speed. The null measurement is made at the rotation speed of the run, 800 rpm.

At this point, the electrolysis is commenced. The potentiostat is switched to the *control-potential* mode. The function generator, which has been programmed to deliver a potential ramp of preset slope up to the desired steady-state potential difference, is

triggered. Applied potential and cell current are monitored on the oscilloscope. When the desired steady-state potential is reached, the ramp is halted, and this potential is applied for the remainder of the run. Temperature is monitored at five-minute intervals; the reference chamber is kept within 2 °C of the reservoir temperature (in the runs at 3 °C ice is added to the bath). Visual observations of the anode surface are periodically recorded.

At the end of the preset time interval for steady-state electrolysis (usually 70 minutes), the current is shut off. Before stopping the rotation, the open-circuit potential is again recorded. The pump is turned off and the cell is drained. The cell floor is detached, and the anode spindle is removed, rinsed in water, and disassembled. Sometimes the rings are frozen on the Lucite shaft, and they can be more easily removed after immersion in ice water. Each ring is rinsed in water and acetone and allowed to dry before weighing. With the rings stacked on a dowel in their original pattern, the appearance of the anode surface is recorded. The flowcell and reservoir are drained and rinsed with deionized water. The oscilloscope data are dumped to the computer and stored on a floppy disk.

Calculation of average current density from the weight loss from a ring is by straightforward application of Faraday's law, based on the assumption of steady-state oxidation of nickel to Ni^{2+} ion with 100-percent current efficiency. It is impossible to account rigorously for the transient period associated with the initial potential ramp; this is treated by adding half of the ramp time to the duration of the isopotential portion of the run. The weight loss during anodic pretreatment is ascribed to the same reaction and is assumed to occur uniformly over the anode surface. In an independent experiment, the current efficiency for this reaction was measured to be 96 $\pm$ 5 percent.

Microelectrode Polarization Measurements

Two kinds of microelectrodes were used to obtain overpotential curves for Nickel 200 in 2 M H_2SO_4 / 0.5 M $NiSO_4$: a rotating-disk electrode (RDE), and a small rotating-cylinder electrode (SRCE) constructed with a single ring segment from the larger anode assembly.

The Pine ASRP2 rotator with ASR speed control was used for both sets of measurements. The rotating-disk electrode, pictured in Figure 1B-9 was made in our machine shop. As illustrated in Figure 1B-10, the active area is one face of a cylindrical section cut from a piece of the same Nickel-200 stock from which the nickel rings were machined. This was done to ensure that both the RDE polarization measurements and the RCE current-distribution measurements were being made on the same anode material. It is noteworthy, however, that the surfaces of the two electrodes (RDE and RCE) are oriented differently with respect to the direction in which the tubing was drawn during fabrication. The nickel core, 0.117 $\pm$ 0.001 cm in diameter, is thermally shrink-fit into the stainless-steel shaft, 1.78 cm in diameter, with 0.3 cm protruding from the shaft. The lower 7 centimeters of this are cast in epoxy and then machined on the lathe to a diameter of 1.96 cm.

The small disk diameter is desirable in that the current distribution approaches uniformity. By comparison with published data (24), one can estimate that the maximum local deviation from the average current density will be less than 5 percent.

The ohmic resistance of the RDE cell can be approximated as $2/\pi^2\kappa r_o$ (15); at the peak anodic current density, the ohmic drop is less than 2 mV. This can be safely neglected, and when the anode potential is referred to its open-circuit value it approximates the surface overpotential, η.

It is desirable to measure both the polarization behavior and the current

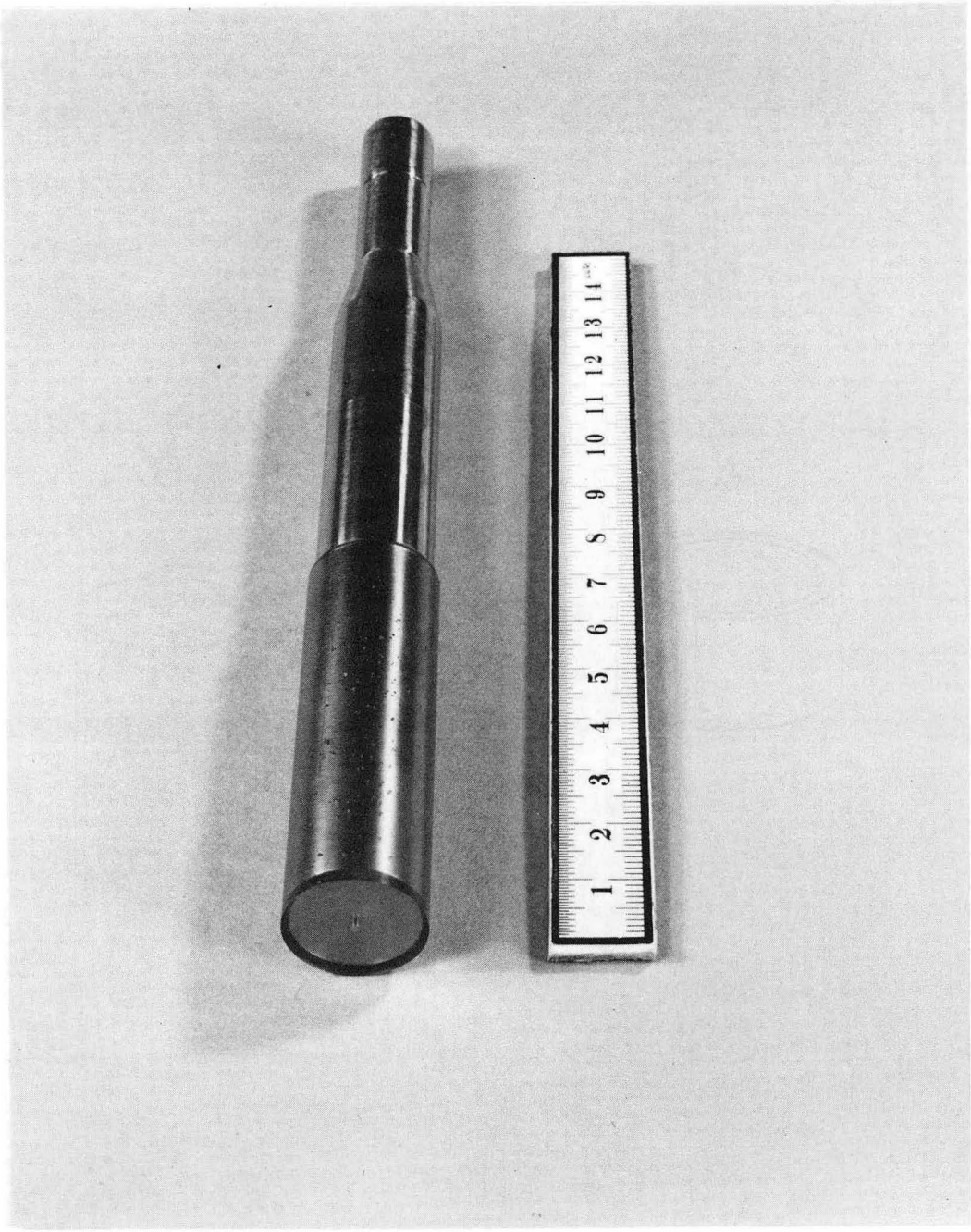


Figure 1B-9. Rotating-disk electrode.

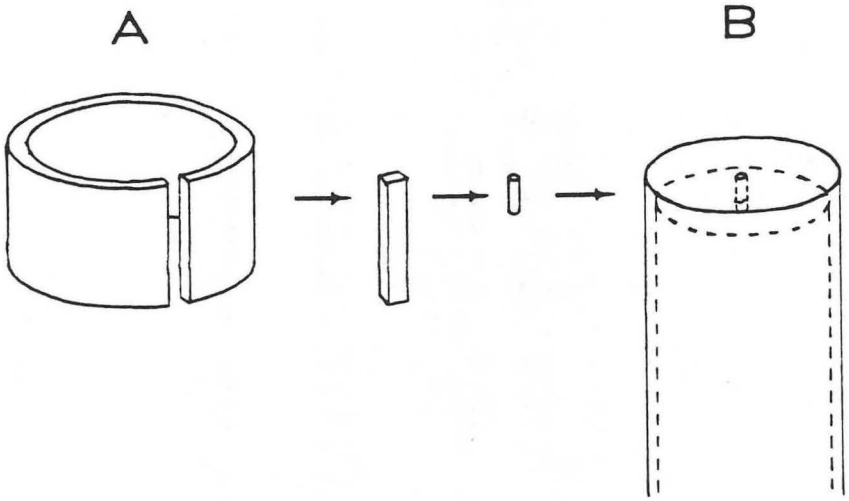


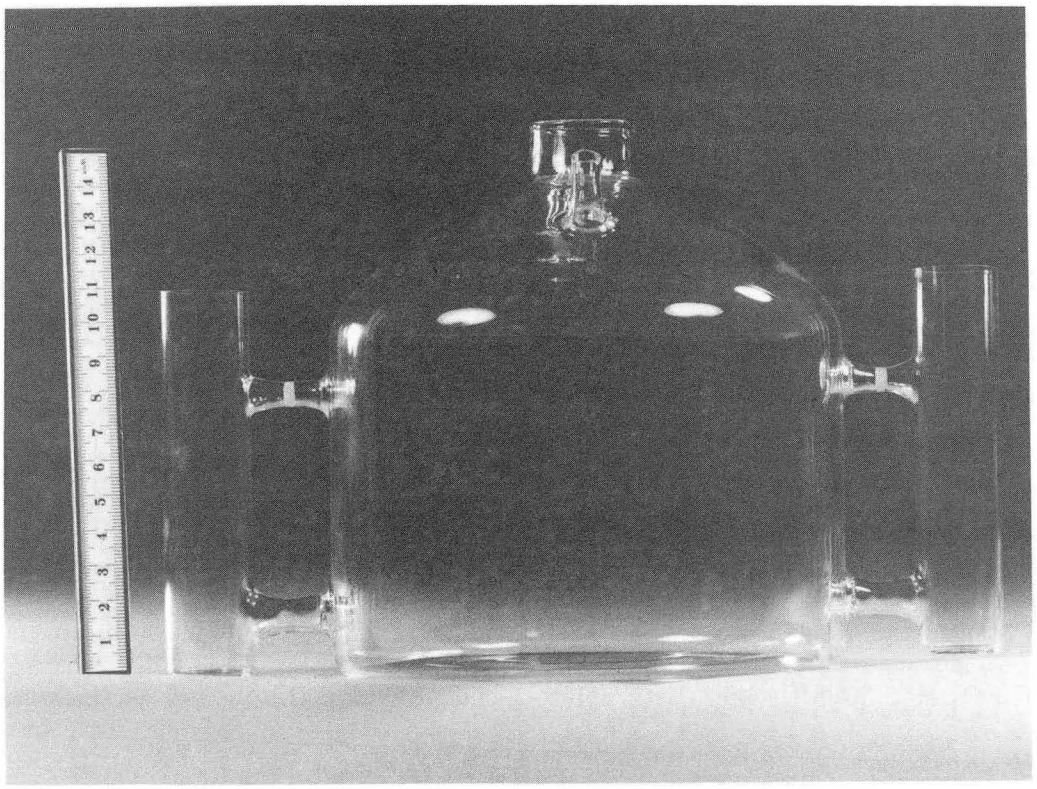
Figure 1B-10. Steps in the fabrication of the rotating-disk electrode **B** from a section of the nickel tubing **A** used to make the rotating-cylinder anode.

distribution at the same condition of mass transfer. The RCE is spun at 800 rpm. According to the correlations for mass transfer to a RCE (25) and to a RDE (26), the disk speed that gives the same limiting current is 1580 rpm.

The glass RDE cell is shown in Figure 1B-11. The area of exposure of the electrolyte to the atmosphere is kept to a minimum, and the chamber is constantly sparged and/or blanketed with nitrogen during operation. The counterelectrode is a platinum-foil strip with an immersed surface area of 30 cm^2. As in the RCE cell, a saturated mercurous sulfate reference electrode is used; this is positioned in a side chamber separated by glass frits from the main compartment. Depending on the desired operating temperature, the glass vessel is either wrapped with heating tape and thermostated with the same controller used in the RCE setup, or partially submerged in an ice bath. Much of the procedure and instrumentation is identical to that used in the RCE experiment.

For pretreatment, the same current-density/time program is applied to the RDE as to the RCE. The RDE is small enough that both the pretreatment and the voltammogram can be performed with a Princeton Applied Research 173 Potentiostat/Galvanostat. The pretreatment is carried out in 20-ml of 60-percent H_2SO_4 in a 50-ml beaker, with a 1-$\times$-5-cm piece of platinum foil as the counter electrode.

Immediately after pretreatment, the RDE is quickly rinsed in the two sparged-H_2O rinse columns used with the RCE and transferred to the RDE chamber which has been sparged with nitrogen for 20 minutes and brought to operating temperature. Upon connection of the three-electrode circuit, the open-circuit potential is monitored and allowed to reach a constant value. From this value, the potential is ramped at 8.6



XBB 869-7740

Figure 1B-11. Cell vessel for rotating-disk electrode.

mV/min using the function generator in conjunction with the PAR 173 potentiostat, taking about three hours to scan to the onset of transpassivity. Temperature is monitored and controlled to within 1 °C. Current-potential data are recorded on the Nicolet 4096 oscilloscope and subsequently transferred to the HP 9825 computer and disk storage.

The second electrode used for polarization measurments is the small-rotating cylinder electrode. This was built to closely approximate the electrode surface condition of the larger RCE cell while preserving a nearly uniform current distribution. The active surface is a single ring segment from the larger anode assembly. The SRCE is shown schematically in Figure 1B-12 and photographed in Figures 1B-13, 14 and 15. The shaft is machined from a bar of aluminum and mounts in the Pine ARCP2 Rotator. The nickel ring, 1.27 cm in height, fits snugly on the bar making electrical contact, and is compressed between a teflon sleeve above and a teflon cap below, which is fastened to the shaft with a teflon bolt (Figure 1B-14). The ring surface is inset 0.18 cm from the cylindrical teflon surface; this prevents edge effects in the primary current distribution. A Lucite frame was fashioned to hold the short annular platinum foil strip, also 1.27 cm in height, in proper axial and concentric alignment to the anode surface (Figure 1B-15). Straight current lines within the interelectrode volume are enforced by a floor and ceiling; these each contain a sparse arrangement of holes to allow the fluid in the interelectrode gap to communicate with the rest of the solution in the vessel. Each run is made with a fresh nickel ring from a single complete set, polished according to the procedure described above. Pretreatment electrolysis is carried out using the same counterelectrode jig, in a different beaker containing 60-percent H_2SO_4. The entire assembly is rinsed twice in deaerated deionized water before immersion in the electrolyte. Again, a saturated-mercurous-sulfate reference electrode is used;

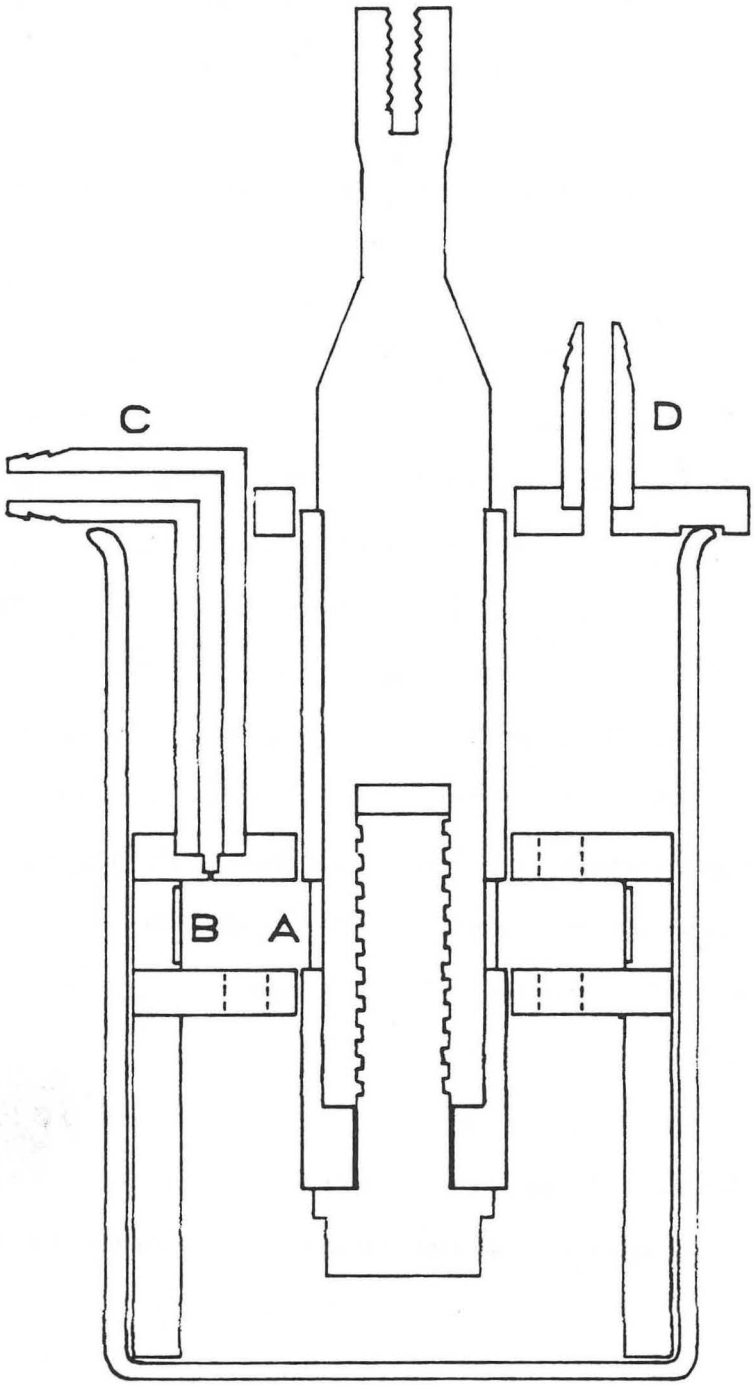
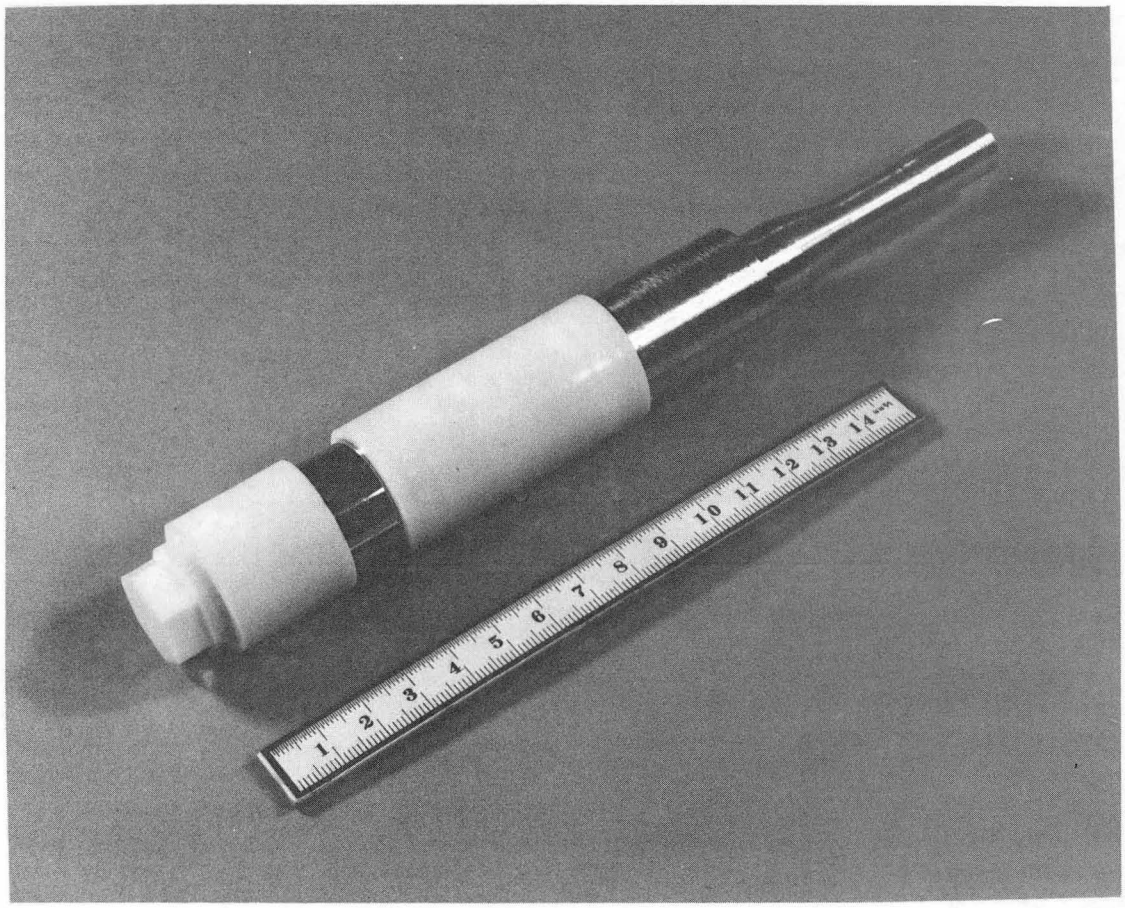
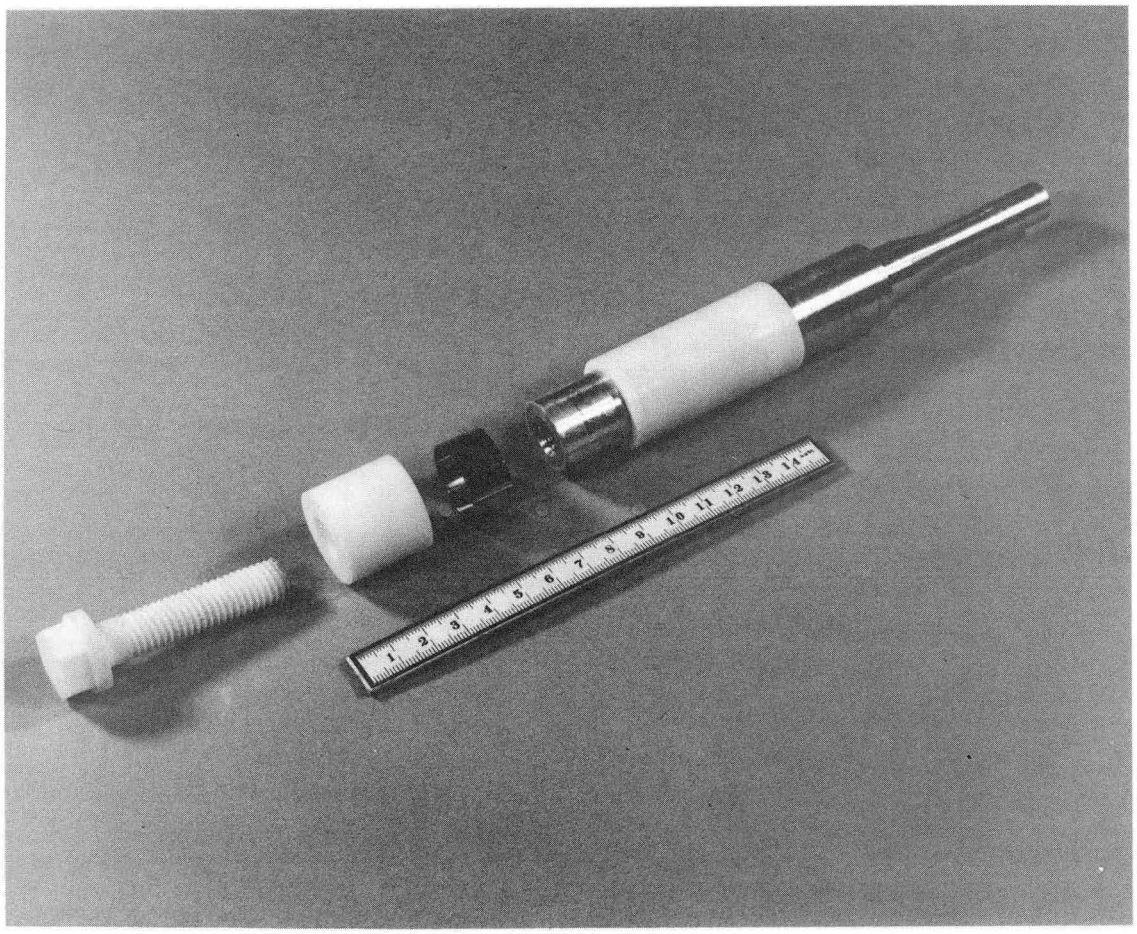


Figure 1B-12. Small rotating-cylinder cell: **A**—anode, **B**—cathode, **C**—Luggin tube, and **D**—shaft



XBB 869-7741

Figure 1B-13. Small rotating-cylinder electrode, assembled



XBB 869-7742

Figure 1B-14. Small rotating-cylinder electrode, disassembled, showing aluminum shaft, nickel ring, teflon cap and screw

XBB 869-7743

Figure 1B-15. Small rotating-cylinder cell

this is housed in a separate chamber connected by a line of electrolyte to an aperture in the ceiling of the interelectrode volume, 2.7 cm from the centerline. The shaft is rotated at the same speed (800 rpm) as the large RCE.

Results and Discussion

Polarization Measurements

Figure 1B-16 shows three polarization curves obtained on the RDE at three different temperatures. In each case a potential ramp of 8.3 mV/min was used. Temperature was varied to find the most advantageous operating condition for the RDE experiment. We chose to perform all but one RCE experiment at 3 °C for two reasons: 1) the broad passive-potential range (high E_P^{max} as defined in Part A) allows a greater range of applied potentials, V_{APP}, for the RCE experiment, and 2) the low active-current-density peak, i_{cr}, allows for a greater range of L_P on our RCE according to the one-dimensional models (Refs. 7-11, Part A). The first series of theoretical current-density profiles presented in Part A was calculated using Curve 1 (Figure 1A-1 of Part A). There is a secondary passivation phenomenon at this low temperature, but this falls outside the potential range of the RCE experiments. While only one voltammogram at 3 °C is shown in Figure 1B-16, five others were recorded under identical conditions, three with anodic pretreatment and two with cathodic pretreatment. Reproducibility was fairly good: the mean value of i_{cr} was 37.6 mA/cm^2 with a standard deviation of 3.6 mA/cm^2 (less than ten percent), and all curves were of similar shape. The curves taken using cathodic pretreatment showed no discernable differences from the others.

Three examples of polarization data measured on the small rotating-cylinder electrode are shown in Figure 1B-17. Each scan was made at 50 mV/min and was

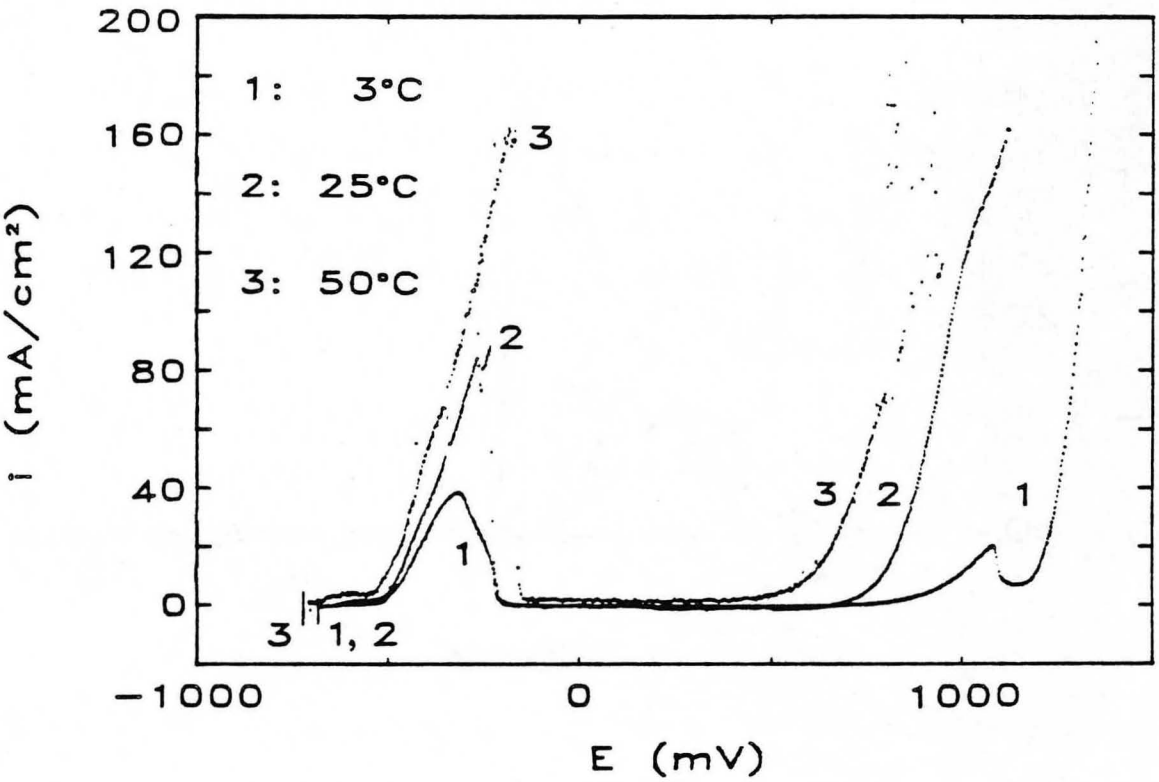


Figure 1B-16. Anodic polarization curves at three temperatures (indicated) for Nickel 200 in 2M H_2SO_4 / 0.5M $NiSO_4$ measured on the rotating-disk electrode at 8.3 mV/min scan and 1580 rpm.

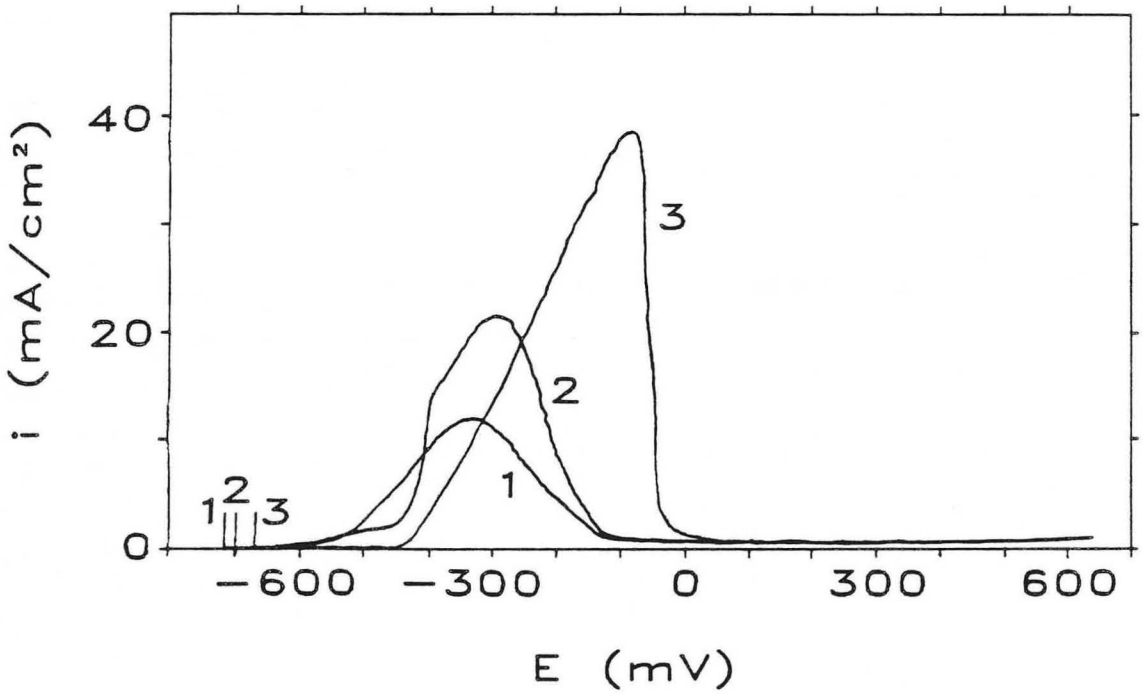


Figure 1B-17. Three anodic polarization curves for Nickel 200 in 2 M H_2SO_4 / 0.5 M $NiSO_4$ measured under the same conditions on the small rotating-cylinder electrode at 50 mV/min scan and 800 rpm.

terminated before the onset of transpassivity. (Comparison to other data taken at 8.3mV/min showed no significant difference.) It is immediately apparent that the repeatability of the measurement is low: i_{cr} ranges from 12.0 to 38.5 mA/cm^2, and there is also variation in shape and in the Flade potential. We do not have a satisfactory explanation for this scatter. However, there are some important differences between the SRCE measurements and the more reproducible RDE measurements. Compared to the RDE, the SRCE has roughly 800 times the surface area to be polished and pretreated. As explained earlier, the two electrode surfaces are oriented differently with respect to the grain structure. Furthermore, the SRCE is prepared for each experiment by a carbide-tool "face cut," which may introduce extra work hardening, and this is followed by unidirectional abrasion and polishing on the lathe. In contrast, the RDE surface is not machined each time, and the abrasion by hand and polishing on the wheel are in random directions.

For the comparison of the experimentally measured to the theoretically predicted current distribution, there is the question of which polarization curve to use in the model. The predictions presented in Part A are based on two polarization curves chosen to represent two extremes in the range of the curves measured: Curve 1 of Figure 1B-16 and Curve 4 of Figure 1B-17 (corresponding to Figures 1A-1 and 1A-8 of Part A).

Current-Distribution Measurements

Table 1A-1 summarizes ten experiments conducted on the rotating-cylinder apparatus. Each experiment was performed with a newly polished set of nickel segments from one of five sets. The cell-current history for one experiment (Run number 2) is shown in Figure 1B-18. The lower box shows the applied-potential program, and the upper one shows the response in cell current. The current rises quickly during the

Table 1A-1: Summary of Rotating-Cylinder Experiments

Run Number	Pretreatment Type	Pretreatment Duration (s)	Ramp Rate (mV/min)	V_{APP} (mV)	Stable A/P Coexistence?
1[1]	anodic	60	333	1000	yes
2	anodic	60	300	1350	yes
3	anodic	60	300	1200	no
4	anodic	60	300	1200	yes
5	anodic cathodic 60	60	300	1275	yes
6	cathodic	10	2400	1200	yes
7	cathodic	10	2400	1200	no
8	cathodic	10	2400	1275	no
9	anodic cathodic	60 300[2]	300	1275	yes
10	anodic	60	300	1000	yes

[1] at 25 °C

[2] at 20 mA/cm^2 in 2M H_2SO_4

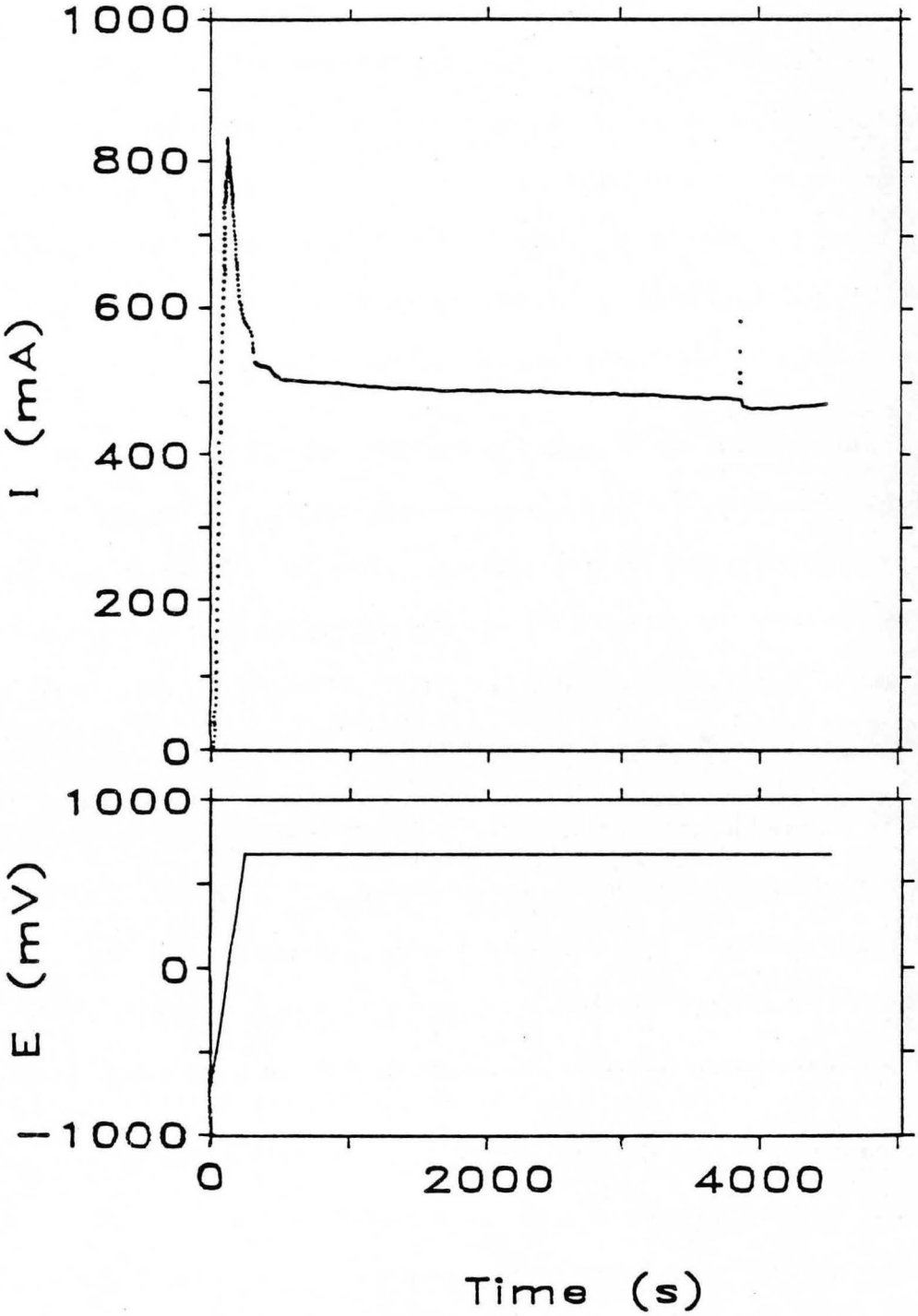


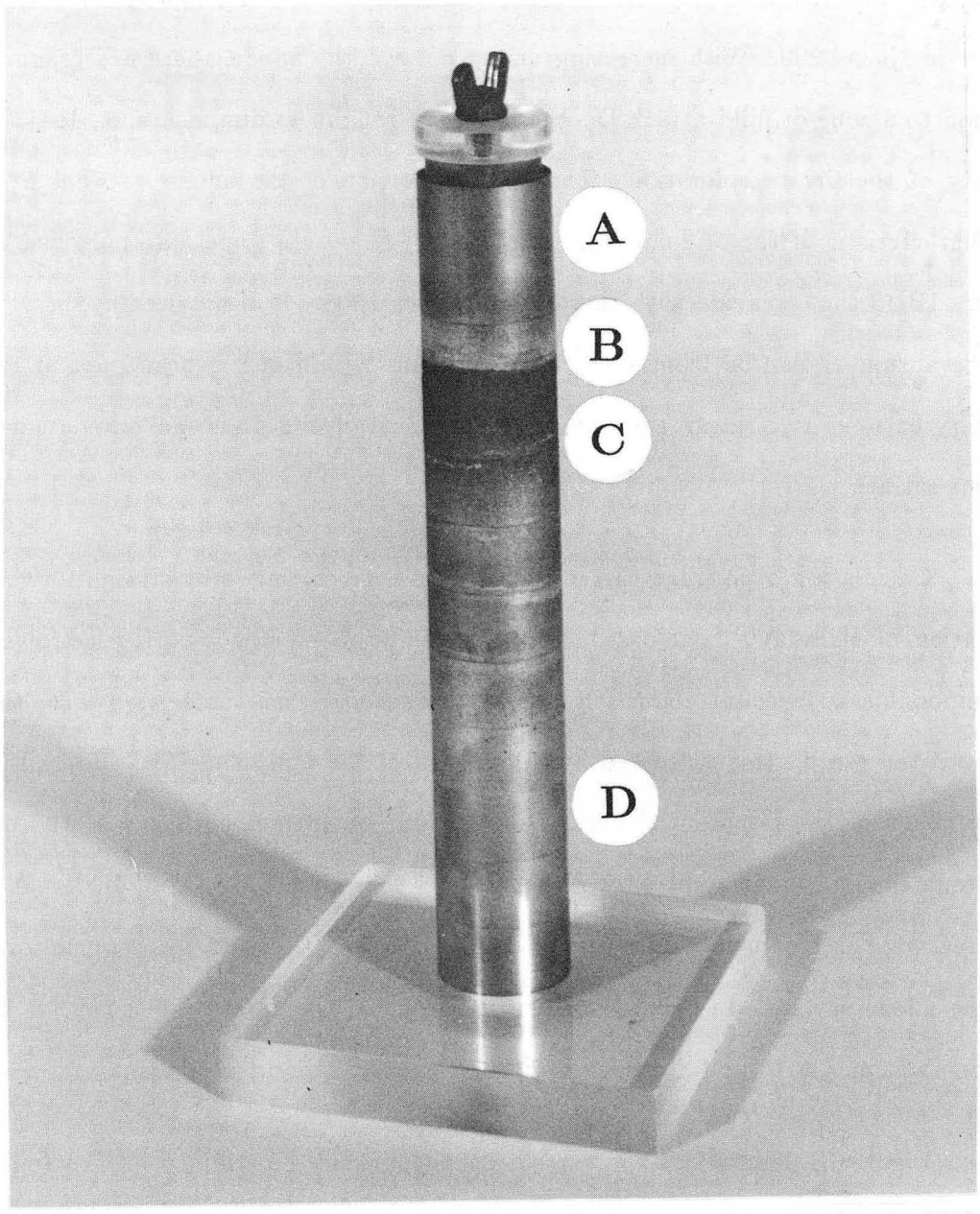
Figure 1B-18. Rotating-cylinder-cell potential program (lower frame) and current response (upper frame).

first part of the ramp, reaches a peak value, and then declines to a nearly steady-state value within about seven minutes. After the initial transient, which outlasts the potential ramp by two to three minutes, the current descends gradually, but does not change by more than 10 percent. The drift could be a consequence of the changing condition of the electrode surface as the dissolution proceeds. It is noteworthy that no oscillatory behavior is apparent on a time scale greater than 4 seconds.

In three experiments (Runs 3, 7 and 8), the behavior was quite different from the that described above. During the potential ramp, the current rose as usual but leveled off at a value lower than normally observed. Then, within five minutes of the start of electrolysis, the current dropped by two orders of magnitude, indicating that the entire electrode surface had reached the passive state. Attempts to reactivate the anode by repeating the potential ramp resulted in even more rapid jumps to total passivity.

The passive film on nickel is known to be on the order of one nanometer in thickness and, of course, invisible. However, after about ten minutes of electrolysis, the border between the regions of passive protection and active dissolution can be discerned. The position of this border can be traced over the course of the run and, in general, is seen to move about 0.3 cm downward from beginning to end.

Figure 1B-19 shows the condition of the anode surface after an experiment (Run 1). Near the top (which was nearest the cathode) there is a zone of passive protection **A**, where the surface has retained some of its original lustre. Below this is the active-passive transition **B**: a speckled region, silver in color, which changes abruptly into a zone of heavy attack **C**. Here the surface is brown in color and appears rough and pitted. Magnification reveals that the roughness corresponds to crystal grains etched to different depths. The grains are axially elongated, reflecting that the nickel tubing was



XBB 869-7744

Figure 1B-19. Nickel rings after a 70-minute experiment:
A—zone of passive protection,
B—active-passive transition,
C— zone of heavy attack,
and **D**—zone of mild attack.

drawn in fabrication. The brown color is probably iron oxide, as iron is a major impurity in Nickel 200. With increasing distance down the anode, there is a gradual transition to a zone of mild attack **D**: the degree of roughness diminishes, as does the intensity of the brown coloration. The bottom portion of the anode also has a shininess characteristic of the original polished surface. There are some markings visible in Figure 1B-19 that coincide with the joints between rings. It is noteworthy, however, that the abrupt transition from silver to brown falls within one ring and not at the boundary between two rings; this is a sign that the segments do in fact simulate a continuous surface.

Calculation of average current density on a given ring is by straightforward application of Faraday's law, based on the assumption of steady-state electrolysis. It is impossible to account rigorously for the transient period associated with the initial potential ramp; this is treated by taking half of the ramp time and adding this to the duration of the isopotential portion of the run. A current efficiency of 100 percent is assumed, for oxidation of pure nickel to Ni^{2+} ion. The weight loss during anodic pretreatment is assumed to correspond to the same reaction and to occur uniformly over the anode surface. In an independent experiment, the current efficiency for this reaction was measured at 100 percent, within 2 percent experimental error.

Figure 1B-20 contains the current-density profiles determined in Experiments 6, 4, 5 and 2. Current density is plotted *versus* the distance along the anode, measured from the edge near the cathode to the opposite edge 15.24 cm away. Panel A of this figure corresponds to Experiment 6, with $V_{APP} = 1200$ mV. The steplike character of the profile reflects the segmenting pattern of the anode: in this example, the shorter rings were positioned between 5.7 and 10.2 cm. The profile agrees qualitatively with expectations: the region nearest the cathode displays a relatively low current density

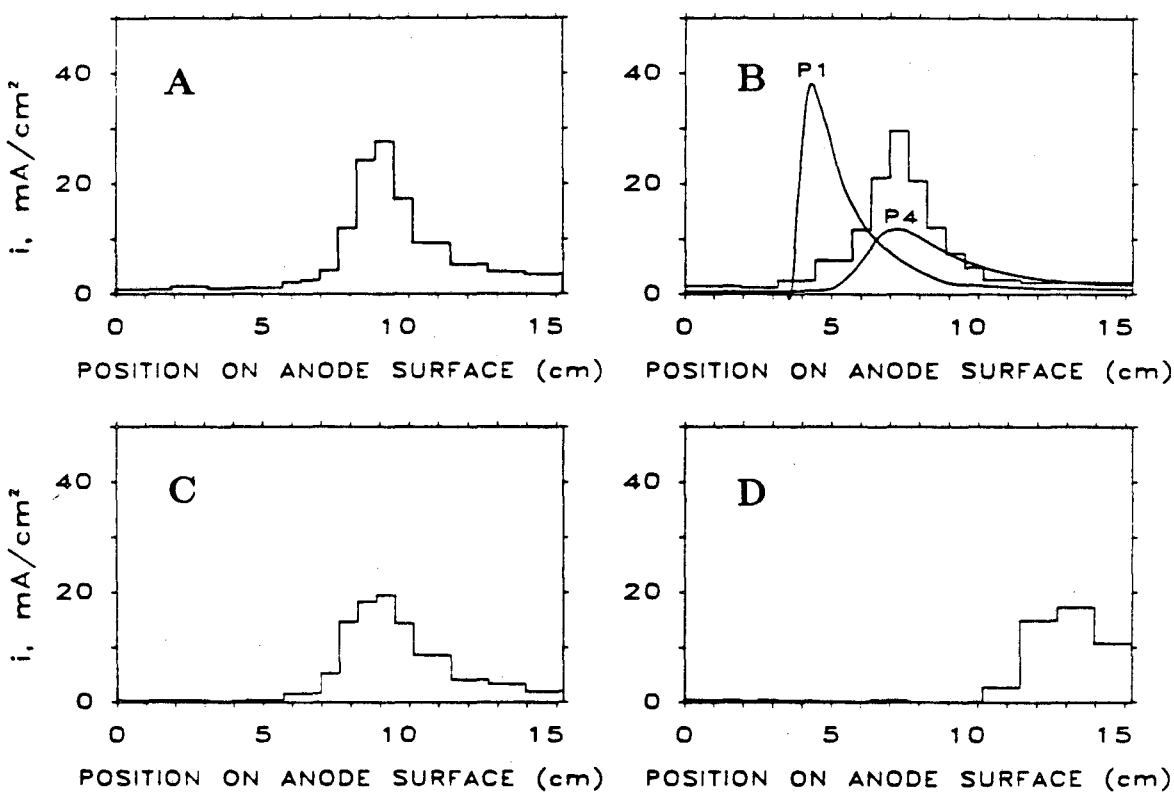


Figure 1B-20. Experimentally deterimined current-density profiles:
A—Experiment 6, $V_{APP} = 1200$ mV;
B—Experiment 4, $V_{APP} = 1200$ mV, with
theoretical predictions based on
two sets of polarization data;
C—Experiment 5, $V_{APP} = 1275$ mV; and
D—Experiment 2, $V_{APP} = 1350$ mV.

characteristic of the passive state; there is a transition to higher, active current density, which reaches a maximum and then decays with distance along the anode. The gross features of the distribution are adequately resolved by the segmented-electrode technique.

The experiment was designed to measure the steady-state current-density profile. However, as we have discussed, the cell current does drift slightly, and the observed active-passive transition migrates a short distance during the experiment. Thus it is possible that the profiles shown in Figure 1B-20 contain some "blurring," *i.e.* the current distribution may have shifted slightly during the experiment, the resulting weight-loss distribution reflecting an average of this course.

Figure 1B-20B shows the distribution recorded in Experiment 4, performed at the same conditions. This profile differs slightly from that in Panel A, most ostensibly in the position of the current-density maximum, but also in the steepness of the active-passive transition. The comparison gives an indication of the reproducibility of the experiment, although a more complete evaluation is made later from Figure 1B-21. Plotted for comparison on Panel B are two theoretical predictions of the current distribution, smooth curves P1 and P4, based on Polarization Curves 1 and 4. It is apparent that neither prediction matches well with the experimental data of Panel B or of Panel A. While the total integrated currents and the current-density maxima match P1 more closely, the positions of these maxima and the slopes of the active-passive transition agree better with P4. The current densities experimentally determined in the passive zone exceed those of both P1 and P4.

Panels C and D of Figure 1B-20 display measured profiles corresponding to Experiments 5 and 2 (V_{APP} = 1275 and 1350 mV). The passive current densities agree

better with the predictions than in panels A and B. The peak current densities lie between those of Polarization Curves 1 and 4. However, in both examples C and D, L_P exceeds the value predicted using either Curve 1 or Curve 4.

Finally, we present Figure 1B-21, a comparison of all of the current-distribution measurements with the theoretical calculations, in terms of the length passivated, L_P. Each open dot represents an experiment; the number labels correspond to Table 1. Experiment 1 was conducted at 25 °C and should be compared to the dotted line segment. Not appearing on the plot are the three experiments, 3, 7 and 8, in which the entire anode became passive within five minutes. It should be remembered that, although we have striven to promote the *partly* passive condition, the *fully* passive condition does correspond to a valid solution of the mathematical model. Moreover, the occurrence of complete passivation is by no means undesirable in applications of anodic protection. It is tempting to attribute the scatter in the measured L_P *–versus–* V_{APP} data to the scatter in the measured polarization behavior. Figure 1B-21 indicates that such a claim may be partly, though not entirely, valid. The points for Experiments 2, 9 and 10 lie well outside the envelope outlined by curves D1 and D4. Despite the low reproducibility of the RCE experiments, we observe without exception that the measured L_P values exceed the model's predictions. We do not have a satisfactory explanation for this. From the standpoint of anodic protection, the model seems to provide a consistent "worst case" estimate of throwing power. This work suggests that a large obstacle to successful application of the model will be the uncertainty in determining the active-passive kinetic data, and, in turn, the uncertainty in the initial condition of the anode surface. The large variability of active-passive kinetic data and the high sensitivity to the initial condition of the anode surface are perhaps largely responsible for the slow acceptance of anodic protection in the corrosion-control industry.

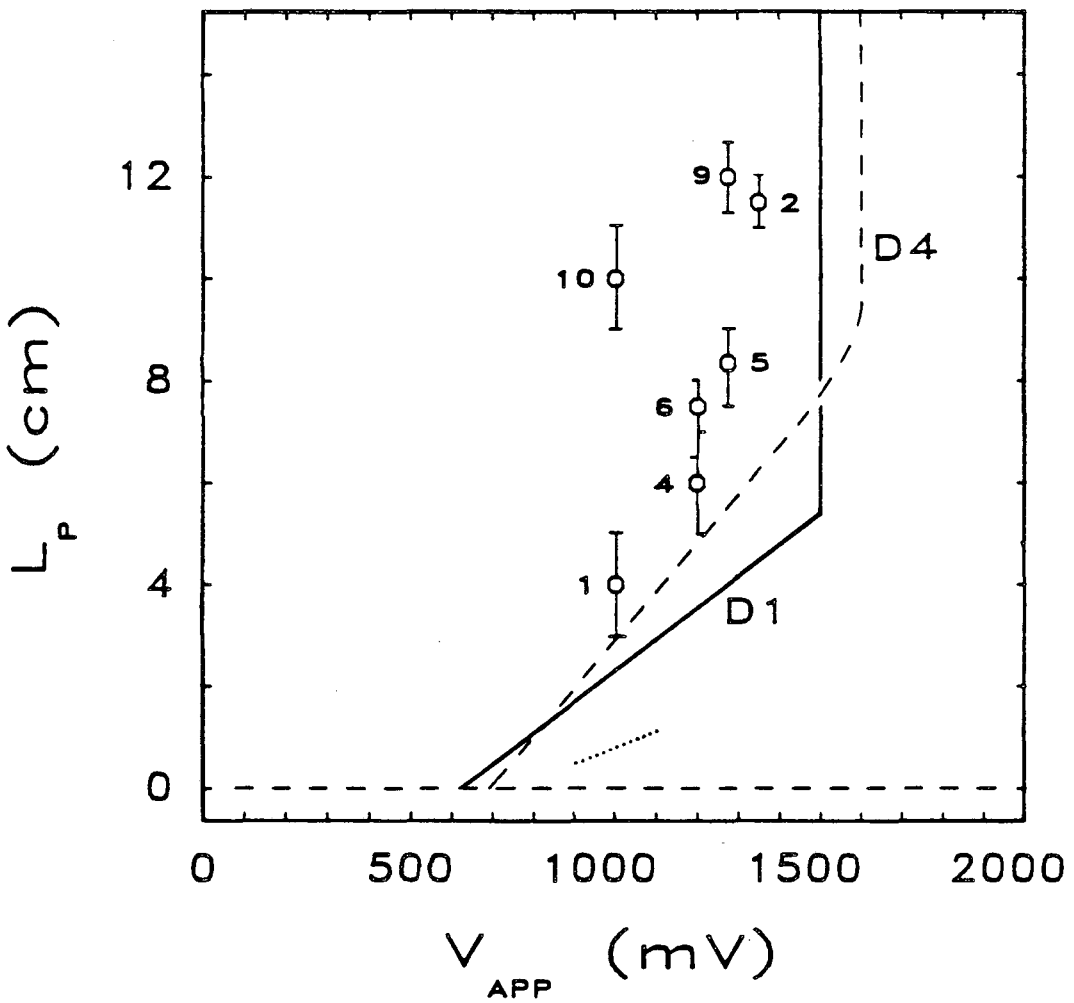


Figure 1B-21. The length of the passive zone *vs.* the applied cell potential: experimental measurements (open dots) compared with theoretical predictions based: on Polarization Curve 1 (solid curve, D1), on Polarization Curve 4 (dashed curve, D4), and on Polarization Curve 2 (dotted line segment). Temperature in all cases is 3 °C except for Point 1 and the dotted line segment, which are at 25 °C.

Conclusion

An experimental study of an active-passive cell with nonuniform current distribution has been performed. The condition of partial passivity was successfully produced on the anode, and the steady-state current distribution was determined with good spatial resolution by the segmented-electrode, weight-loss technique. There was limited reproducibility in the experiment, very likely due to variability in kinetic behavior, as evidenced by the scatter in the polarization measurements. This variability, in turn, probably stems from a high sensitivity to the initial condition of the anode surface and the difficulty of preparing the anode surface in a reproducible manner. A more fastidious program of experiments might give more reproducible results, but the applicability of these results to systems of practical interest, in which the anode surface cannot be methodically prepared, is uncertain. The scatter in the current-distribution measurements precludes a conclusive evaluation of the theoretical model of Part A. However, in every instance, the extent of the passive zone measured in the experiment exceeded the prediction by the model. Accordingly, the experiments suggest that the model provides a conservative estimate of throwing power.

References

1. L. L. Shreir, ed., "Corrosion," Newnes-Butterworths, Boston, 1976.

2. M. G. Fontana and N. D. Greene, "Corrosion Engineering," 2nd ed., McGraw-Hill, New York, 1978.

3. O. L. Riggs and C. E. Locke, "Anodic Protection: Theory and Practice in the Prevention of Corrosion," Plenum Press, New York, 1981.

4. R. C. Alkire and D. B. Reiser, *Electrochimica Acta*, **28**, 1309 (1983).

5. R. Sautebin and D. Landolt, *J. Electrochem. Soc.*, **129**, 946 (1982).

6. G. A. Prentice and C. W. Tobias, . Electrochem. Soc., **129**, 316 (1982).

7. A. Tvarusko, . Electrochem. Soc., **121**, 661 (1974).

8. W. Engelmaier, T. Kessler, and R. Alkire, *J. Electrochem. Soc.,* **125**, (1978).

9. Y. Awakura, A. Ebata, and Y. Kondo, *J. Electrochem. Soc.,* **126**, 23 (1979). R. H. Rousselot, Proceedings of "Surface 66," Basel, p. 39, 1966.

10. H. Y. Cheh, *J. Electrochem. Soc.,* **117**, 609 (1970).

11. F. L. LaQue, *Corrosion,* **13**, 303t (1957).

12. C. G. Law, Jr. and J. Newman, *J. Electrochem. Soc.,* **133**, 37 (1986).

13. I. Epelboin, C. Gabrielli, M. Keddam, J. C. Lestrade and H. Takenouti, *J. Electrochem. Soc.,* **126**, 1632 (1972).

14. P. P. Russell and J. Newman, *J. Electrochem. Soc.,* **130**, 547 (1983).

15. C. G. Law, Jr. and J. Newman, *J. Electrochem. Soc.,* **126**, 2150 (1979).

16. C. Edeleanu and J. G. Gibson, *Chemistry and Industry,* **10**, 301 (1961).

17. M. N. Fokin and V. A. Timonin, *Dokl. AN SSSR,* **164**, 150 (1965).

18. W. A. Mueller, *J. Electrochem. Soc.,* **110**, 698 (1963).

19. V. A. Makarov, Ya. M. Kolotyrkin, V. M. Knyazheva, and E. B. Mamin, *Zashchita Metallov,* **1**, 662 (1965).

20. V. A. Timonin and M. N. Fokin, *Zashchita Metallov,* **2**, 307 (1966).

21. B. MacDougall and M. Cohen, *J. Electrochem. Soc.,* **123**, 1783 (1976).

22. N. Sato and G. Okamoto, *J. Electrochem. Soc.,* **110**, 605 (1963).

23. T. S. de Gromoboy and L. L. Shreir, *Electrochimica Acta,* **11**, 895 (1966).

24. J. S. Newman, "Electrochemical Systems," p. 347, Prentice Hall, Englewood Cliffs, N.J. (1973).

25. M. Eisenberg, C. W. Tobias, and C. R. Wilke, *J. Electrochem. Soc.,* **101**, 306 (1954).

26. B. Levich, *Acta Physicochim. URSS*, **17**, 257 (1942); **19**, 117 (1944).

LIST OF SYMBOLS

E_P^{max}	maximum potential range that can be spanned in the passive regime without incurring transpassivity, mV
i	current density, mA/cm^2
i_{cr}	peak active current density on polarization curve, mA/cm^2
i_p	current density in passive potential range, mA/cm^2
I	cell current, mA
L_P	length of the passive zone, cm
r_o	radius of rotating-disk electrode, cm
V_{APP}	voltage between anode and cathode in excess of the open-circuit value, mV
κ	electrolyte conductivity, $ohm^{-1}cm^{-1}$
η	surface overpotential, mV

Chapter 2

The Influence of Attached Bubbles on Potential Drop and Current Distribution at Gas-Evolving Electrodes

ABSTRACT

A theoretical study is presented of the effects of bubbles attached to the surface of a gas-evolving electrode, with emphasis on their influence on the local current distribution and on the potential drop at the electrode. The mathematical model accounts for the combined influence of: 1) ohmic obstruction within the electrolyte; 2) area masking on the electrode surface, which raises surface overpotential by increasing the effective current density; and 3) decreased local supersaturation, which lowers the concentration overpotential. The electrolytic transport is described by potential theory, and the dissolved gas is assumed to obey steady-state diffusion within a concentration boundary layer. The coupled field equations are solved numerically using the boundary-element method. The model is applied to hydrogen evolution in potassium-hydroxide solution. For gas evolution in the Tafel kinetic regime, the current distribution is nearly uniform over the unmasked electrode area, and the increase in surface overpotential is the dominant voltage effect. However, outside the Tafel regime (eg. on cathodes of greater catalytic activity) the current density is strongly enhanced near the bubble-contact zone, and the supersaturation-lowering effect is quite strong, largely offsetting the ohmic and surface-overpotential effects. Proceeding from a set of base conditions, we perform a systematic examination of attached-bubble effects, their relative importance, and their dependence on system variables.

Introduction

Electrochemical gas evolution continues to occupy a prominent role in the electrolytic industries. The chloralkali industry alone, which consumes nearly two percent of the electric power generated in the U.S., calls for research that can lead to decreased energy losses. Gas-evolving cells are characterized by the abundance and complicated behavior of electrolytic bubbles, which give rise to numerous effects on the electrode process and on cell performance. In studies aimed at raising the level of fundamental understanding of gas-evolution, it is helpful to examine these effects individually. While such isolation is often difficult to achieve experimentally, theoretical analysis can sometimes serve as a valuable probe, identifying trends, ranking the importance of competing effects, and suggesting improvements. The scope of this paper is restricted to those bubbles that are attached to the surface of a gas-evolving electrode, and we use a theoretical model to examine their influence on the reaction. We further restrict our inquiry to voltage effects: the influence of attached bubbles on ohmic drop and on electrode polarization.

Much of the literature on bubble-induced ohmic drop deals with the increase in bulk-electrolyte resistivity caused by dispersed bubbles. These studies are reviewed by Meredith and Tobias (1) and more recently by Vogt (2). The major relations that have won acceptance are Maxwell's equation (3), the Bruggeman equation (4), and an equation developed by Meredith and Tobias (1,5).

There is also a growing body of literature on the thin layer of electrolyte close to the electrode surface. This layer is known to be more crowded with bubbles than the bulk electrolyte, and the contribution of this "bubble curtain" to the total cell resistance has received special attention. Efforts to characterize this layer and its voltage effect have been both experimental and theoretical. Janssen and Barendrecht (6) have

measured the relative increase in surface-layer resistance at a rotating-disk electrode using AC-impedance methods. In a particularly illuminating study, Bongenaar-Schlenter *et al.* (7) used a microscopic technique to determine the density of bubbles as a function of distance from the electrode surface. They report two regions: a "layer adjacent to the electrode, crowded with bubbles," and a second region where "the bubble population is much lower." The crowded layer is at least several average bubble diameters in thickness and features a gas void fraction that drops nearly linearly with distance. These authors proceed to advance a resistance model which is based on the Bruggeman equation and takes into account the spatial variation of void fraction.

Other authors (6,8,9) have estimated the increment in resistance due to the surface layer of bubbles using bulk-dispersion models such as the Bruggeman equation. These treatments generally assume a bubble-layer thickness of approximately one average bubble diameter. Comparison with the analogue experiments done by Sides and Tobias (10) shows that these bulk-dispersion models are relatively successful at predicting the ohmic effect of attached bubbles in the absence of polarization (even though, as Sides and Tobias (11) point out, attached bubbles offer less obstruction than free bubbles since the field disturbance due to an attached bubble is truncated by the isopotential electrode surface).

Another approach is the constriction model of Sides and Tobias (10), in which, at any plane parallel to the electrode surface, current density through the "unvoided" electrolyte is taken as uniform. Lanzi and Savinell (12) present a modification of this approach wherein the electrolyte among the attached bubbles is taken as a three-dimensional dispersion of smaller bubbles.

This paper will focus entirely on those bubbles which are actually attached to the

electrode surface, or "fixed" as opposed to "free." We carry out a separate treatment of these fixed bubbles, not because they are more crowded (since we know that the crowding extends beyond the fixed layer), or because their ohmic effect is significantly different from that of the free bubbles (the successful application of bulk-dispersion and constriction models to the fixed layer suggests that this is not so). Rather, the main reason for devoting special attention to attached bubbles is that they exert two other voltage effects beyond ohmic considerations: they increase the surface overpotential by masking portions of the electrode and raising the current density over the remaining area, and they change the concentration overpotential by influencing the level of gas supersaturation at the electrode surface. Accordingly, the attached bubbles can be expected to exert a disproportionately large effect on cell voltage.

The two overpotential effects have received little attention in the literature. The kinetic effect is given cursory mention by Hine *et al.* (13) and Lanzi *et al.* (12) and is treated in slightly more depth by Hine *et al.*(14). To our knowledge, only Sides (15) has described a "total voltage penalty associated with the presence of bubbles," which includes all three components (ohmic, kinetic, and concentration). He circumvents analysis of distributed effects by treating spatial averages of surface and concentration overpotential, and defines "hyperpolarization" as the excess surface overpotential attributable to the presence of bubbles. He measures this quantity in an analogue experiment by a current-interruption technique.

A rigorous theoretical description of the electrical effects of attached bubbles must account for the nonuniform distribution of current density and gas supersaturation. Since this requires solution of difficult field problems in two and three dimensions, few distributed models have been put forth. Sides and Tobias (11) give an exact solution of the potential field around an isolated bubble tangent to an electrode surface.

neglecting polarization effects. From this result, they proceed to calculate the resistance increment due to a sparse array of such bubbles. Wilson and Hulme (16) extend this analysis to bubbles of arbitrary contact angle. In addition, these authors solve for gas supersaturation, which, they acknowledge, influences the electrode potential; however, they do not account for this concentration overpotential in coupling the potential field to the concentration field, and moreover, they employ an unrealistic boundary condition away from the electrode.

In the present analysis, we calculate the influence of attached bubbles on the potential drop at a gas-evolving electrode, including both surface- and concentration-polarization effects. The procedure is to solve numerically for the potential and gas-concentration fields around a single attached bubble, and then to extend this result to a pattern of bubbles on the surface. The model is first developed in general terms, and then discussed in the context of a specific example, namely hydrogen evolution under conditions typical of a membrane-type chloralkali cell.

The Model

Reported observations of gas evolution on the microscopic scale reveal that the electrode surface is the site of frequent nucleation, growth, and detachment of bubbles. At any instant in time, there are many attached bubbles in different stages of growth, randomly arranged on the surface.

Our first step in constructing a tractable model for predicting the voltage effects of attached bubbles is to restrict our attention to one point in time, and to solve the transport problem prevailing at this instant. We make the key assumption that the transport phenomena are pseudosteady-state with respect to bubble growth. The validity and consequences of this assumption are discussed in detail later in this section.

A second idealization is to describe the assortment of attached bubbles as a mono-sized array: all bubbles are taken to be spherical, each having the same diameter and the same contact angle with the electrode surface. Moreover, the geometric arrangement of bubbles on the surface is idealized as a regular hexagonal array. This allows us to solve the field problem for one bubble in its hexagonal unit cell, and to extend this solution by symmetry to the entire array of bubbles. (This idealization is used by Sides (10).)

In a further simplification, the hexagonal unit cell is replaced by a cylinder of the same projected area. This "equal-area-cylinder" approximation, illustrated in Figure 2-1, reduces a three-dimensional problem to an axisymmetric one. (The validity of this approximation will be addressed later in the discussion of Figure 2-6.)

The boundary-value problem corresponding to the physical model chosen to describe the vicinity of a single attached bubble is stated in Table 1, which refers to Figure 2-2. The next four paragraphs give a physical interpretation of this problem statement.

The electrolytic transport can be described by potential theory (17) since, in most important instances of gas evolution, the electrolyte concentration is high and does not vary significantly, even within the mass-transfer boundary layer. Thus the potential obeys the Laplace equation throughout the electrolyte (Eq. 1). (The fact that we have restricted our attention to a single instant in the course of the bubble's growth does not introduce any error at this point, since the potential field and the electrical double layer relax much more quickly than the bubble diameter increases.)

The gradient in potential can have no component crossing either the axis of symmetry Γ_3, the bubble surface Γ_2, or the cylindrical boundary Γ_5 (Eq. 2). Far from the

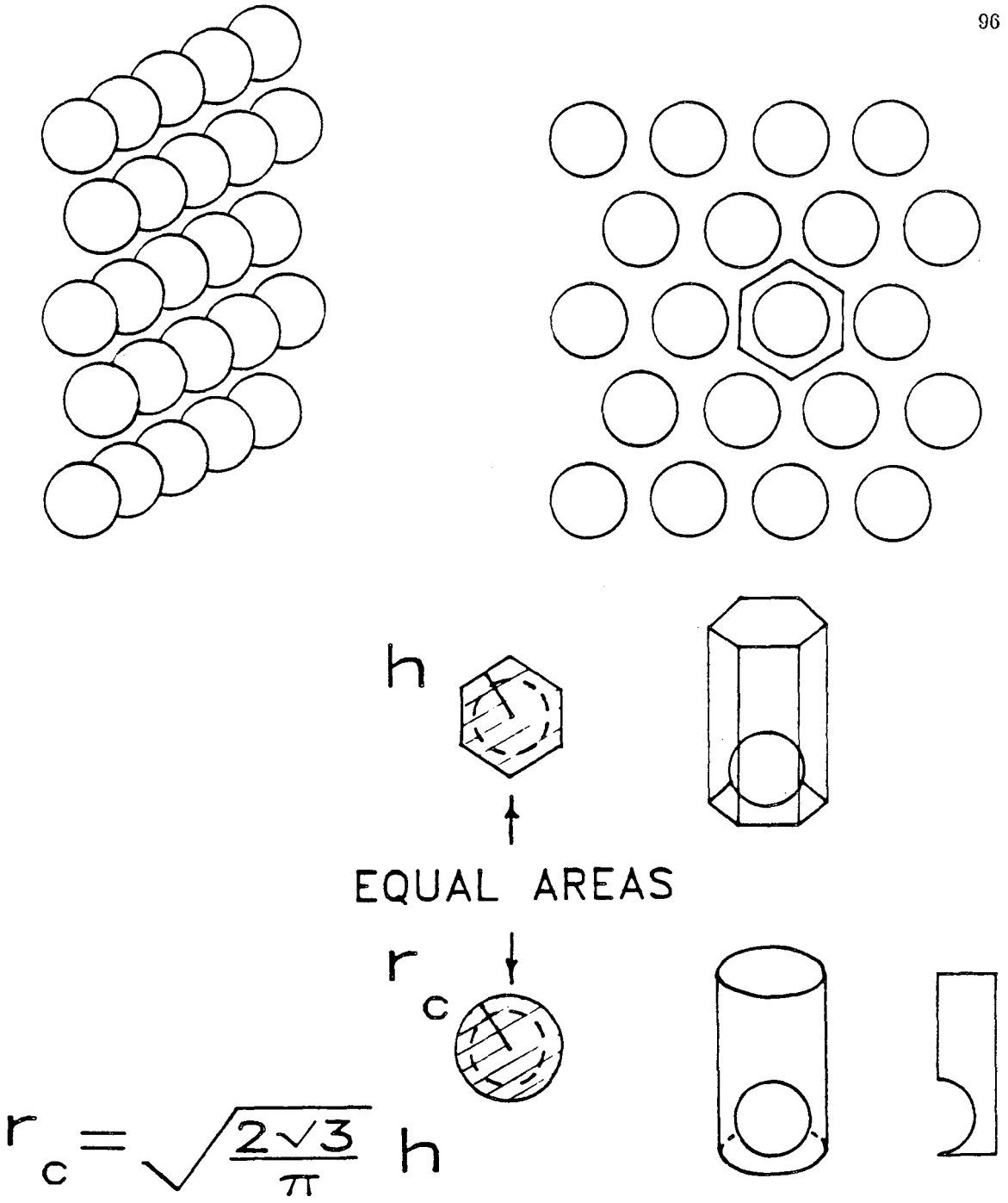


Figure 2-1. Illustration of the equal-area-cylinder approximation for the symmetry condition of a bubble in a monosized hexagonal array.

Table 1: Problem Statement

Dimensional Form	Dimensionless Form	Domain or Boundary	Eq.
$\nabla^2\phi = 0$	$\nabla^{2*}\phi^* = 0$	Ω_A	(1)
$\frac{\partial\phi}{\partial\mathbf{n}} = 0$	$\frac{\partial\phi^*}{\partial\mathbf{n}^*} = 0$	$\Gamma_2, \Gamma_3, \Gamma_5$	(2)
$\frac{\partial\phi}{\partial z} = -\kappa\, i_{AVE}$	$\frac{\partial\phi^*}{\partial z^*} = -\delta$	Γ_4	(3)
$\nabla^2 c_g = 0$	$\nabla^{2*} c_g^* = 0$	Ω_B	(4)
$\frac{\partial c_g}{\partial\mathbf{n}} = 0$	$\frac{\partial c_g^*}{\partial\mathbf{n}^*} = 0$	Γ_{3B}, Γ_{5B}	(5)
$c_g = c_g^{SAT}$	$c_g^* = 1$	Γ_2	(6)
$c_g = c_g^{BULK}$	$c_g^* = \frac{c_g^{BULK}}{c_g^{SAT}}$	Γ_6	(7)
$\frac{\partial c_g}{\partial z} = -\frac{s_g\ \kappa}{n\ F\ Dg}\frac{\partial\phi}{\partial z}$	$\frac{\partial c_g^*}{\partial z} = -\Psi\frac{\partial\phi^*}{\partial z^*}$	Γ_1	(8)
$\phi = -\eta,$	$\phi^* = -\eta^*;$ where	Γ_1 where	(9)
$-\frac{\kappa}{i_o^o}\frac{\partial\phi}{\partial z} = c_g^{s_g} e^{\frac{\alpha_a F}{RT}\eta} - e^{-\frac{\alpha_c F}{RT}\eta}$	$-\frac{1}{J}\frac{\partial\phi^*}{\partial z^*} = c_g^{*\,s_g} e^{\alpha_a\eta^*} - e^{-\alpha_c\eta^*}$		(10)

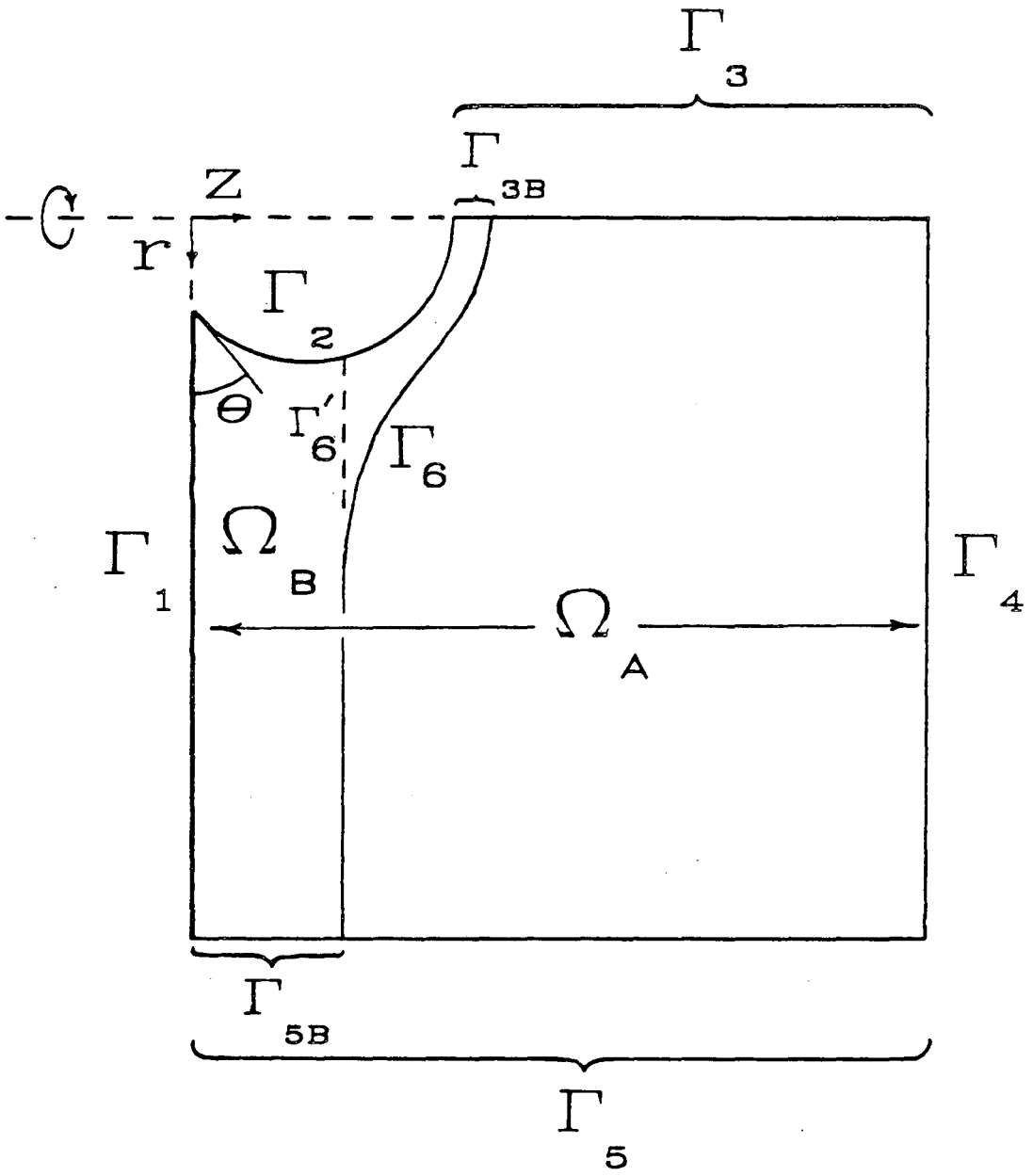


Figure 2-2. Geometric configuration of the dual boundary-value problem.

electrode, the potential gradient is fixed (Eq. 3) at the value corresponding to the average current density applied at the electrode, i_{AVE}, according to Ohm's law:

$$\nabla\phi = -\frac{i}{\kappa}. \tag{11}$$

At the electrode surface Γ_1, the potential of the metal electrode differs from the potential in solution by a total overpotential η, which obeys the relation:

$$\frac{i}{i_o^o} = \left(\frac{c_g}{c_g^{SAT}}\right)^{s_g} e^{\frac{\alpha_a F}{RT}\eta} - e^{-\frac{\alpha_c F}{RT}\eta}. \tag{12}$$

Variations in electrolyte concentration are ignored. The total overpotential, η, is the sum of the concentration overpotential,

$$\eta_c = -\frac{s_g}{n}\frac{RT}{F}\ln\frac{c_g}{c_g^{SAT}}, \tag{13}$$

and the surface overpotential η_s, expressed in this study by the Butler-Volmer expression,

$$\frac{i}{i_o} = e^{\frac{\alpha_a F}{RT}\eta_s} - e^{-\frac{\alpha_c F}{RT}\eta_s}, \tag{14}$$

with

$$i_o = i_o^o\left(\frac{c_g}{c_g^{\mathrm{SAT}}}\right)^{s_g\left(1-\frac{\alpha_a}{n}\right)}. \tag{15}$$

(It is assumed that $\alpha_a + \alpha_c = n$.)

The transport of dissolved gas is coupled to the electrolytic transport. This coupling is expressed in the two matching conditions at the electrode surface (Eqs. 8 and 9). These conditions link the two boundary-value problems (concentration and potential). One of these matching conditions, the overpotential expression (Eq. 9), has already been discussed. Perhaps the simplest interpretation of this condition is the

following: a buildup of dissolved gas at the electrode surface tends to promote the reverse reaction and diminish the overall current density; as current density is proportional to the potential gradient, it is clear that the potential problem is coupled to the concentration problem.

The other matching condition is a flux equality (Eq. 8). This states that the rate at which charge is conducted through the electrolyte to the electrode surface is related by stoichiometry to the rate at which dissolved-gas product is formed and carried away by diffusion.

In treating the transport of dissolved gas, we make two important simplifications. The first is to assume that the dissolved gas moves only by diffusion within a stagnant boundary layer. Beyond this Nernst diffusion layer, the solution is taken to be well mixed. The second key simplification, which has already been introduced, is the pseudosteady-state assumption. This allows us to treat a single instant in the lifetime of a growing bubble as a steady-state problem, as if the bubble were not actually changing in size. This approximation holds only if the time constant of bubble growth greatly exceeds the time constant for relaxation of the concentration field around the bubble. One might expect that, since both of these events occur by the same mechanism (diffusion), the two time constants should be on the same order of magnitude. Comparison of the diffusion time constant,

$$\tau_{DIFF} = \frac{a^2}{D_g}, \tag{16}$$

to the age of a bubble growing according to Scriven's model (18) confirms this expectation. It is therefore clear that the pseudosteady-state approximation represents a departure from full rigor.

In summary, neither the stagnant-layer assumption nor the pseudosteady-state assumption is strictly valid, but they are both of great value in simplifying the problem to the point where it can be solved. (The unsimplified problem would require solving the transient convective-diffusion equation, which first requires solving the transient fluid flow around a growing bubble, including proper initial conditions and the influence of gross stirring by free bubbles. The computational effort required for this problem is prohibitive.)

It should be emphasized that the loss of rigor ensuing from these two assumptions affects only those cases in which the effect of supersaturation plays a significant role. As will be shown later, there is a wide and important range of conditions over which supersaturation does not influence the voltage drop. In these instances, the secondary current distribution prevails and, as explained earlier, the pseudosteady-state approximation is fully justifiable.

Finally, it is important to consider that, despite the idealizations made in posing the concentration problem, the model still offers considerable value as: 1) a preliminary investigation of supersaturation effects of attached bubbles, useful for identifying trends and parameter ranges of interest; 2) a valuable limiting-case solution;[†] and 3) a starting point for more sophisticated models.

With the concentration problem idealized to pseudosteady-state diffusion, we can write the Laplace equation for the dissolved-gas concentration within the Nernst diffusion layer (Eq. 4). No gas is allowed to diffuse across the axis of symmetry Γ_{3B} or across the cylindrical boundary Γ_{5B} (Eq. 5). At the bubble surface Γ_2, the electrolyte

[†]It is safe to say that the pseudosteady-state analysis predicts an overall supersaturation-lowering effect that is uniformly exaggerated.

solution contacts the gas phase which is, to a good approximation, at the operating pressure of the electrolyzer. Assuming phase equilibrium, we set the dissolved-gas concentration to its saturation value at this surface (Eq. 6). Beyond the diffusion boundary layer, the solution is assumed to be well mixed; the bulk concentration of dissolved gas obtains at the boundary-layer edge Γ_6 (Eq. 7).

The complete solution to the problem consists of the entire potential and concentration fields, but of primary interest is a quantity we refer to as ΔV: this is the net change in the cell voltage which arises from the presence of bubbles on the surface of the electrode in question. ΔV is calculated by first using the model to calculate the average potential difference between the boundary Γ_4 and the electrode metal in the *presence* of bubbles, and then subtracting this same difference calculated (by a simple, one-dimensional calculation) in the *absence* of bubbles.

Application of the Buckingham-Π theorem (19) indicates that, for a given geometry, six independent dimensionless groups are essential to the problem. One advantageous set of dimensionless groups consists of δ, J, $\frac{\alpha_a}{\alpha_c}$; Ψ, s_g, and ΔV^*. The last parameter ΔV^* contains ΔV, the voltage increment defined above, which embodies the solution to the problem. The first three parameters, δ, J, and $\frac{\alpha_a}{\alpha_c}$, are used by Newman (20) to characterize the secondary current distribution. One can think of δ as the dimensionless average current density, and J as the dimensionless exchange-current density; the ratio $\frac{\alpha_a}{\alpha_c}$ reflects the asymmetry of the reaction kinetics. The group Ψ characterizes the proportionality between the dissolved-gas flux at the elec-

trode surface and the current density.[†]

For a given problem, our procedure is to set the values of the first five parameters δ, J, $\frac{\alpha_a}{\alpha_c}$, Ψ and s_g to solve the dual field problem, and finally to evaluate the sixth parameter, ΔV^*, from the solution. The geometric configuration for a given problem, assuming spherical bubbles, can be characterized by three parameters: the contact angle θ, the interbubble spacing $\frac{s}{a}$, and the diffusion-boundary-layer thickness $\frac{l}{a}$. In the examples reported here (with $\frac{l}{a}$ generally not smaller than unity), *the exact shape of the boundary layer near the bubble is found, surprisingly, to exert no significant influence on the solution.* The explanation for this is that concentration effects are felt only at the electrode surface, which, in the vicinity of the bubble, is much more strongly influenced by the bubble itself than by the diffusion-layer edge, which is relatively far away. For our purposes, then, the curved boundary Γ_6 in Figure 2-2 can be redrawn for simplicity as the straight dotted line Γ_6' .

Base Case

In developing the present model for attached bubbles, we sought to make it as general as possible so that it could be applied to different gases and a wide range of conditions (current density, bubble density, gas solubility, contact angle, transport and kinetic properties, bubble size, etc.). It is necessary, however, to select one reaction as an example and to choose a set of *base conditions* typical for that system. From these base conditions, we can vary one parameter at a time to learn how the system behaves.

[†]While Ψ is not an operating variable, but rather a constant for a given reacting system (fixed by stoichiometry, temperature, gas solubility and transport properties), it is sometimes artificially set to zero as a convenient means of excluding supersaturation effects from the model.

For the basis of our study, we have chosen hydrogen evolution in aqueous potassium hydroxide. To the extent possible, we adhere to conditions typical of hydrogen evolution in a membrane-type chloralkali cell. The base conditions taken for this study are summarized in Tables 2, 3, 4 and 5. The remainder of this section serves to explain and support the choices made.

The properties listed in Table 3 represent best estimates available from the literature, sometimes adjusted for temperature differences. References are cited in the table. The diffusivity of dissolved hydrogen was estimated by starting with the value in pure water at 16 °C (22) and correcting this according to the Walden rule to the appropriate viscosity (23) and temperature. The solubility of H_2 was temperature corrected from a value at 30 °C (24) by analogy to solubility-vs.-temperature data in pure water (25). The first exchange-current density listed is an average of several figures cited for

Table 2. Base Case: Operating Conditions

Reaction:	Hydrogen evolution $H_2O + e^- = OH^- + \frac{1}{2} H_2$
Electrolyte:	30-weight-percent KOH in water
Temperature:	80 °C
Current density:	300 mA/cm^2
Pressure:	1 bar

Table 3. Base Case: Physicochemical Properties

Property	Symbol	Value	Reference
Conductivity	κ	0.94 $\Omega^{-1}cm^{-1}$	(21)
Dissolved-gas diffusivity	Dg	7.1×10^{-5} cm^2/s	(22)
Gas solubility	c_g^{SAT}	9.5×10^{-8} $mole/cm^3$	(24)
Exchange-current density (conventional cathode)	i_o^o	0.1 mA/cm^2	(26)
Exchange-current density (activated cathode)	i_o^o	160 mA/cm^2	(27)
Transfer coefficients	α_a α_c	0.57 0.43	(26)

hydrogen evolution in 30-to-50-percent NaOH at 80 °C on nickel and graphite (26). The transfer coefficients were obtained from the same source. The second value of exchange-current density listed was calculated† from polarization data reported for an activated cathode recently developed by the Dow Chemical Company (27).

In the present work, we take the gas concentration beyond the boundary layer, c_g^{BULK}, to equal the solubility, c_g^{SAT}. Vogt (50) reports that c_g^{BULK} usually exceeds c_g^{SAT}, but that the degree of supersaturation in the bulk is very small compared to

† With the values of the transfer coefficients α_a and α_c unavailable, we adopt the values listed in Table 3.

that at the electrode surface.

Table 4 gives a geometric description of the attached-bubble layer. The choice of these values for our base case is based upon detailed descriptions of the bubble layer which are reported in the literature.

Sillen and van Stralen (32) report an average radius of 20 μm for bubbles in the bulk at our base conditions (hydrogen evolution at 300 mA/cm^2 in 30-percent KOH at 80 $^\circ$ C) with a background flow velocity of 30 cm/s. We take this value as our figure for attached bubbles, knowing that further growth by diffusion after detachment is minimal, and that coalescence is relatively unimportant in alkaline hydrogen evolution (33,28,32).

There is little published data on contact angle. Glas and Westwater (31) report from their observations that "the contact angle always changes during the growth of

Table 4. Base Case: Geometry

Parameter	Symbol	Value
Bubble radius	a	20 μm
Contact angle	θ	40 $^\circ$
Diffusion-boundary-layer thickness	l	20 μm
Interbubble spacing	s	1 bubble diameter (closest packing)

every electrolytic bubble." Hydrogen bubbles in 30-percent H_2SO_4 range from 70° to 20° as they grow. We take 40° for our base value in this study.

The thickness of the diffusion boundary layer is discussed by various researchers (33,30,34) in characterizing mass transfer at gas-evolving surfaces. Janssen (33) reports a value of 20 μm for hydrogen evolution at 300 mA/cm^2 in 6 M KOH on a horizontal platinum electrode. This is obtained by measuring the limiting current density of an indicator ion (indirectly, by determining the quantity of indicator ion reduced during an experiment). The Nernst-diffusion-layer thickness, l, is defined by the well-known equation:

$$i_L = \frac{n\ F\ D_p\ c_p}{l} \tag{17}$$

where D_p, c_p and i_L are the diffusivity, bulk concentration, and limiting current of the indicator ion. The above estimate agrees reasonably well with other measurements at similar conditions (29,30,35).

From various qualitative descriptions of alkaline hydrogen evolution, the bubble curtain is known to be densely packed with bubbles. Since the maximum void fraction attainable is higher for a distribution of bubble sizes (with small bubbles occupying the interstices of larger ones) than for a monosized array (the idealization made in our model), we chose an interbubble spacing of unity (closest packing) as the most realistic condition under the constraints of the model. Indeed, the void fraction in the attached-bubble layer is estimated by Bongenaar-Schlenter, *et al.* (7) to range from 0.60 to 0.85; the lower figure agrees well with the void fraction of a close-packed hexagonal array.†

† The experimental findings of Janssen, *et al.* (28), while not performed at exactly the same conditions of concentration, temperature, and current density as listed in Table 2, do suggest that the closest-packing assumption may exaggerate the true density of at-

The values selected to describe our base case (Tables 2, 3 and 4) are used to evaluate the dimensionless groups that enter the model. These base-case parameter values are listed in Table 5.

Numerical Method

The coupled pair of boundary-value problems is solved in a cyclic iteration scheme, which employs a Newton-Raphson routine to speed convergence. The starting point in the cycle is to choose an initial estimate of the current distribution on the electrode surface. This fixes both the potential gradient and the concentration gradient at the

Table 5. Base Case: Dimensionless Parameters

Parameter	Symbol	Value
Dimensionless current density	δ	2.05×10^{-2}
Dimensionless exchange-current density	J	6.84×10^{-6}
Dimensionless gradient ratio	Ψ	2.25×10^{4}
Stoichiometric ratio	$\frac{s_g}{n}$	$\frac{1}{2}$
Ratio of transfer coefficients	$\frac{\alpha_a}{\alpha_c}$	1.33

tached bubbles. These authors define the degree of screening by attached bubbles, p, as "the fraction of the electrode surface covered by projection of the bubbles," and report, for example, a value of $p = 0.56$ for unstirred hydrogen evolution at 150 mA/cm^2 in unstirred 1M KOH at 30 °C on a transparent nickel electrode, whereas the base case chosen for our model corresponds to $p = 0.91$.

electrode surface and, in doing so, supplies the full complement of boundary conditions for each elliptic partial differential equation (PDE). The potential-field problem and the concentration-field problem are solved independently by the boundary-element method. Finally, the newly calculated surface values of potential and concentration are used in the overpotential matching condition (Eq. 9) to produce a revised estimate of the current distribution.

For solving the field problems, the boundary-element method (BEM) was chosen. A relatively new technique, related to boundary-integral-equation methods (BIEM) and influenced by finite element (FEM) concepts, BEM offers several attractive features, most of which are consequences of the reduction in dimensionality of the problem: the boundary-value problem is restated and solved entirely on the boundaries of the domain. Among other benefits, this greatly reduces the amount of input data to describe the problem geometry, while the method appears to be better than or comparable to the domain methods (finite-element and finite-difference) in terms of accuracy and computational efficiency.

Quadratic elements were chosen to represent the boundary in our problem: each element contains three nodes so that both the element's shape and its functional interpolation are parabolic. Double nodes are employed at the corners of the domain. Figure 2-3 is an example of a boundary-element nodal structure used in this work. The discretization for the potential-field problem contains 46 elements and 96 nodes; the concentration-domain contains 34 elements and 72 nodes.

The Fortran code written for the present model is based on a direct formulation of the boundary-element method for potential problems outlined by Brebbia (36). Special provision is made for the axisymmetric problem. The element matrices are generated

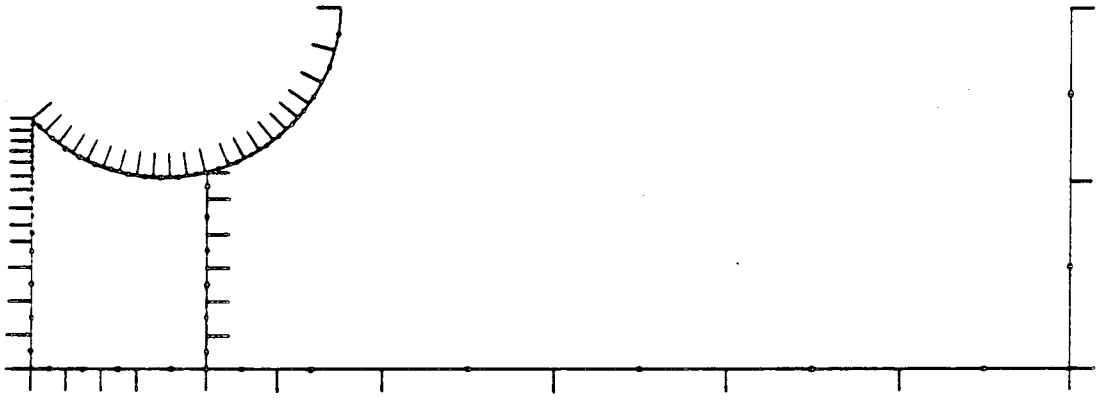


Figure 2-3. Boundary-element nodal structure for an example problem.

by integrating the source function u and its normal derivative q over the axisymmetric domain boundary. Integration in the circumferential direction is straightforward and gives expressions containing the complete elliptic integrals. Subsequent integration in the longitudinal direction is performed by 12-point Gaussian quadrature, but is complicated by a logarithmic singularity at the source point. This difficulty can be circumvented in the integration of q, since the diagonal matrix elements can be calculated by difference as outlined by Brebbia (37). The u integration, however, requires special attention. The procedure used in this work is the following: the integrand is expanded using the polynomial approximation for the elliptic integrals; since only one term in the expansion is unbounded at the point of singularity (38), this term alone is integrated analytically while the remainder of the expression is integrated by quadrature. (An IMSL (39) library subroutine is used to evaluate the elliptic integrals.) Compared to the other integration schemes used in the relatively few published papers on boundary-element solutions of axisymmetric potential problems (40,41,42,43), this approach represents an improvement in simplicity (no subdivision of the element is needed) and accuracy (no additional approximation is introduced to treat the singularity).

The convergence order of the method was found to be two or higher, in a series of runs with successively refined nodal structures for a primary-current-distribution problem.

The matrix problems are solved by Gaussian-elimination with full pivoting. A substantial savings is achieved by performing the expensive forward-reduction step only once for each problem geometry. The forward-reduced matrices and a record of row manipulations and pivots are stored so that, upon each subsequent field calculation in the iterative cycle, only the back-substitution step must be repeated.

As an example of a typical calculation, we solve the base-case problem for the activated cathode ($J = 0.011$) and an interbubble spacing of 2 (the nodal structure shown in Figure 2-3). (Figure 2-16 contains the voltage-increment answer to this problem.) The two system matrices (potential and concentration) are 96×96 and 72×72 in size. The Jacobian matrix is 27×27, as there are 27 nodes on the electrode boundary. Convergence to a relative tolerance of 10^{-10} is achieved in 7 Newton-Raphson iterations, each requiring 56 solutions of Laplace's equation. This takes 300 seconds of CPU time on a VAX 8600 computer at double precision. A subsequent problem having the same node configuration takes only 84 seconds.

Results and Discussion

Throughout this article, the dimensionless groups will sometimes be referred to by incomplete names for brevity. For example, δ will be called "current density," which deemphasizes the fact that δ also embodies the scale of the problem and the conductivity.

Primary Current Distribution

Our starting point is to use the model to solve the primary distribution, for which overpotential is neglected (in the context of our model, $J = \infty$ and $\Psi = 0$). The electrode surface is at a uniform potential, the electric field problem is directly analogous to heat conduction problems, and the current distribution depends only on geometry. Figure 2-4 shows the primary current distribution for a lone bubble with a contact angle of 40 degrees. As in the other current distribution plots in this paper, the current density is normalized with respect to i_{AVE}, the current density that would prevail if no bubbles were present on the surface. As expected for a primary distribution, the current density falls to zero where the bubble touches the surface at an acute

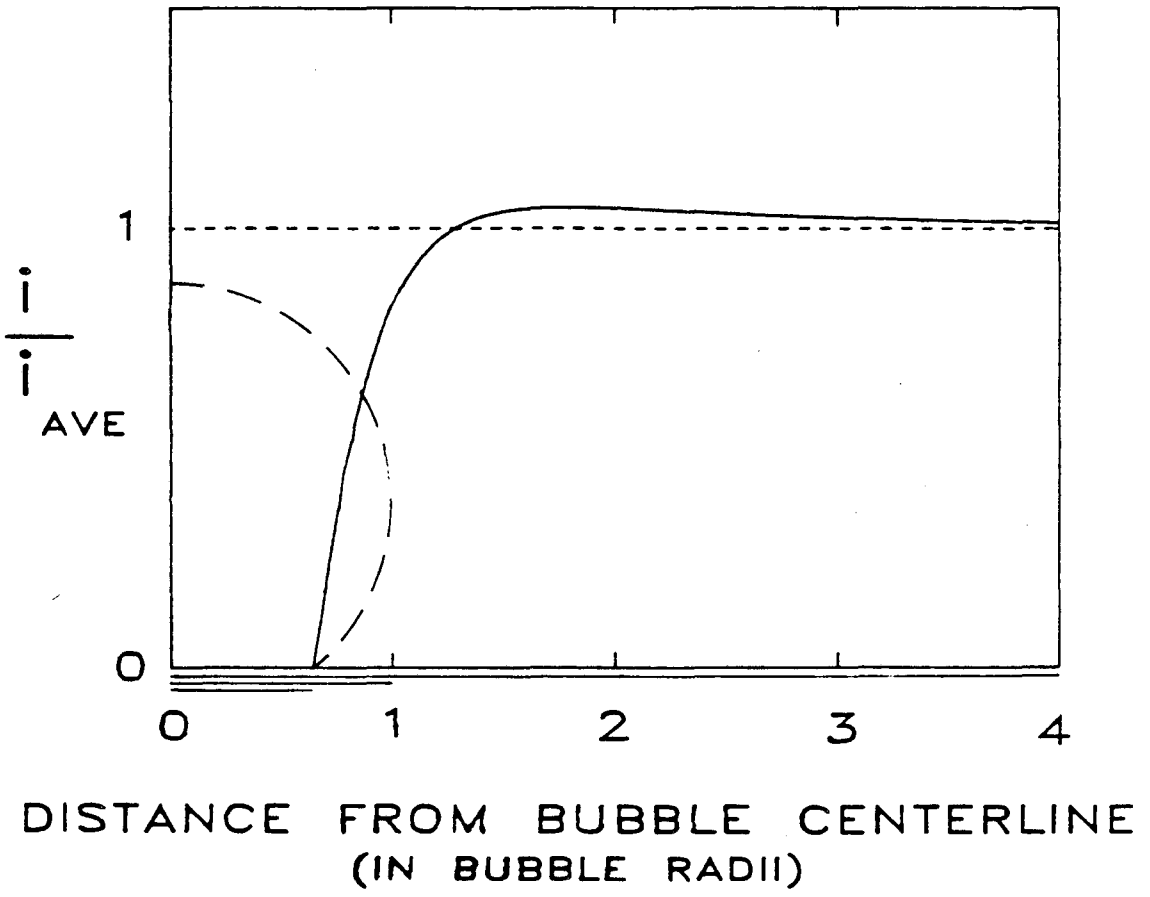


Figure 2-4. Primary current distribution near an isolated attached bubble with a contact angle of 40 degrees.

angle. Far from the bubble, of course, its influence is not felt, but several radii away from the centerline, the current density slightly exceeds its average value, compensating for the decreased current on and near the bubble contact area. This same behavior is predicted by Sides and Tobias (11) in their analytic solution of the primary distribution around a tangent bubble (contact angle = 0). We find that when the contact angle, and consequently the bubble's contact area with the electrode, is nonzero, the magnitude of the compensation effect is increased. In this first plot of current distribution, an outline of the bubble profile is drawn so that the geometry can be better visualized. Also, in this figure and in subsequent current-distribution plots, there are three horizontal lines below the abscissa which describe the geometric configuration: the bottom line spans the electrode region covered by the bubble; the middle line covers the projected area of the bubble; and the top line represents the the equal-area cylinder used to approximate the hexagonal symmetry cell. (In the case of Figure 2-4, for a lone bubble, this cylinder is of infinite radius and the top line spans the entire radius scale.)

According to visual observations (7,28,30,31,44,45,51), the attached bubbles on a gas-evolving electrode are usually quite crowded and the treatment of an isolated bubble is not applicable. With the use of the equal-area-cylinder idealization, the model is able to describe arrays of attached bubbles of arbitrary interbubble spacing. Figure 2-5 is a plot of primary current distribution near one bubble in a hexagonal array; each curve represents a different interbubble spacing. For example, the curve labeled $s = 1.4$ corresponds to a regular hexagonal array of attached bubbles, with nearest neighbors centered 1.4 diameters apart. The curve ends at a radius approximately five percent higher than half the spacing, according to

$$r = \left(\frac{2\sqrt{3}}{\pi}\right)^{\frac{1}{2}} s. \tag{18}$$

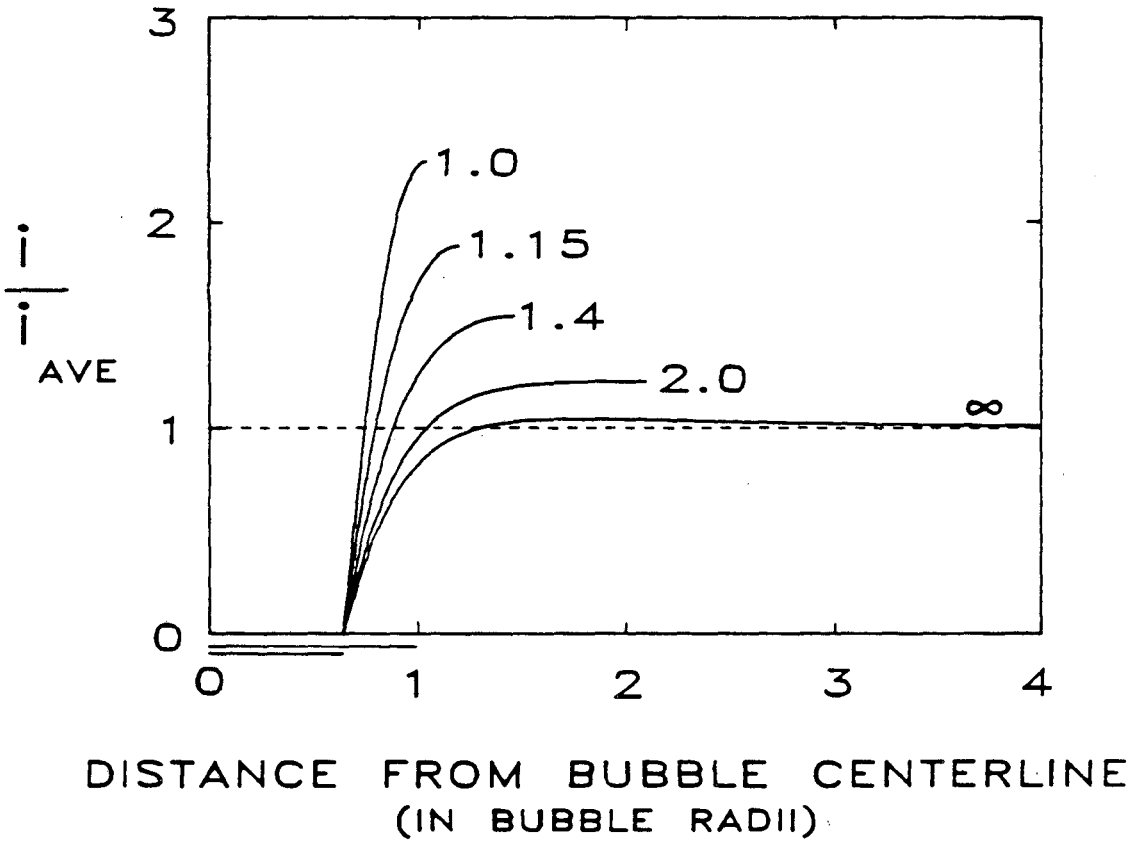


Figure 2-5. Primary current distribution near a 40-degree bubble with interbubble spacing, s , as a parameter.

This is the position of the symmetry boundary. As should be expected, each current-density profile meets the symmetry boundary with zero slope. For the finite values of interbubble spacing represented on this figure, the current density is also at its maximum at this boundary. (However, the current corresponding to $s = 5$, for example, would not be expected to follow this description.) The value of this maximum current density is seen to increase with the density of bubbles on the surface. This is consistent with expectations because, when the bubbles are more crowded, a greater fraction of the electrode area is unavailable for passing current, and so to compensate for this loss, a greater current density must pass on the remaining area. Not surprisingly the current density is highest away from those recessed regions near the contact area to which the flow of current is sterically deterred.

As mentioned in the previous section, it is the extra potential drop caused by the presence of attached bubbles that is of primary interest to engineering practice. Figure 2-6 shows how this voltage increment, ΔV^*, depends on the interbubble spacing of a hexagonal array. For a primary potential problem (neglecting polarization), we achieve a more general expression of the solution by plotting *resistance* increment, $\Delta V^*/\delta$, instead of *voltage* increment, ΔV^*. (Once polarization has been introduced, the solution is no longer general to all values of δ .) The solid curve is predicted by the model. There is a pronounced increase in the voltage increment as spacing approaches unity (the condition of closest packing).

The asterisks on Figure 2-6 are data from an analogue experiment performed by Sides and Tobias (10). In this experiment, dielectric spheres were placed inside an electrochemical cell which is hexagonal in cross section, and whose walls correspond to the symmetry boundaries of a hexagonal unit cell (see Fig. 2-1). The good agreement between our model and the analogue experiment indicates that the equal-area-cylinder

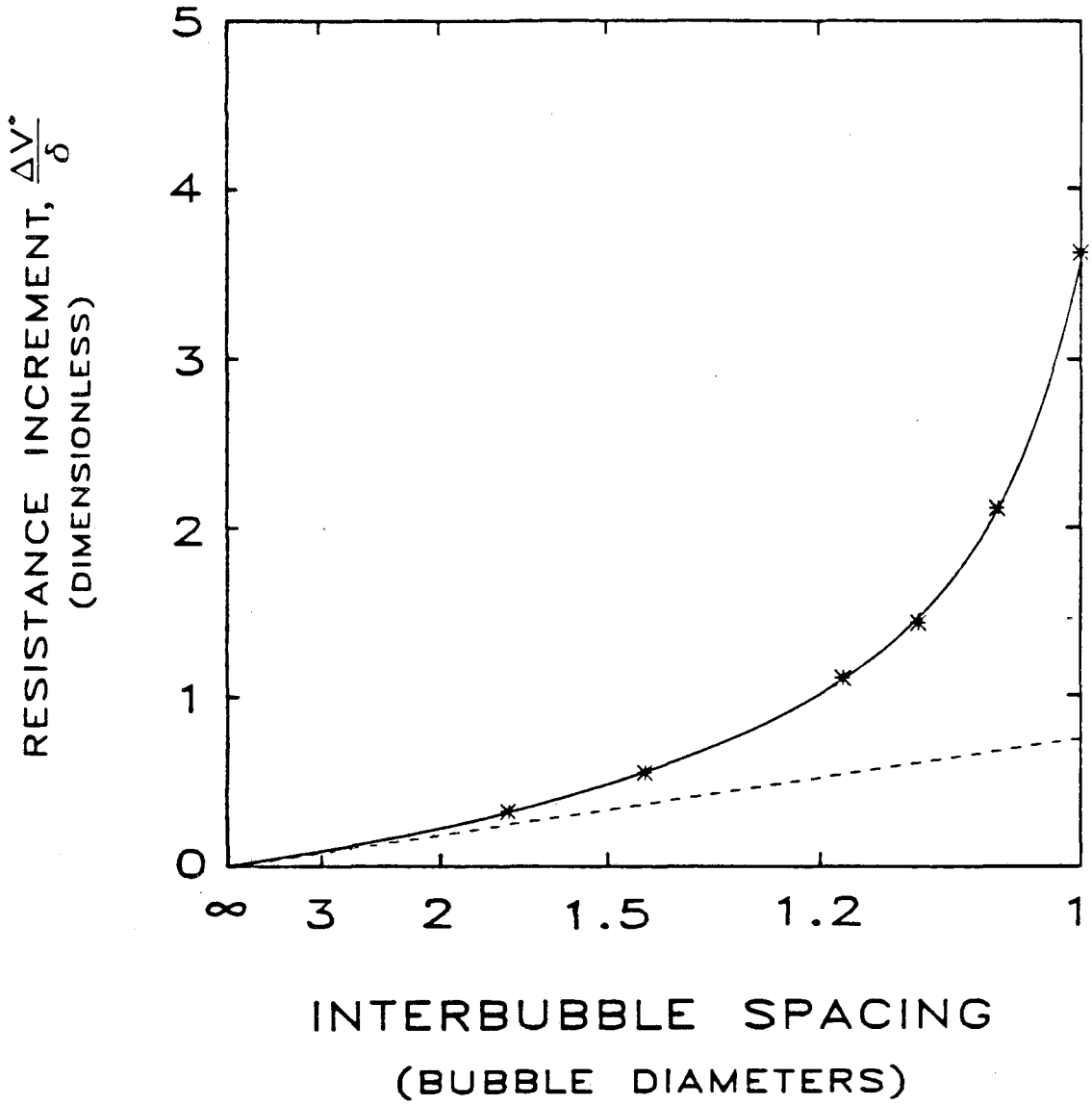


Figure 2-6. Resistance increment, $\Delta V^* / \delta$, as a function of interbubble spacing, s, for bubbles tangent to an isopotential electrode surface. Asterisks represent data from an analogue experiment (10).

approximation is well justified. The dotted line in Figure 2-6 corresponds to the voltage increment that would be calculated if one were to begin with the isolated-bubble result and extrapolate this on a unit-area basis to closer spacings. (The dotted line is straight since we have plotted the inverse square of interbubble spacing on the abscissa.) As can be seen, and as discussed by Sides and Tobias (10), the latter simplistic approach substantially underestimates the voltage increment due to densely-packed attached bubbles. The reason for this difference is that an isolated attached bubble creates a disturbance in the potential field which extends radially for several bubble diameters, whereas, when the bubbles are closely packed, their potential-field disturbances interact. When the bubble layer is crowded, the current can no longer take a wide detour around each bubble, but must pass through the narrow interstitial voids.

It should be noted that Figure 2-6 does not correspond exactly to the current-distribution family of Figure 2-5; the former is for tangent bubbles (allowing direct comparison to the analogue experiment (10)) whereas the latter is for a 40-degree contact angle.

Secondary Current Distribution

In this section, we introduce the surface overpotential (*i.e.* we consider finite values of J) while continuing to ignore concentration polarization (*i.e.* Ψ is fixed at zero).

Figure 2-7 shows the influence of bubble density on the voltage increment with J as a parameter and δ fixed at its base-case value. (Again, J is the dimensionless exchange-current density, a measure of the kinetic speed of the electrode reaction.) The lowest curve, for infinite J, is very similar to the curve from Figure 2-6; (they are for different contact angles). As a trend, the smaller the value of J, the larger is the

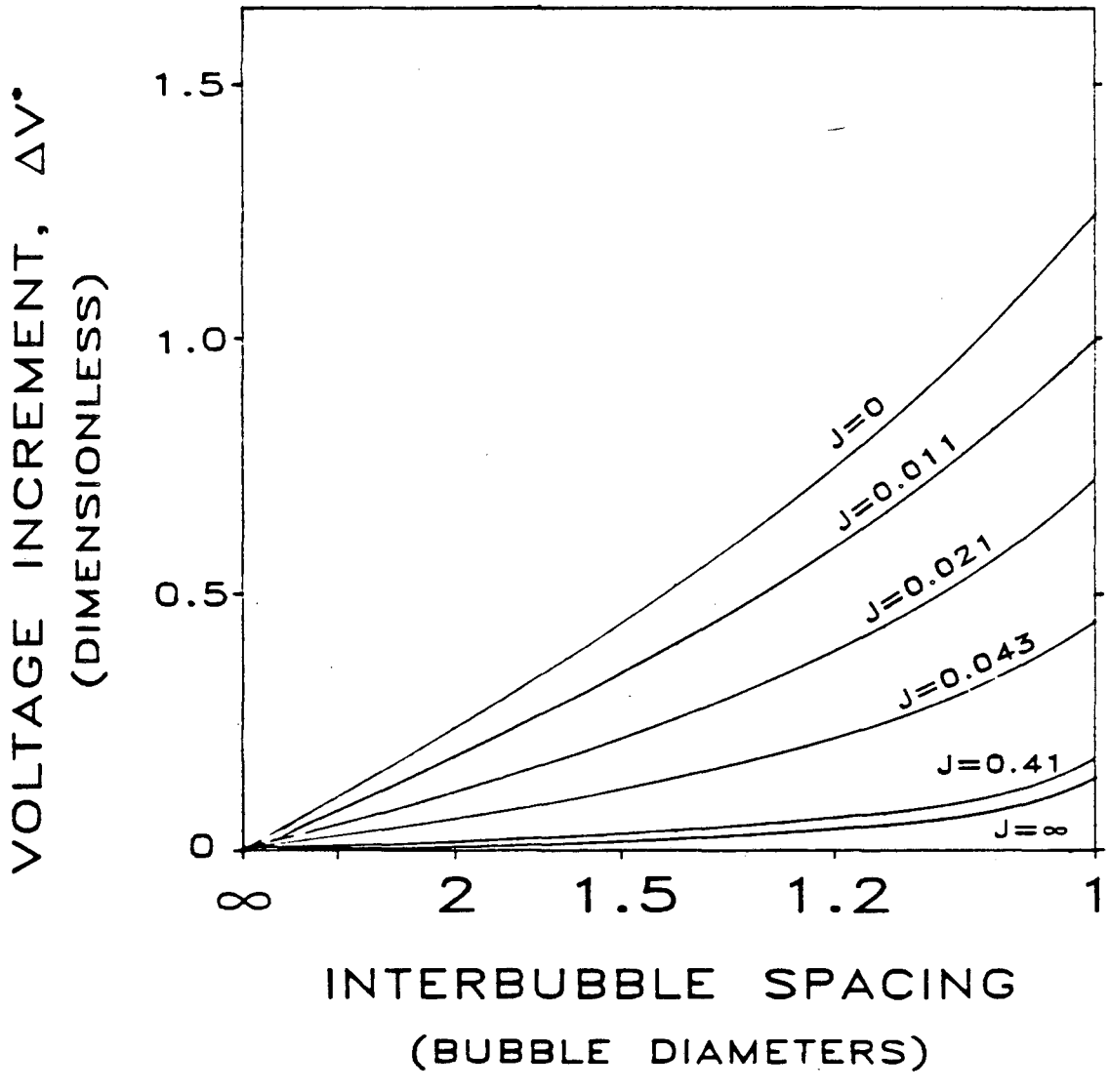


Figure 2-7. The effect of interbubble spacing, s, on the voltage increment due to a hexagonal array of bubbles, ΔV^*, for different values of the dimensionless exchange-current density, J. (At base conditions)

voltage increment caused by the layer of attached bubbles. Conceptually, one can view each higher curve as having more sluggish kinetics and, in turn, requiring an increasingly large surface overpotential to drive the electrode reaction. In the limit of $J = 0$, the voltage increment can be calculated purely from kinetic effects as follows:

$$\Delta V^* = \ln \frac{1}{1-\sigma}, \tag{19}$$

where σ is the fraction of the electrode area masked by attached bubbles. In this limit, the voltage increment is due solely to the rise in surface overpotential associated with the reduction in area available for the electrode reaction. In the context of our model, σ is related to bubble spacing and contact angle by a simple geometric argument:

$$\sigma = \frac{\sqrt{3}\,\pi \sin^2\theta}{6\,s^2}. \tag{20}$$

At 80 °C, the maximum value of ΔV^* (at $J = 0$ and $s = 1$) corresponds to an increment of 39 millivolts.

As explained in the section on base conditions, we have chosen closest packing ($s = 1$) as the base condition for our example of alkaline hydrogen evolution. The remainder of the results presented correspond to this case.

The interplay between ohmic and kinetic effects can also be vividly seen in the current distribution. Figure 2-8 shows a family of current-density profiles for different values of J (at base values of δ, s, θ, and with $\Psi = 0$). The limiting case of $J = \infty$ is the primary current distribution for this geometric configuration (presented on a different scale in Figure 2-5). The current density vanishes at the bubble contact area. For successively smaller values of J, the distribution becomes more uniform. In the limit of $J = 0$, all ohmic effects disappear, and the current density is uniform over the region on the electrode surface that is not masked by bubbles. The figure clearly illus-

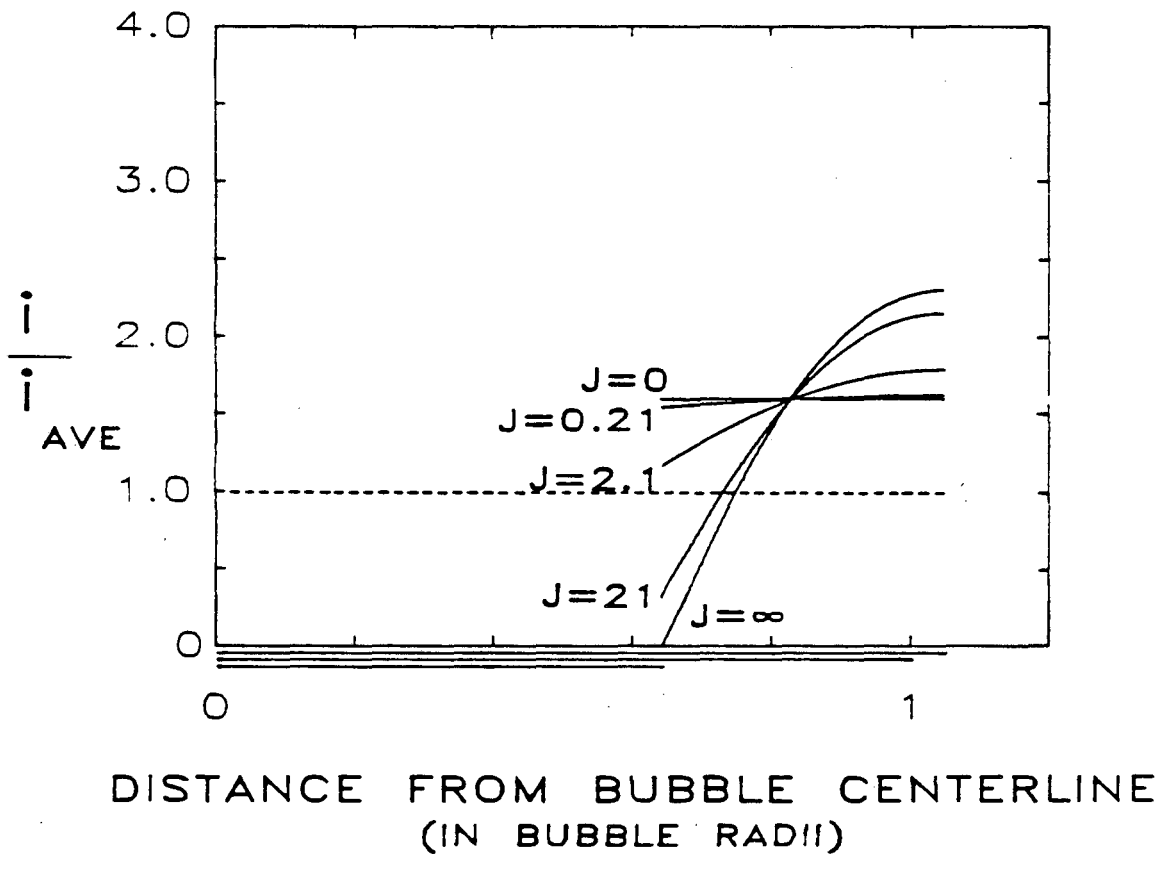


Figure 2-8. Current distribution near one attached bubble with dimensionless exchange-current density, J, as a parameter, *neglecting* supersaturation effects.
(At base conditions)

trates that this current density exceeds the superficial current density, i_{AVE}. In this particular example the effective current density is 60 percent higher than i_{AVE}.

In the section describing the model, the voltage increment for a given geometric configuration was shown to depend on five dimensionless groups. Two of these, Ψ and s_g, are concerned with concentration effects and can be disregarded in a treatment of secondary current distribution. In such a problem, then, ΔV^* is a function of three parameters, δ, J, and $\frac{\alpha_a}{\alpha_c}$, and it is difficult to obtain and represent the general solution in any manageable way. Figure 2-9 shows how ΔV^* depends on δ and J for a given bubble configuration (the base case) and at a single value of $\frac{\alpha_a}{\alpha_c}$ (the base value of 1.33). This log-log plot spans a very wide range of δ and J and its features are interpreted in the following paragraph.

In the limit of high current density, the voltage increment due to attached bubbles is dominated by the ohmic effect (which is linear with current while surface overpotential is only logarithmic.) Accordingly, the exchange-current density plays no role, and the family of curves collapses into a single curve. Perhaps the best way to interpret the remainder of the plot is to begin on this high-current asymptote and to follow one curve at a time into the low-current region. Beginning with the curve labeled $J = 6.8\times10^{-2}$ (facile kinetics), one sees that this curve departs only slightly from the ohmic asymptote because there is little kinetic resistance. At the opposite extreme, a case with very slow kinetics, such as $J = 6.8\times10^{-7}$, the behavior is more complicated: Proceeding downward from the high-current region, there is first a departure from the asymptote, reflecting the combined influence of ohmic and kinetic resistance, with the electrode kinetics in the Tafel region. Farther down, a horizontal asymptote is

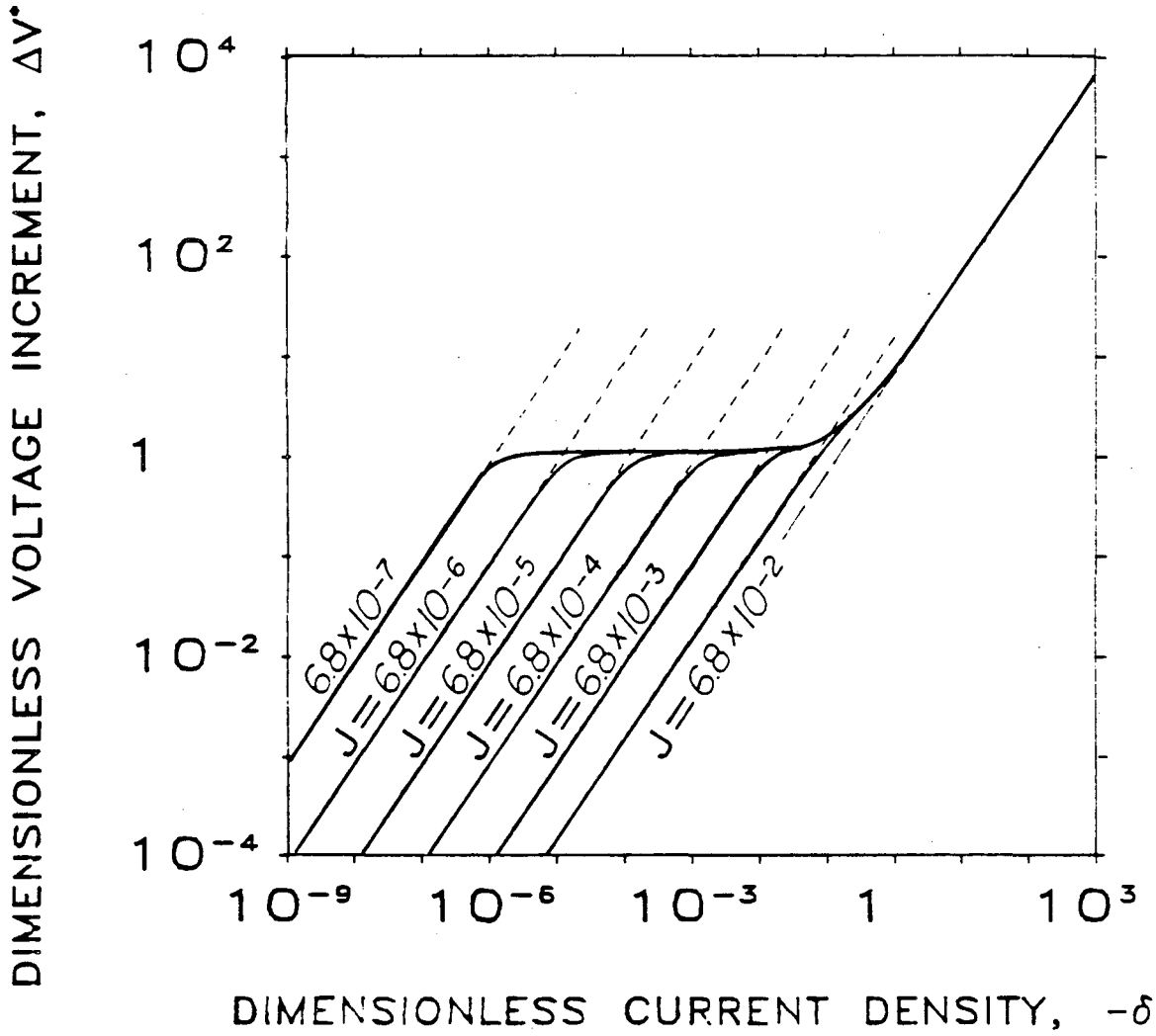


Figure 2-9. Voltage increment due to attached bubbles, ΔV^*, as a function of current density, δ, for different exchange-current densities, J, *neglecting* supersaturation effects. (At base conditions)

reached: this is a regime dominated by Tafel kinetics. Here, the voltage increment depends neither on the exchange current nor on the current and can be expressed solely as a function of σ, the fractional electrode coverage by bubbles. Continuing to lower currents, one departs from the horizontal asymptote into a region where the kinetic overpotential (which continues to dominate the voltage increment) is in transition between Tafel and linear behavior. Finally, in the low-current limit, one reaches a linear-kinetics asymptote. The other curves for low J in the family obey the same description. However, when the reaction is relatively fast (eg. $J = 6.8 \times 10^{-2}$) the Tafel region is skipped, and the low-current asymptote represents the combined contributions of linear kinetics and the ohmic effect.

It is quite instructive and useful to restate the dependence of ΔV^* on δ, J and $\frac{\alpha_a}{\alpha_c}$ in a different form. While on a rigorous basis,† three parameters are required to characterize the secondary current distribution, it is customary in electrochemical-engineering practice to use a single dimensionless group, the Wagner number,

$$W = \frac{\left(\frac{\partial \eta_s}{\partial i}\right)_{i_{AVE}} \kappa}{a} \tag{21}$$

where the function $\eta_s(i)$ is given by Eq. 14. Conceptually, the Wagner number is the ratio of kinetic to ohmic resistance. A plot of the resistance increment $\Delta V^*/\delta$ as a function of W is given in Figure 2-10; all computed points used to construct Figure 2-9 now fall very nearly onto a single curve. Not shown are several points calculated at different values of $\frac{\alpha_a}{\alpha_c}$, which also collapse fairly well onto the curve. With due consideration for the fact that the one-parameter treatment is approximate, and that there

†according to the Buckingham-Π theorem (19)

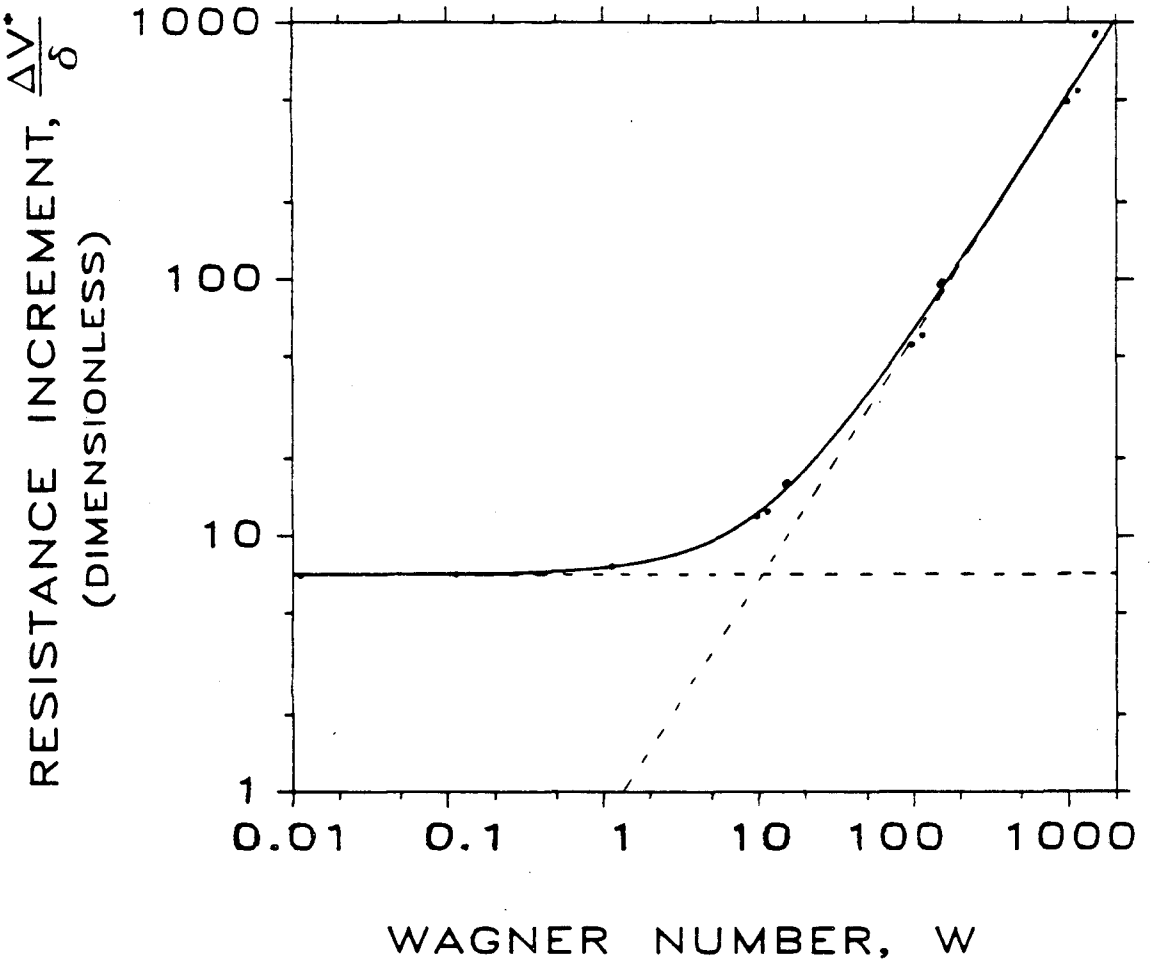


Figure 2-10. The dimensionless resistance increment due to attached bubbles, $\frac{\Delta V^*}{\delta}$, as a function of Wagner Number, W, for the base configuration, *neglecting* supersaturation effects. Dots (some coinciding) represent data used to construct Figure 2-9.

is no guarantee of equal success for other geometric configurations, it is important to recognize the practical value of this way of correlating the data: a three-parameter model collapses into a one-parameter correlation.

The dependence of the resistance increment on Wagner number can be described in terms of three regimes shown on Figure 2-10. At low Wagner number, the ohmic effect overwhelms the kinetic effect, and the extra resistance due to the presence of attached bubbles is constant (the horizontal asymptote). At high Wagner numbers, kinetic limitations overwhelm the ohmic effect and $\Delta V^*/\delta$ is proportional to W (the diagonal asymptote). At this limit, the kinetic regime can be linear, Tafel or between the two. Finally, there is the transition region, for Wagner numbers on the order of unity, where both ohmic and kinetic effects are important.

The versatility of the Wagner number in characterizing secondary current-distribution problems can be partially understood in terms of its limiting forms: in the Tafel region, where J is unimportant and the solution depends on δ, W is identical to $1/\delta$; conversely, in the linear regime, where J is dominant, W is identical to $1/J$.

Current distribution is another aspect of the solution, besides the resistance increment, which can be characterized to good accuracy by Wagner number. Figure 2-11 shows a family of current-density profiles, each at a different Wagner number, for the base configuration. Two limiting cases are apparent: the primary distribution at $W = 0$, and the uniform distribution (over the unmasked electrode area) at $W = \infty$.

The effect of attached bubbles on potential drop and current distribution also depends on the contact angle of the bubbles. Figure 2-12 shows the dependence of voltage increment on contact angle for a close-packed array of bubbles at base conditions. This dependence is shown to vary strongly with the degree of kinetic control. When

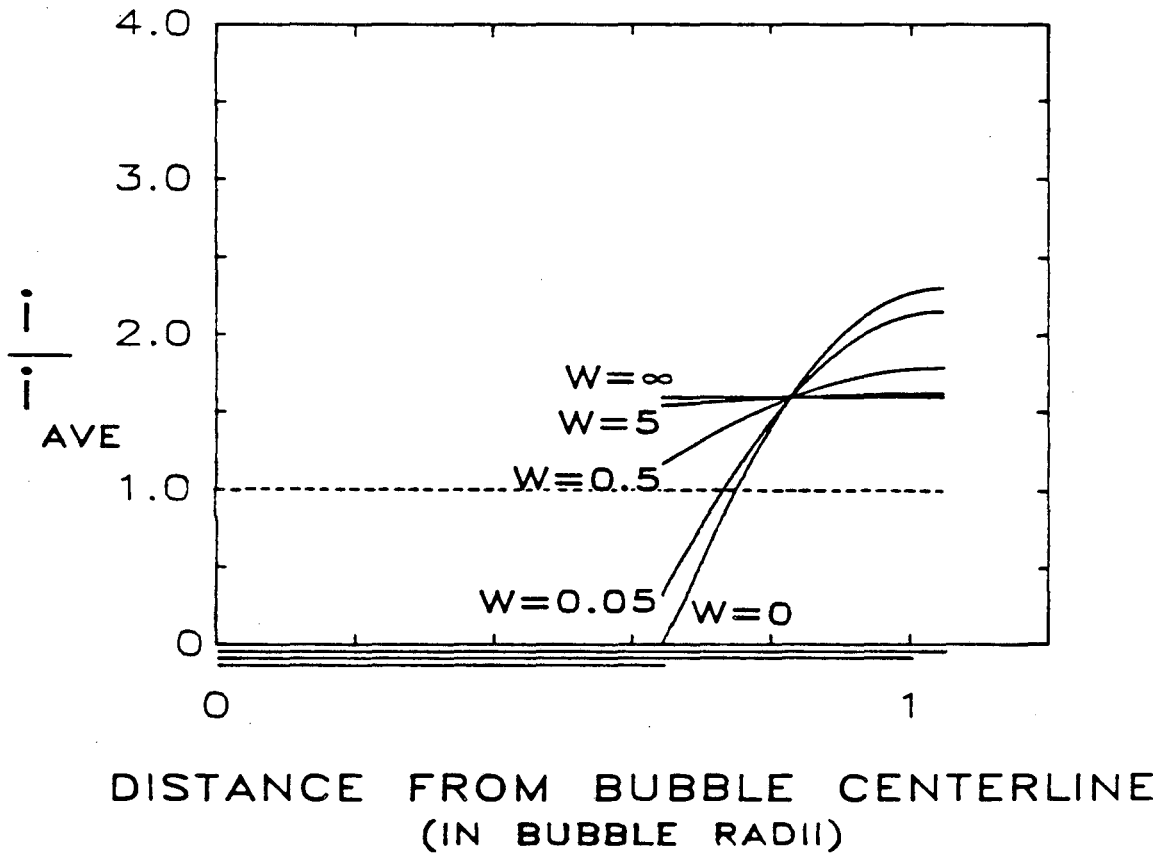


Figure 2-11. Dependence of current distribution on Wagner number, W, for the base configuration, *neglecting* supersaturation effects.

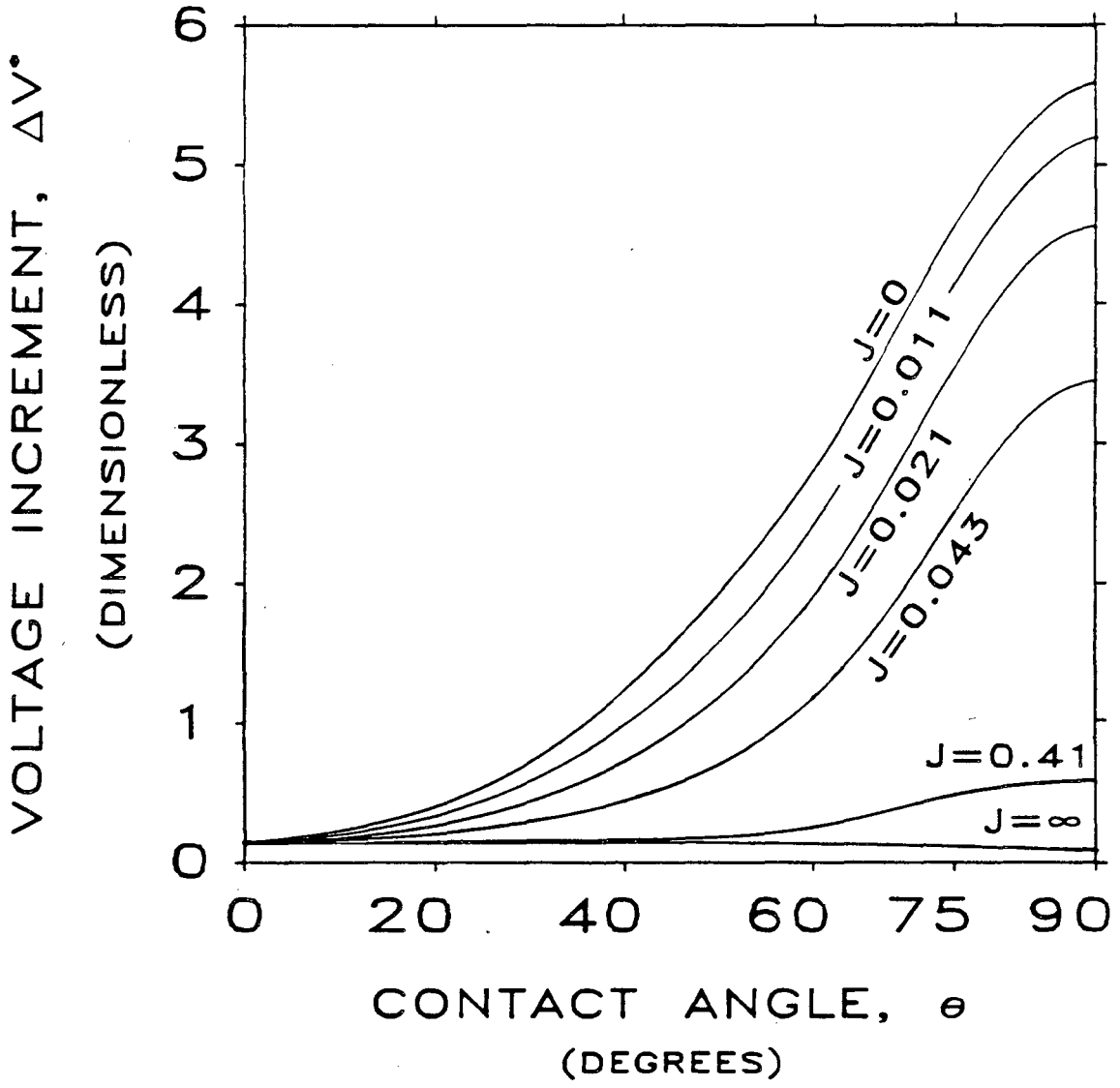


Figure 2-12. The effect of contact angle, θ, on the voltage increment due to attached bubbles, ΔV^*, for different values of the dimensionless exchange-current density, J, *neglecting* supersaturation effects. (At base conditions)

kinetic resistance dominates (the case of $J = 0$) the voltage increment due to the presence of attached bubbles rises dramatically with contact angle. Again, the overriding consideration is the area available for the electrode reaction. At a contact angle of 90 degrees (hemispherical bubbles) there is a maximum in electrode coverage and, consequently, in voltage increment. At the opposite extreme of no kinetic resistance (the case of $J = \infty$), the voltage increment decreases with contact angle. This agrees with the expectation that, in terms of ohmic effects alone, half a bubble offers less obstruction than a whole bubble. It should be remembered that Figure 2-12 pertains to the case of closest packing of bubbles; at lower packing densities, the dependence on contact angle will be weaker. It is also valuable to return to Eq. 20 and recall that contact angle, together with interbubble spacing, determines the electrode coverage, σ, which, in the important limit of kinetic control in the Tafel regime, uniquely determines the voltage increment.

Supersaturation Effects

In treating the combined effects of surface overpotential and dissolved-gas-concentration overpotential, (the general case of finite J and nonzero Ψ), we begin by examining the current distribution. Figure 2-13 shows a family of current-density profiles calculated for hydrogen evolution at base conditions. This is the same problem as illustrated in Figure 2-8 except that, in the present case, Ψ has been set to its base value of 22,500. The most striking feature of Figure 2-13 is the *inversion in the current distribution:* the current density is highest where the bubble touches the electrode. This curious effect can be explained in terms of decreased concentration overpotential in the vicinity of the bubble. Away from the bubble, the electrolyte near the electrode becomes supersaturated to a high degree with the dissolved-gas product (46,47,48). This gives rise to a substantial concentration overpotential. At the bubble

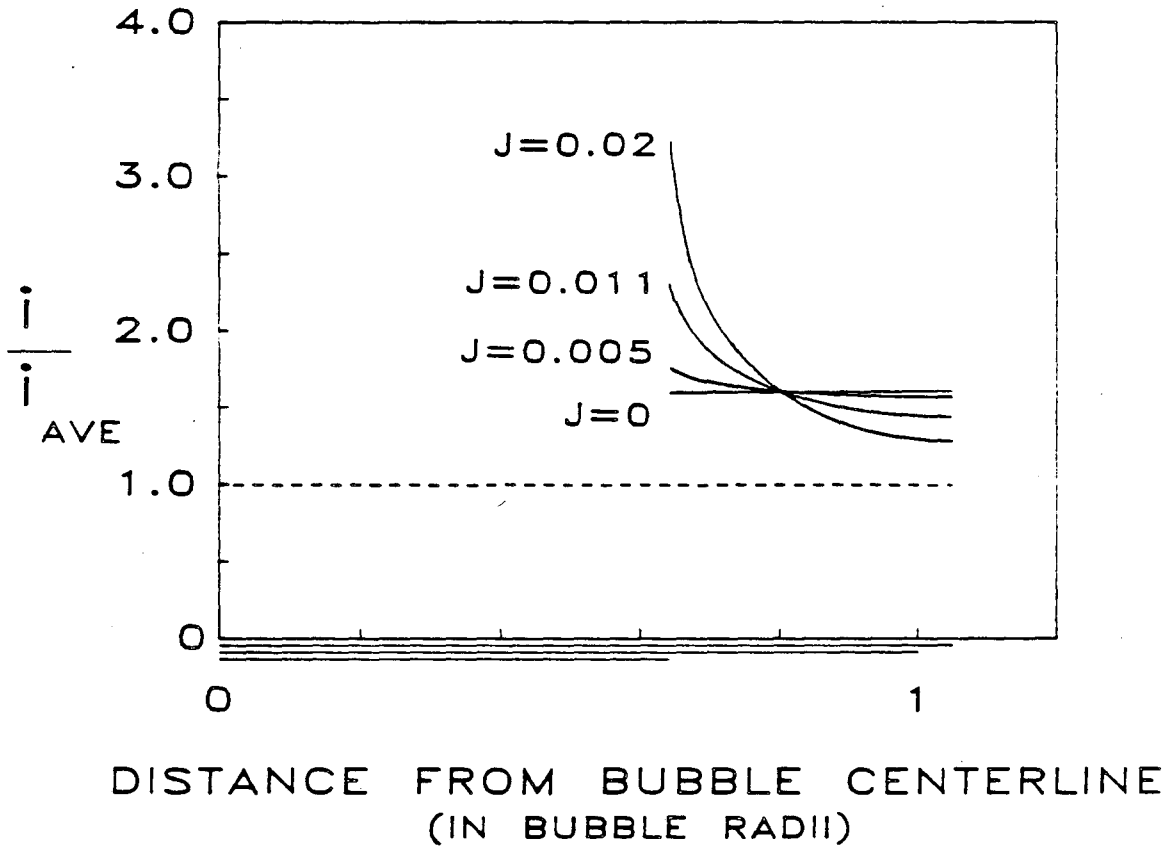


Figure 2-13. Current distribution near one attached bubble with dimensionless exchange-current density J as a parameter, *including* supersaturation effects.
(At base conditions)

surface (including the ring at which the bubble contacts the electrode), there can be no supersaturation if phase equilibrium is obeyed; therefore, the concentration overpotential can be expected to be depressed near the bubble and to vanish entirely at the contact ring. This localized depression of a large voltage obstacle causes current to be preferentially attracted to the region near the bubble. This phenomenon will be referred to subsequently as the "enhancement effect" of attached bubbles on current density.

Interestingly, the enhancement effect, which can be quite dramatic under certain conditions, is seen to die out completely in the limit of slow electrode kinetics; the current-density profile labeled $J = 0$ in Figure 2-13 is level, exactly as would be predicted in the absence of supersaturation effects (cf. Fig. 2-8). This important phenomenon, *the disappearance of supersaturation effects under slow electrode kinetics,* is worth discussing in both mathematical and physical terms.

In the equation for electrode kinetics, Eq. 10 of Table 1, the reaction rate is expressed in two exponential terms. For small J, one term vanishes; this limiting case corresponds to the Tafel regime of electrode kinetics. Since the dissolved-gas concentration appears only in this vanishing term, the influence of concentration on the reaction rate must vanish as well. Physically, the two terms correspond to the forward and reverse components of the electrode reaction. In the Tafel regime, the forward reaction (hydrogen evolution) overwhelms the reverse reaction (oxidation of hydrogen gas), in a complete departure from the reversible or Nernstian condition. The strong potential driving force essentially prohibits the back reaction, and it becomes impossible for the concentration of the products to influence the rate of reaction. Accordingly, *in the Tafel kinetic regime, the buildup of supersaturated gas product has no influence on the overpotential of the reaction.* Finally, since this overpotential is the only way for concentration to enter the potential-field problem, we see that, *in the Tafel regime, neither*

the current distribution nor the bubble-induced voltage increment is sensitive to supersaturation effects.

Given that the enhancement effect disappears in the sluggish-reaction limit, it is remarkable how strong the effect can be under relatively slow kinetics. The most nonuniform current-density profile in Figure 2-13 is obtained for $J = 0.02$, at which value the secondary distribution would hardly deviate from uniformity (*cf.* Fig. 2-8). Results at higher values of J are not reported because the strong variation in current density begins to exceed the resolving power of the numerical method.

The inclusion of supersaturation effects into the model also has a profound influence on the voltage increment calculated for an array of attached bubbles. Figure 2-14 shows how the voltage increment depends on current density with different electrode kinetics. This figure is directly analogous to Figure 2-9, the only difference being the inclusion of base-case concentration effects. The most noteworthy feature of these results is that, under some conditions, the calculated value of ΔV^* is negative. This means that the net effect of having bubbles present on the electrode surface is to lower the voltage of the cell. It should be remembered that ΔV^* is referred to a hypothetical condition in which dissolved gas is transported entirely by diffusion across a bubble-free boundary layer. When bubbles are present, they act as sinks for the supersaturated gas and decrease the concentration overpotential. A negative value of ΔV^* reflects that this beneficial concentration effect of attached bubbles has outweighed the disadvantageous ohmic and kinetic effects.

The nature of the supersaturation effects can be better understood by comparing Figure 2-14 to Figure 2-9. With slow electrode kinetics ($J = 6.8\times10^{-7}$ or $J = 6.8\times10^{-6}$), the behavior is identical: at moderate current densities, the reaction is

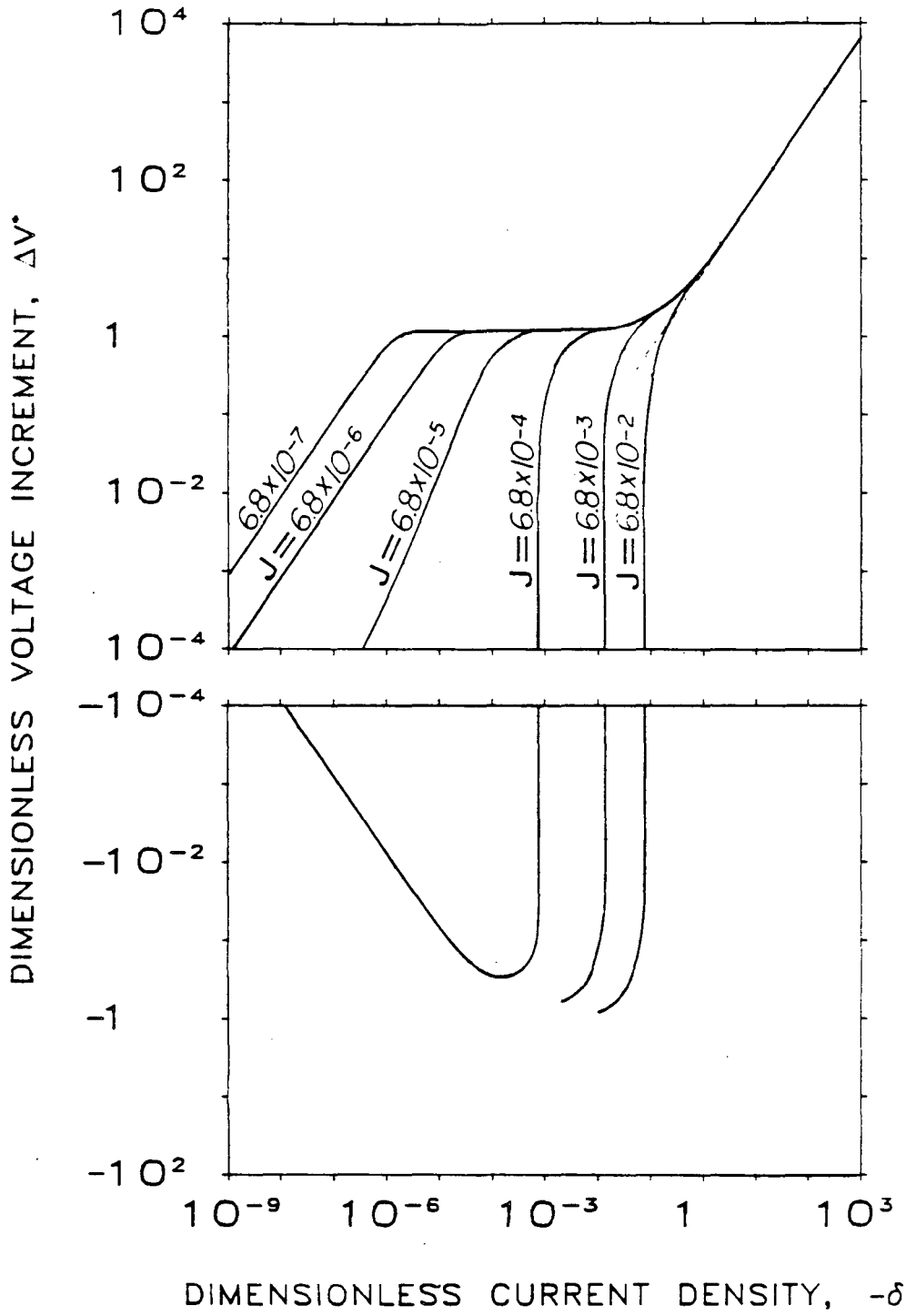


Figure 2-14. Voltage increment due to attached bubbles as a function of current density for different exchange-current densities (all dimensionless), *including* supersaturation effects. (At base conditions) Negative values of ΔV^* are shown in the lower box.

in the Tafel regime (where supersaturation has no effect), and at current densities low enough to be in the linear regime, the degree of supersaturation is negligible. For $J = 6.8\times10^{-5}$, the reaction is facile enough to permit mild supersaturation in the linear kinetic regime (at low δ) so that there is slight depolarization. (Figure 2-14 shows a downward deviation from the analogous line on Figure 2-9.) At still faster kinetics ($J = 6.8\times10^{-4}$, the supersaturation-lowering effect is strong enough to produce a net depolarization of the electrode, *i.e.* ΔV^* becomes negative. At lower current densities, the magnitude of this effect approaches zero asymptotically. This case of $J = 6.8\times10^{-4}$ is the highest exchange-current density for which the computational procedure converged with δ less than 10^{-3}. Under more facile kinetics, ($J = 6.8\times10^{-3}$ and $J = 6.8\times10^{-2}$), there is significant depolarization at moderate current densities. The lowest values of ΔV^* calculated are roughly equal in magnitude and opposite in sign to the value of ΔV^* corresponding to Tafel kinetics, which translates to 30 mV for the present example.

A different vantage point is achieved by considering how the voltage increment depends on exchange-current density at a given current density (namely $\delta = 0.0205$, the base case). Figure 2-15 shows this behavior for cases with ($\Psi = 22{,}500$) and without ($\Psi = 0$) supersaturation effects. In the Tafel limit (low J or slow kinetics), the curves coincide as expected. With increasingly facile kinetics, ΔV^* drops for the no-supersaturation case and eventually approaches a constant value corresponding to the ohmic limit. With supersaturation effects included in the model, ΔV^* departs from the other curve as soon as the reaction departs from the Tafel regime (*i.e.* as soon as there is some appreciable reverse component to the electrode reaction). With increasing J the voltage increment drops steeply, passing through zero and continuing to negative values until the convergence limit of the method is reached. Fortuitously,

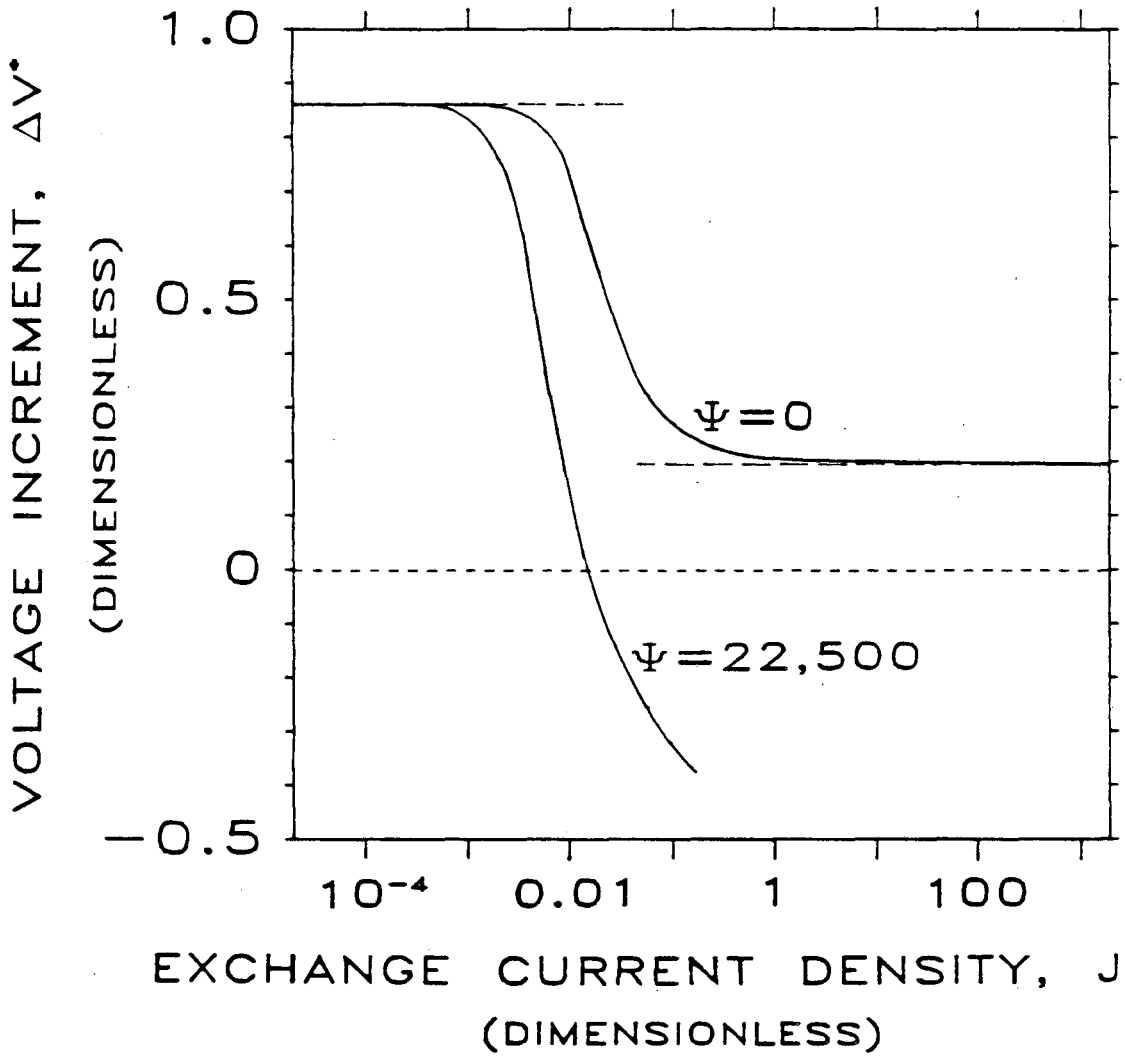


Figure 2-15. Voltage increment due to attached bubbles as a function of the dimensionless exchange-current density, J, *neglecting* supersaturation effects (upper curve), and *including* supersaturation effects (lower curve). (At base conditions).

ΔV^* becomes negative at roughly the value of J that corresponds to the activated cathode listed in Table 3.

It is important to recognize that, at the base conditions chosen to represent hydrogen evolution on conventional surfaces, the reaction is in the Tafel regime and hence there is no influence from supersaturation effects. Supersaturation effects may indeed be unimportant in many instances of present industrial practice. On cathodes that are more catalytically active (27,49) (for example, the activated cathode introduced in the previous section), a departure from Tafel kinetics is realized at these same operating conditions, and supersaturation effects do come into play. On such a surface, one can expect to see marked differences upon introduction of supersaturation effects into the model. Figures 2-16 and 2-17 illustrate these differences. In Figure 2-16, the dependence of voltage increment on interbubble spacing is shown. The supersaturation effect produces a net depolarization of the electrode over the entire range of bubble densities. The difference between the two curves, a measure of the magnitude of the concentration effect, is seen to increase with bubble density. In Figure 2-17, the effect of contact angle is shown for base conditions (including closest packing) on the activated cathode ($J = 0.011$). Again, when supersaturation effects are included, the model predicts a lower voltage increment due to attached bubbles, with ΔV^* negative for contact angles below 40 degrees. The magnitude of this voltage lowering is nearly independent of contact angle.

Conclusions

A model has been developed for predicting the current distribution and the increment in cell voltage caused by the presence of bubbles attached to the electrode surface.

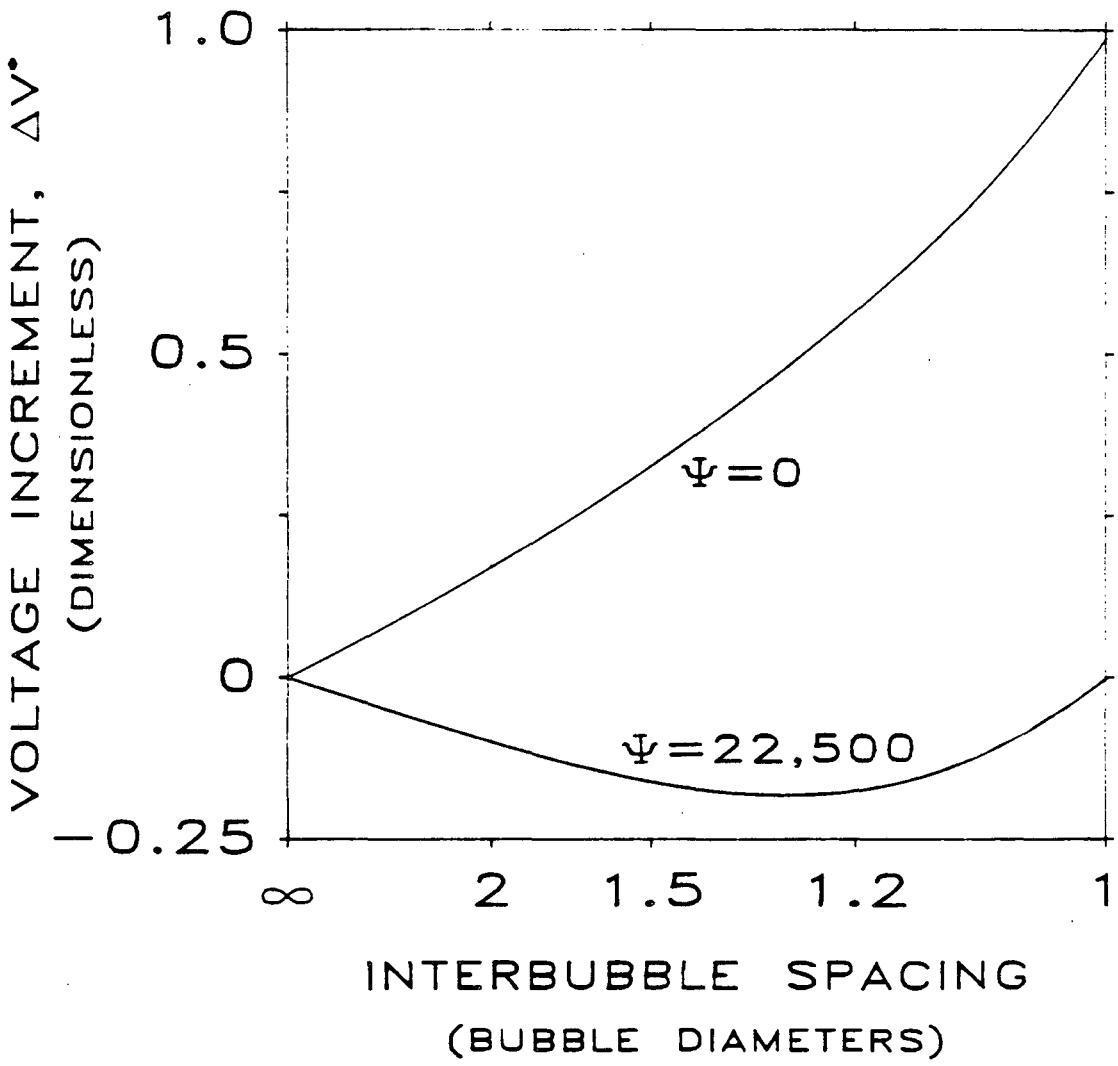


Figure 2-16. The effect of interbubble spacing on the voltage increment ΔV^* due to a hexagonal array of bubbles on an activated electrode ($J = 0.011$), with ($\Psi = 22{,}500$) and without ($\Psi = 0$) concentration effects. (At base conditions)

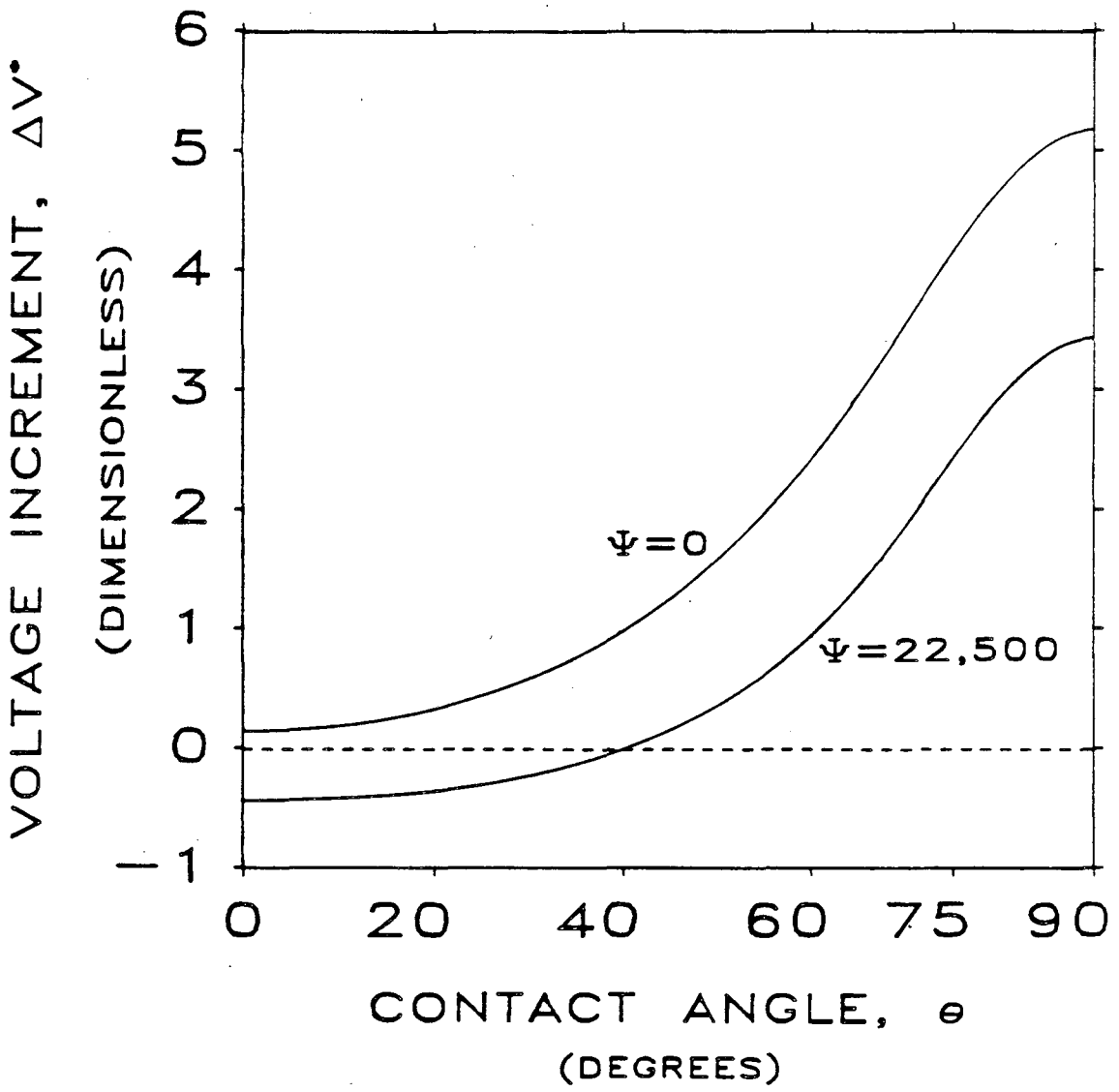


Figure 2-17. The effect of contact angle on the voltage increment ΔV^* due to a hexagonal array of bubbles on an activated electrode ($J = 0.011$), calculated at base conditions with ($\Psi = 22{,}500$) and without ($\Psi = 0$) concentration effects.

The model necessarily relies upon several idealizations of the actual condition at a gas-evolving surface. Accordingly, we do not emphasize its quantitative, predictive value but rather its usefulness as an aid to fundamental understanding of gas evolution.

In describing the geometric configuration of the bubble layer, great computational savings are achieved by approximating the hexagonal symmetry cell with a cylindrical one. This approximation is shown to be successful by comparison with analogue measurements of primary resistance (10).

When supersaturation effects do not influence the problem (either because they have been omitted or because they are precluded by irreversible electrode kinetics), both the current distribution and the resistance increment associated with attached bubbles can be accurately correlated, for a given geometric configuration, by a single parameter, the Wagner number. When supersaturation does influence the problem, the solution depends on five parameters (δ, J, $\frac{\alpha_a}{\alpha_c}$, Ψ, and s_g), and the model predicts a lower voltage increment than would be calculated without including concentration effects. In this case, the current distribution is strongly nonuniform, with highest current density to the electrode region nearest the bubble contact area. These effects depend strongly on electrode kinetics as well as on contact angle and on the density of the attached-bubble layer.

Calculations made at conditions typical for hydrogen evolution in the chloralkali industry reveal several things:

1) The ohmic penalty paid for attached bubbles is small compared to the extra voltage needed to drive charge transfer with part of

the electrode surface area masked by bubbles. To a good approximation, the voltage increment due to attached bubbles can be calculated solely on the basis of this area loss.

2) On conventional cathodes, operating in the Tafel region of electrode kinetics, the level of gas supersaturation (which may be significantly altered by attached bubbles) has no effect on cell voltage.

3) On cathodes of higher catalytic activity, where there is a departure from Tafel kinetics, the attached bubbles exert a supersaturation-lowering effect, which can decrease the overall voltage increment due to attached bubbles even to the extent that a net depolarization of the electrode is achieved.

References

1. R. E. Meredith and C. W. Tobias, in *Advances in Electrochemistry and Electrochemical Engineering,* Vol. 2, P. Delahay, Ed., pp. 15-48, Wiley Interscience, New York (1970).

2. H. Vogt in "A Comprehensive Treatise on Electrochemistry," Vol. 6, Eds. E. Yeager, J. O'M. Bockris, B. Conway and S. Surangapai, pp. 471-473, Plenum Press, New York (1981).

3. J. C. Maxwell, "Electricity and Magnetism," Vol. 1, 3rd ed., p. 440, Oxford (1892).

4. D. A. Bruggeman, *Ann. Physik.*, **24,** 636 (1935).

5. R. M. De La Rue and C. W. Tobias, *J. Electrochem. Soc.*, **106,** 827 (1959).

6. L. J. J. Janssen and E. Barendrecht, *Electrochim. Acta,* **28,** 341 (1983).

7. B. E. Bongenaar-Schlenter, L. J. J. Janssen, S. J. D. Van Stralen, E. Barendrecht, *J. Appl. Electrochem.*, **15,** 537 (1985).

8. H. Vogt, *J. Appl. Electrochem.*, **13, 87** (1983).

9. H. Vogt, *Electrochim. Acta*, **26,** 1311 (1981).

10. P. J. Sides and C. W. Tobias, *J. Electrochem. Soc.*, **129,** 2715 (1982).

11. P. J. Sides and C. W. Tobias, *J. Electrochem. Soc.*, **127,** 288 (1980).

12. O. Lanzi and R. F. Savinell, *J. Electrochem. Soc.*, **130,** 799 (1983).

13. F. Hine, M. Yasuda, R. Nakamura, and T. Noda, *J. Electrochem. Soc.*, **122,** 1185 (1975).

14. F. Hine, S. Yoshizawa, and S. Okada, *Denki Kaguku (J. Electrochemical Soc. Jpn.)*, **24,** 370 (1956).

15. P. J. Sides, *Electrochemical Society Extended Abstracts, B85-1,* (Spring Meeting, Toronto, Ontario, Canada, May12-17, 1985) pp. 600-601.

16. S. D. R. Wilson and A. Hulme, *Proc. R. Soc. Lond. A*, **387,** 133 (1983).

17. J. Newman, "Electrochemical Systems," pp. 340-341, Prentice Hall, Englewood Cliffs, NJ (1973).

18. L. E. Scriven, *Chem. Eng. Sci.*, **10,** 1-13 (1959).

19. E. Buckingham, *Trans. ASME*, **37**, 263-296 (1915).

20. J. Newman, "Electrochemical Systems," pp. 346-348, Prentice Hall, Englewood Cliffs, NJ (1973).

21. L. J. J. Janssen, J. J. M. Geraets, E. Barendrecht and A. D. J. van Stralen, *Electrochim. Acta*, **27,** 1 (1982).

22. W. M. Rohsenow and H. Choi, "Heat, Mass, and Momentum Transfer," p. 525, Prentice Hall, Engelwood Cliffs, New Jersey (1961).

23. S. U. Falk and A. J. Salkind, "Alkaline Storage Batteries," p. 588., Wiley, New York (1969).

24. P. Ruetschi and R. F. Amlie, *J. Phys. Chem.*, **70,** 718 (1966).

25. J. H. Perry, ed. "Chemical Engineers Handbook," 4th ed., McGraw Hill, New York (1963).

26. B. V. Tilak, P. W. T. Lu, J. E. Colman and S. Srinivasan, in "A Comprehensive Treatise of Electrochemistry," Vol. 2, J. O'M Bockris, B. E. Conway, E. Yeager, R. E. White, Eds., pp. 18-21, Plenum Press, New York (1981).

27. R. N. Beaver and S. L. Kelly, *Electrochemical Society Extended Abstracts,* **85-1,** p.599, (Spring Meeting, Toronto, Ontario, Canada. May 12-17, 1985).

28. L. J. J. Janssen, C. W. M. P. Sillen, E. Barendrecht and S. J. D. van Stralen, *Electrochim. Acta,* **29,** 633 (1984).

29. L. J. J. Janssen and E. Barendrecht, *Electrochim. Acta,* **30,** 683 (1985).

30. L. J. J. Janssen and J. G. Hoogland, *Electrochim. Acta,* **18,** 543 (1973).

31. J. P. Glas and J. W. Westwater, *Int. J. Heat Mass Transfer,* **7,** 1427-1443 (1964).

32. C. W. M. P. Sillen and S. J. D. van Stralen, "Proceedings of the 4th International Conference on Alternative Energy Sources," Miami (1981).

33. L. J. J. Janssen, *Electrochim. Acta,* **23,** 81 (1978).

34. L. J. J. Janssen and S. J. D. van Stralen, *Electrochim. Acta,* **26,** 1011 (1981).

35. M. G. Fouad and G. H. Sedahmed, *Electrochim. Acta,* **18,** 55-58 (1973).

36. C. A. Brebbia and S. Walker, "Boundary Element Techniques in Engineering," pp. 25-53, Newnes-Butterworths, Boston (1980).

37. C. A. Brebbia, "The Boundary Element Method for Engineers," p. 58, Wiley & Sons, New York (1978).

38. Allison Hooper, private communication (1985).

39. IMSL Library, Edition 9, IMSL, Inc., Houston, Texas (1982).

40. L. C. Wrobel and C. A. Brebbia, in "New Developments in Boundary Element Methods," Proceedings of the Second International Seminar, Southampton, pp. 77-89 (1980).

41. G. T. Symm, in "Boundary Element Techniques in Computer-Aided Engineering," C. A. Brebbia, Ed., pp. 101-110, Martinus Nijhoff, Boston (1984).

42. J. L. Blue. *The Bell System Technical Journal,* **57,** 2797-2822 (1978).

43. F. Yoshikawa and M. Tanaka, "Boundary Element Methods in Engineering," Proceedings of the Fourth International Seminar on Advances in BEM, Southampton, England, 1982, C. A. Brebbia, Ed., pp. 101-111, Springer-Verlag, Berlin (1982).

44. P. J. Sides and C. W. Tobias, *J. Electrochem. Soc.,* **132,** 583 (1985).

45. P. J. Sides, "Bubble Dynamics at Gas-Evolving Electrodes," Doctoral Thesis, University of California, Berkeley (1980).

46. H. Vogt, *Electrochim. Acta,* **25,** 527 (1980).

47. S. Shibata, *Bull. chem. Soc. Japan,* **36,** 53 (1963).

48. S. Shibata, *Bull. chem. Soc. Japan,* **33,** 1635 (1960).

49. E. Endo, H. Otouma and T. Morimoto, *Electrochemical Society Extended Abstracts* **85-1** pp. 614-615, (Spring Meeting, Toronto, Ontario, Canada. May 12-17, 1985).

50. H. Vogt, *Electrochim. Acta,* **30,** 265 (1985).

51. P. Cettou and C. W. Tobias, Report LBL-13632, University of California, Lawrence Berkeley Laboratory, 17 (1981).

LIST OF SYMBOLS

c_g	concentration of dissolved gas, mol/cm^3
c_g^{BULK}	concentration of dissolved gas in the bulk electrolyte, mol/cm^3
c_g^{SAT}	solubility of dissolved gas, mol/cm^3
a	bubble radius, cm
Dg	diffusivity of dissolved gas in electrolyte, cm^2/s

F	Faraday's constant, 96,487 C/equiv
h	radius of inscribed circle of symmetry hexagon, cm
i	current density, A/cm^2
i_{AVE}	average current density at electrode surface, A/cm^2
i_o	exchange current density of the electrode reaction A/cm^2
i_o^o	exchange current density with each reacting species at its bulk concentration, A/cm^2
J	dimensionless exchange current density
l	diffusion-boundary-layer thickness, cm
n	number of electrons participating in the electrode reaction
$\mathbf{n}$	unit vector normal to domain boundary, dimensionless
p	fraction of electrode area covered by projection of attached bubbles, termed s by Janssen, *et al.* (28)
r	distance from bubble centerline, cm
r_c	radius of equal-area cylinder, cm
R	universal gas constant, J/mol-K
s	distance between centers of neighboring bubbles in bubble diameters
s_g	stoichiometric coefficient of the gas species in the electrode reaction, after Newman (17), p. 172.
T	temperature, K
ΔV	voltage drop associated with the presence of attached bubbles, V
W	Wagner number
z	distance from electrode surface, cm
α_a	anodic transfer coefficient
α_c	cathodic transfer coefficient
δ	dimensionless average current density
∇^{2*}	Laplacian operator nondimensionalized by the characteristic length a
κ	electrolyte conductivity, $ohm^{-1}cm^{-1}$
ϕ	potential, V
ϕ_{METAL}	potential of the electrode, V
σ	fraction of electrode surface masked by bubbles

θ	bubble contact angle measured through liquid, degrees
η	total overpotential, V
η_s	surface overpotential, V
η_c	concentration overpotential, V
Ψ	proportionality between the normal derivatives of $c_g{}^*$ and ϕ^* at the electrode surface, dimensionless

Nondimensionalized Variables and Constants

$$c_g^* = \frac{c_g}{c_g^{SAT}}$$

$$\phi^* = \frac{\phi - \phi_{METAL}}{\frac{RT}{F}}$$

$$\delta = (\alpha_a + \alpha_c)\frac{i_{AVE}\, a}{\frac{RT}{F}\,\kappa}$$

$$J = (\alpha_a + \alpha_c)\frac{i_o^o\, a}{\frac{RT}{F}\,\kappa}$$

$$\Psi = \frac{\frac{RT}{F}\,\frac{s_g}{n}\,\kappa}{F\, D_g\, c_g^{SAT}}$$

$$r^* = \frac{r}{a}$$

$$\Delta V^* = \frac{\Delta V}{\frac{RT}{F}}$$

$$z^* = \frac{z}{a}$$

Chapter 3

Simulation of Leveling in Electrodeposition

ABSTRACT

A numerical model is developed for simulating the evolution of electrode microprofiles with deposition in the presence of a diffusion-controlled leveling agent. Current distribution is calculated by potential theory, and the electrode boundary is adjusted according to Faraday's law at each forward-Euler time step. The dependence of cathodic current density on surface overpotential and leveling-agent flux is expressed as an interpolation of electrode-kinetic data measured at several well-defined conditions of mass transfer. The agent-flux distribution is solved at each time step assuming steady-state diffusion within a concentration boundary layer. Both fields (potential and agent-concentration) are solved by quadratic boundary elements, and the boundary is advanced by a flexible algorithm. Application of the model is demonstrated in an example with coumarin in a Watts nickel bath: published polarization data are employed, and predicted leveling performance is compared to profilometric measurements reported in the same study (3).

Introduction

The purpose of this work is to develop a model for simulation of the electrode shape change in leveling: given polarization data at several conditions of mass transport, we seek to predict the evolution of an arbitrary electrode microprofile under specified conditions. The model employs a rigorous treatment of dilute-solution transport phenomena and direct curve fitting of measured kinetic data for the inhibited deposition reaction; the latter could be easily replaced by more fundamental kinetic relations should they become available.

Leveling can be defined as the attenuation of electrode-surface features smaller than 100 microns during electrodeposition. "True" leveling requires the presence of a dilute additive or *leveling agent*, and should be distinguished from "geometric leveling," which occurs under any condition of nearly uniform current density (1). Advances in the understanding of leveling are documented by several authors (1,2,3,4,5,6). The widely accepted explanation of the leveling effect is the "adsorption-diffusion theory" (1,7,8,9,10,11): deposition is preferentially *inhibited* on those portions of the electrode surface that are more accessible to the leveling agent by mass transport.

Much effort has been devoted to characterizing the leveling power of certain plating baths and developing a fundamental basis for the prediction of leveling performance. The relationship between leveling performance and the influence of mass-transfer condition on polarization behavior has been recognized qualitatively by a number of researchers (3,5,9,12,13,14,15,16,17). Several of these workers (7,8,9,18,19) have proposed theoretical treatments that allow one to predict leveling performance from polarization data at different concentrations or mass-transport conditions.

As early as 1957, Watson and Edwards (9) derived a relation between leveling

power and polarization curves measured at different bulk leveling-agent concentrations, c_A^{bulk}. They define leveling power as the normalized difference in current density between two points, "peak" and "recess," on the cathode profile:

$$Leveling\ \ Power = \frac{\Delta i}{i_{AVE}} \tag{1}$$

For low-amplitude features and high electrolyte conductivity, they propose

$$\frac{\Delta i}{i} = k \frac{c_A^{bulk}}{i} \frac{\partial \eta}{\partial c_A^{bulk}} \frac{\partial i}{\partial \eta}, \tag{2}$$

(We take the liberty of reexpressing their original equation in our nomenclature: i is current density, c_A^{bulk} is bulk leveling-agent concentration, η is surface overpotential.) The constant k characterizes the profile geometry and the choice of location for peak and recess. Watson and Edwards suggest evaluating k by the analogy between the concentration field and the primary current distribution, available for certain geometric configurations. They evaluate the partial derivatives in Equation 2 from families of polarization curves, each measured at the same hydrodynamic condition but at different c_A^{bulk}. Predicted leveling performance shows qualitative agreement with deposit-profile measurements for three leveling agents in a Watts nickel bath over a wide range of c_A^{bulk}.

In earlier work, Kardos and Foulke (7,8) present "a qualitative or crudely semi-quanititative method for predicting current distribution on microprofiles by substituting the polarization curve obtained with strong stirring for the polarization curve on a peak point and the curve obtained without stirring for the polarization curve on a recess point." Assuming uniform overpotential, they achieve fair agreement with profilometric measurements. They proceed to outline a more quantitative approach

based on evaluating the equivalent boundary-layer thickness[1] δ_N, at various points on the profile, by analogy to primary current distribution. Again, they assume uniform overpotential on the microprofile, affirming that this is approximately valid in typical plating baths. The proposed treatment is detailed more fully in a later publication by the same authors (4).

Kruglikov and Smirnova (18) review the aforementioned "indirect methods" for predicting leveling performance from polarization data, and develop a formula for the amplitude decay of a sinusoidal profile. Their basis is the analytic formula,

$$\frac{i_p - i_r}{i_{ave}} = \frac{4\pi H}{a} P , \tag{3}$$

where i_p and i_r refer to peak and recess, a is wavelength, and H is amplitude. The above follows by analogy to the primary current distribution and is restricted to $H << a \leq \delta_N^{ave}$. For diffusion-controlled leveling, an expression for P is postulated,

$$P = \left(\frac{\partial \ln i}{\partial \ln \delta_N} \right)_\phi , \tag{4}$$

and recognized by the authors as equivalent to the "leveling power" parameter of Watson and Edwards (9). Evaluating P in difference form from polarization measurements at two different agitation conditions (at which δ_N is obtained by limiting-current techniques with an indicator ion), Kruglikov and Smirnova found that the predicted leveling agrees fairly well with measurements by profilometry. These authors are the first to apply polarization data taken under well characterized conditions of mass transport. Kruglikov *et al.* (3,15) make extensive use of the rotating-disk electrode for this purpose.

---

[1] As clearly explained by Ibl (12), δ_N may deviate dramatically from the actual mass-transfer boundary-layer thickness.

The above methods suffer from some or all of the following limitations:

1. The full current distribution and profile shape change are not predicted. Only relative growth between two points on the profile is described.

2. The analogy to analytic solutions of primary distribution holds only when profile amplitude is much smaller than the diffusion-layer thickness.

3. Analytic solutions to primary-distribution problems are available only for the simplest geometries, and most of these become inapplicable once the profile has been modified by deposition.

4. A one-parameter description of the leveling effect is accurate only for small deviations in δ_N and i from their average values. Even low-amplitude profiles can defy this description.[2]

5. As we shall demonstrate later in this chapter, the assumption of uniform overpotential is only approximately correct.

Model

According to the adsorption-diffusion theory, leveling arises from the nonuniform supply of leveling agent to the electrode surface. This means that the agent is continuously *consumed* at the electrode, and that the rate of consumption is *influenced* by mass transfer. In fact, there is abundant evidence that mass transfer *controls* the rate of consumption of leveler, *i.e.* its codeposition is very facile, and is limited in rate by its arrival by convective diffusion. Citing articles (11,20,21,22,23) by five different

[2] For example, on a triangular-wave profile, diffusive flux is infinite to the peak and zero to the recess.

research teams, Ibl (12) states, "It can be concluded that for the three systems – deposition of nickel from a Watts bath containing thiourea or coumarin, deposition of copper from an acid copper bath with thiourea – the codeposition rates of coumarin and thiourea[3] are completely mass transport controlled, with $c_e = 0$." Moreover, it is *reasonable* to assume that an effective leveling system will operate under transport control: since some degree of transport influence is necessary for leveling, we would expect the highest leveling power in the limit of transport control. Our model will be based on the assumption of mass-transfer control.

The manner in which a leveling agent is consumed has been the subject of numerous studies (20,24,25,26,27,28,29,30,31). In the most general description, the leveling-agent A undergoes a series of transformations after reaching the electrode surface. These may include adsorption, reaction (often electroreduction) leading to one or more products, incorporation of the intact agent or its breakdown products in the growing deposit, desorption of products, further reaction of products, etc. A variety of adsorbed species, B_j^*, may reside on the electrode surface before desorbing, reacting, or being covered by fresh deposit.

It is logical to assume that, in the ongoing electrode process, each adsorbed species B_j^* will assume a steady-state surface concentration, $\Gamma_{B_j^*}$. For a given leveling system at a given temperature, these surface concentrations will be dictated by the detailed kinetics of the consumption reaction. In addition, they will depend, at most, on the following: the surface overpotential of the electrode, η; the supply rate of leveling agent, N_A; and the flux of depositing metal, N_M. Accordingly, we postulate the func-

[3] Ibl refers to coumarin and thiourea as "the typical leveling agents studied so far."

tional dependence:[4]

$$\Gamma_{B_j^*} = \Gamma_{B_j^*}(\eta, N_A, N_M). \tag{5}$$

discounting temporarily the possibility that η, N_A, and N_M are not independent.

Turning to the kinetics of the metal deposition, and acknowledging that the adsorbates B_j^* are responsible for the inhibition effect and assuming that the consumption reaction is not influenced by any foreign species or by the transport of desorbed products from the electrode, we propose the following functional dependence:

$$N_M = N_M(\eta, \Gamma_{B_j^*}). \tag{6}$$

Combining Equations 5 and 6, we arrive at the premise

$$N_M = N_M(\eta, N_A), \tag{7}$$

namely, that *the deposition rate depends exclusively on the overpotential and on the flux of the leveling agent.* This will hold regardless of whether mass transfer limits the consumption of leveler. However, if there were mixed control between mass transfer and kinetics, it would be impossible to determine N_A without knowing the detailed kinetics of the consumption reaction. Under mass-transport control, we can calculate N_A from transport considerations alone, ignoring consumption kinetics, which may be quite complicated and specific to a given leveling system.

Having postulated the above functional dependence for the inhibition effect, the next step is to supply the actual function. This can be derived from an analysis of the kinetics of inhibited deposition, or alternatively, constructed by interpolation from polarization measurements. We follow the latter approach, which offers a direct link

[4] It is noteworthy that this description applies equally to those levelers known to incorporate in the deposit and to those that do not. The sole criterion is that the leveling agent be consumed at the electrode surface at a rate dictated entirely by mass transport.

between measured kinetic data and the model's kinetics expression.[5] The rotating-disk electrode is ideally suited for such measurements, as the limiting flux of the leveling agent can be precisely controlled and modulated by changing rotation speed.

Leveling-agent mass transfer is idealized in terms of a Nernst boundary layer (12), a hypothetical stagnant region near the electrode surface, in which the leveling agent moves by diffusion alone, and beyond which concentration obtains its bulk value. In our examples, the Nernst layer envelops all electrode-surface features, and its outer boundary is smooth on the microscale.

Further, we make the "pseudosteady-state assumption:" transients in the concentration field die out much more quickly than the boundary moves. This holds whenever the time constant for deposition,

$$\tau_{DEP} = \frac{L\ n\ F\ \rho}{i\ M} \tag{8}$$

greatly exceeds the time constant for diffusion,

$$\tau_{DIF} = \frac{L^2}{D}. \tag{9}$$

In the examples presented in this chapter, typical of leveling systems, $\tau_{DEP} = 15{,}000$ seconds and $\tau_{DIF} = 3.4$ seconds.

Finally, we describe electrochemical transport by potential theory (32). This is appropriate when the conductivity is nearly uniform, as in a well supported plating bath at moderate current density.

Having made the foregoing assumptions, we can describe the system by the dual

[5] It is emphasized, however, that the former approach could be easily implemented in the numerical model with the modification of a single subroutine.

boundary problem stated in Table 1, which

Table 1: Problem Statement

Equation or Condition	Dimensional Form	Dimensionless Form	Domain or Boundary	Eq.
PDE A	$\nabla^2 c_A = 0$	$\nabla^{2*} c_A^* = 0$	Ω_A	(10)
BC 1A	$c_A = 0$	$c_A^* = 0$	Λ_1	(11)
BC 2A	$\frac{\partial c_A}{\partial \mathbf{n}} = 0$	$\frac{\partial c_A^*}{\partial \mathbf{n}^*} = 0$	Λ_{2A} , Λ_{4A}	(12)
BC 3A	$c_A = c_A^{bulk}$	$c_A^* = 1$	Λ_{3A}	(13)
PDE B	$\nabla^2 \phi = 0$	$\nabla^{2*} \phi^* = 0$	Ω_B	(14)
BC 1B	$\frac{\partial \phi}{\partial \mathbf{n}} = -\frac{nF}{\kappa} N_M(\eta, -\frac{\partial c_A}{\partial \mathbf{n}})$ where $\eta = -\phi$	$\frac{\partial \phi^*}{\partial \mathbf{n}^*} = -N_M^*(\eta^*, N_A^*)$ where $\eta^* = -\phi^*$	Λ_1	(15) (16)
BC 2B	$\frac{\partial \phi}{\partial \mathbf{n}} = 0$	$\frac{\partial \phi^*}{\partial \mathbf{n}^*} = 0$	Λ_{2B} , Λ_{4B}	(17)
BC 3B	$\frac{\partial \phi}{\partial y} = -\kappa\, i_{AVE}$	$\frac{\partial \phi^*}{\partial y^*} = -\delta$	Λ_{3B}	(18)

refers to the diagram in Figure 3-1. The solution procedure is first to calculate the leveling-agent concentration field, (BVP A). This supplies the profile of fluxes, N_A, at the electrode surface, which is required in the electrode boundary condition (Eq. 15) of

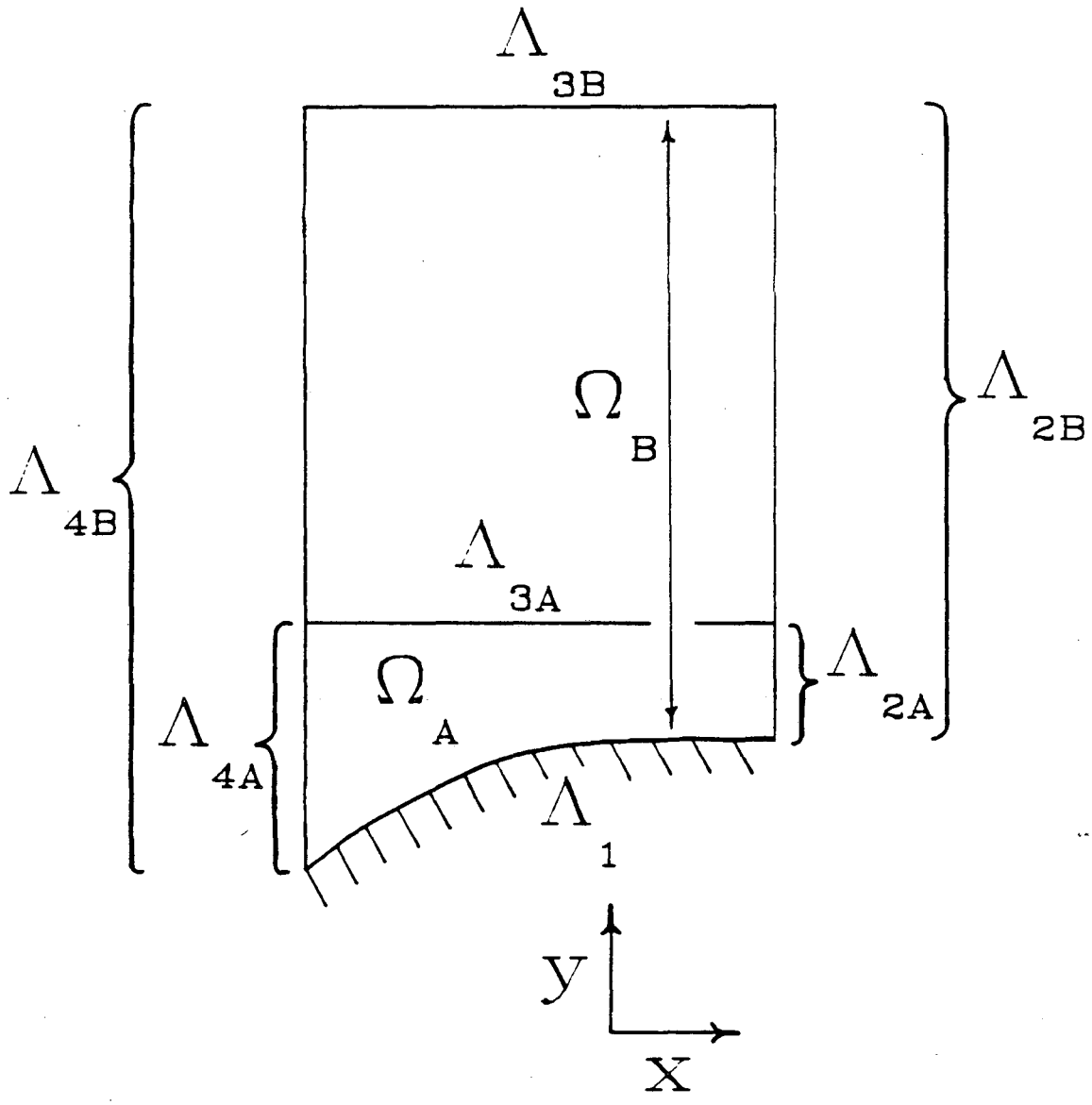


Figure 3-1. Geometric configuration of the dual boundary-value problem.

the potential-field problem (BVP B).

The concentration-field problem is the Laplace equation with zero flux at either of two parallel symmetry boundaries, Λ_{2A} and Λ_{4A}; concentration fixed at its bulk value at the boundary-layer edge, Λ_{3A}; and zero concentration at the electrode surface, Λ_1. The potential field must also obey Laplace's equation and satisfy the following conditions: no current may cross either symmetry boundary, Λ_{3B} and Λ_{4B}; current density is uniformly fixed far from the electrode, Λ_{3B}; and current density is related to surface potential at the electrode, Λ_1, by the overpotential expression, Eqs. 15 and 16.

Here we have assumed that

$$N_M = \frac{i}{nF}, \tag{19}$$

which is only approximate because the electroreduction of the leveling agent may consume a small fraction of the current.[6] Nevertheless, we shall treat the function $i\ (\eta, N_A)$ as virtually equivalent to $N_M\ (\eta, N_A)$.

Expressions for the function $i\ (\eta, N_A)$ can, of course, be alternatively determined from mechanistic models of the inhibited electrodeposition reaction. Obstacles to this approach are 1) the unavailability of such kinetic models for systems of interest, 2) the limited predictive power of existing treatments, and 3) wide variations in mechanism from system to system. A pragmatic alternative, the approach we follow, is to construct the expression by interpolation among measured kinetic data. These can be collected in a straightforward manner on a rotating-disk electrode: one need only record a series of polarization curves at a different value of N_A (varied by changing either rota-

[6] It would be no more difficult to account for this current component, except that, in some of the cases studied, the exact flux of leveling agent is not reported and/or the number of electrons transferred in this step is not known.

tion speed or bulk leveling-agent concentration). Such measurements are reported by Kruglikov, *et al.* (3,15) for several common leveling systems. A leveling model based on measured kinetic data offers generality and the promise of descriptive power for specific applications.

Our functional expression is a simple interpolation among experimentally measured values. Since our solution algorithm calls for an evaluation of η as a function of i and N_A, we interpolate the function $\eta\ (\ i\ ,\ N_A\ )$ rather than $i\ (\ \eta,\ N_A\ )$. (This keeps with the literature practice of recording electrode potential at given values of N_A and i, forming a rectangular grid in i-η space). We use *piecewise-bilinear interpolation*, a common finite-element functional representation, in which linear interpolation is done in both dimensions within the rectangle formed by the four surrounding points (33).

Numerical Method

The field problems are solved by the boundary-element method (BEM) using quadratic elements. We have followed Brebbia's formulation for potential problems (34,35). Figure 3-2 shows the nodal structure used for the simulation shown in Figure 3-6D. Integration is by six-point Gaussian quadrature. Additional details of our implementation are given elsewhere (36).

As the overpotential boundary condition (Eq. 15) is nonlinear, we use Newton-Raphson iteration to produce the converged solution. As an example, the solution corresponding to Figure 3-6D, obtained using the nodal structure of Figure 3-2, required six iterations on the first time step (with uniform overpotential as the starting iterate) but only two to three iterations on subsequent steps. The 100-step simulation required 4500 C.P.U. seconds on a VAX 8600 computer.

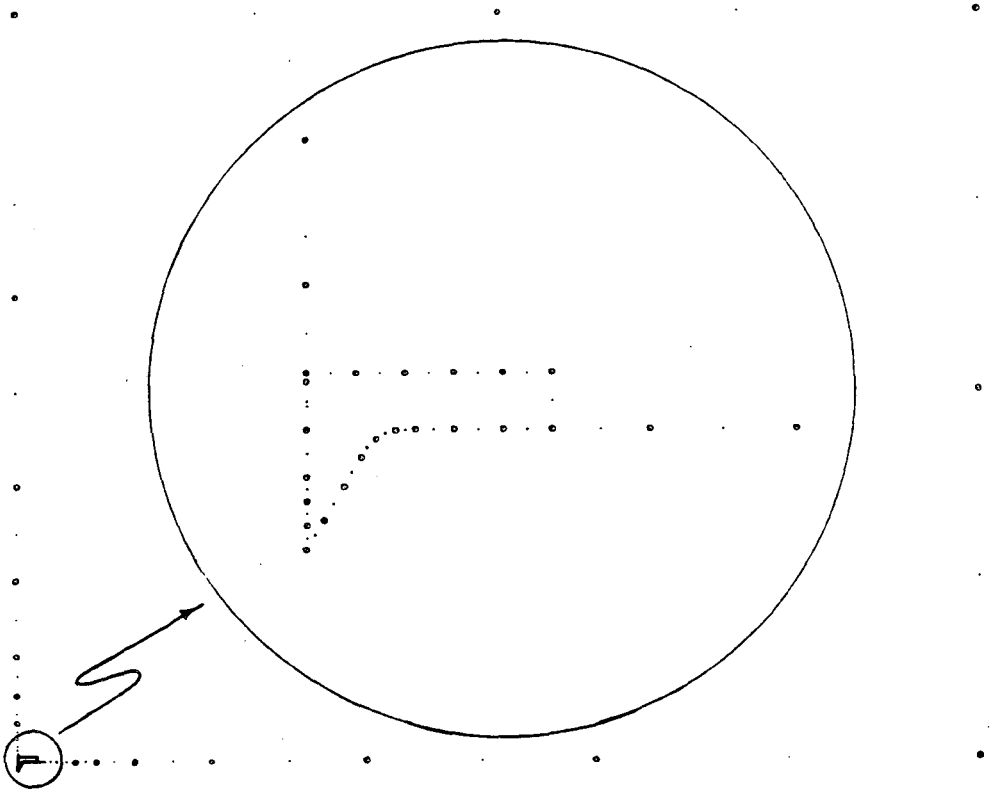


Figure 3-2. Boundary-element nodal structure for the grooved-electrode problem of Figure 3-6D.

The basis for the moving-boundary scheme is common to other electrode-shape-change studies (37,38,39,40,41,42,43,44,45, 46,47,48): in the course of a small time step, each point moves, in a direction normal to the surface, a short distance corresponding to the volume of metal that would deposit according to the local current density and Faraday's law. Within this description, there remain many alternatives for repositioning the nodes. The boundary-element method provides a certain advantage of simplicity, as there are no interior nodes to relocate. However, the algorithm developed for this study, illustrated in Figure 3-3, is still somewhat complicated because of its emphasis on versatility. The procedure involves five steps:

1) Move each node (plain dots) the prescribed distance normal to the surface (lower curve).
2) Construct a "false" surface (upper curve) by interpolating among these new nodes (X's).
3) Locate the new nodes (circled dots) by arranging them according to their original relative arc-length spacing on the portion of the false surface that lies within the symmetry boundaries (l_1 and l_2).
4) On each vertical symmetry boundary, reposition the nodes again preserving their original relative spacing.
5) Move each node of the boundary-layer edge vertically, maintaining a fixed distance between it and the highest point on the electrode profile.

Figure 3-3 illustrates only steps 1, 2 and 3, showing some of the subtleties of the algorithm. Node A moves outside the symmetry domain and is ultimately repositioned on the boundary by interpolation along the false surface. Node B is the border between two quadratic boundary elements, e_1 and e_2. As the profile slope may be discontinuous across this border, we begin by moving normal to each element, and then we locate a "compromise" false node at the intersection of perpendicular vectors as shown. Node

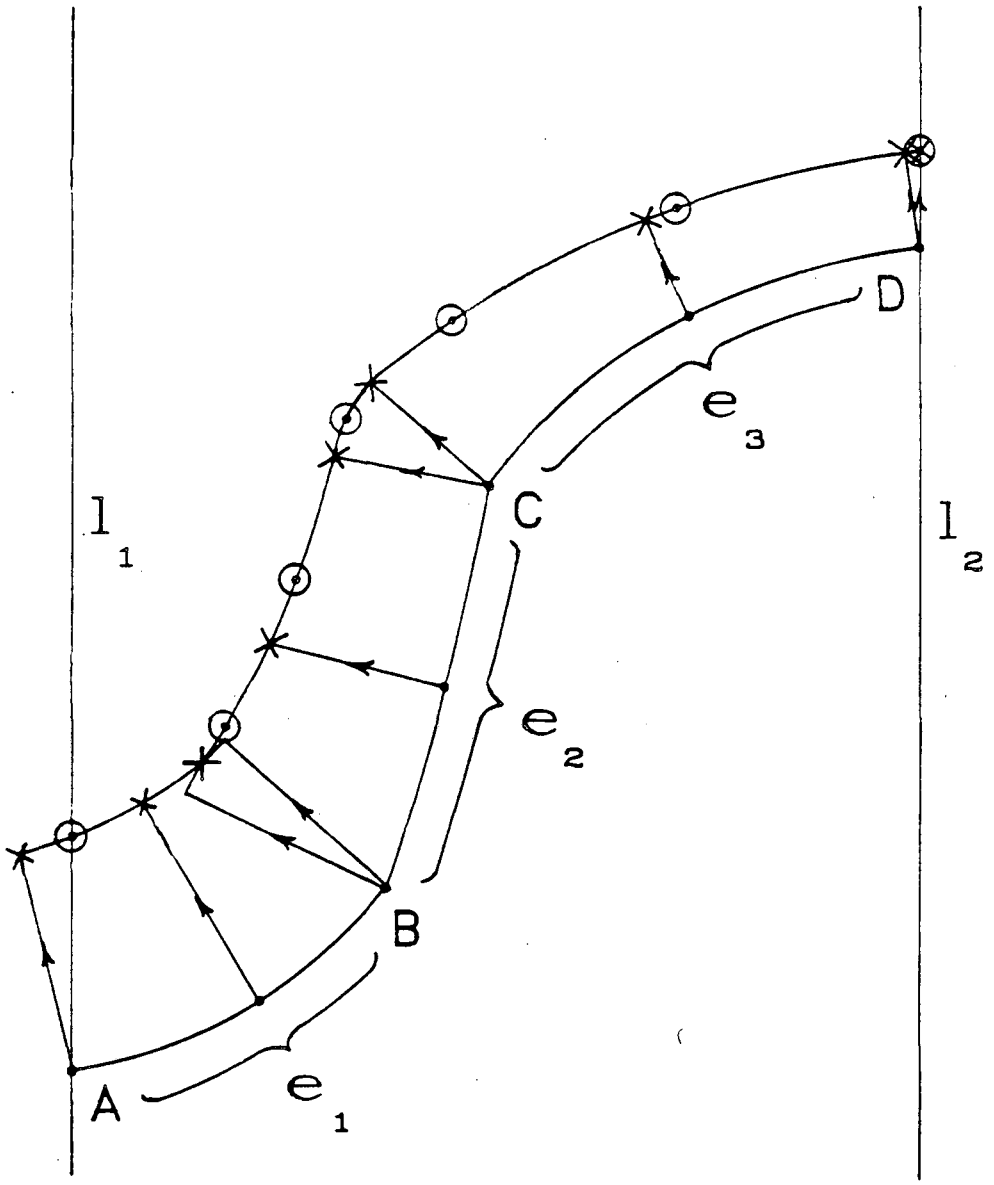


Figure 3-3. Exaggerated illustration of the scheme for repositioning the electrode-surface nodes in the moving-boundary algorithm

C is another "border" node, which, instead of overlapping, "diverges" into two, and we bridge these with a circular arc centered at C. Finally, node D strays inward from the symmetry line l_2, and we bridge this gap also with a circular arc.

The above scheme is flexible and especially advantageous for electrode profiles that do not meet the symmetry boundary at a right angle (*eg.* for triangular notches and waves), or for steep-walled or overhanging profiles. Other moving-boundary algorithms in the electrochemical literature are, as a rule, somewhat less flexible or more restrictive, especially those based on the domain methods (finite-element and finite-difference).

From the standpoint of numerical analysis, leveling problems are advantageous in that profile features do not grow. Accordingly, there is no tendency for the simulation to become unstable as in ordinary electrodeposition. Consequently, it is unnecessary to resort to smoothing routines such as those employed in various other simulations of electrodeposition.

For the purpose of testing the moving-boundary algorithm against a known analytic solution, a simulation of pure geometric leveling was performed on a semicircular profile as shown in Figure 3-4. The solution to this classical problem is a simple geometric construction with each semicircle growing evenly in radius from fixed centers. The challenge posed by this problem to the moving-boundary algorithm is to describe the semicircular recess as it vanishes. In the example shown, each symmetric section of the electrode boundary contains 27 nodes with pronounced refinement near the symmetry plane at the recess. Deviations in the simulation from the known solution are visible but not large. After depositing one profile amplitude, the deviation from the exact solution is below 0.5 percent.

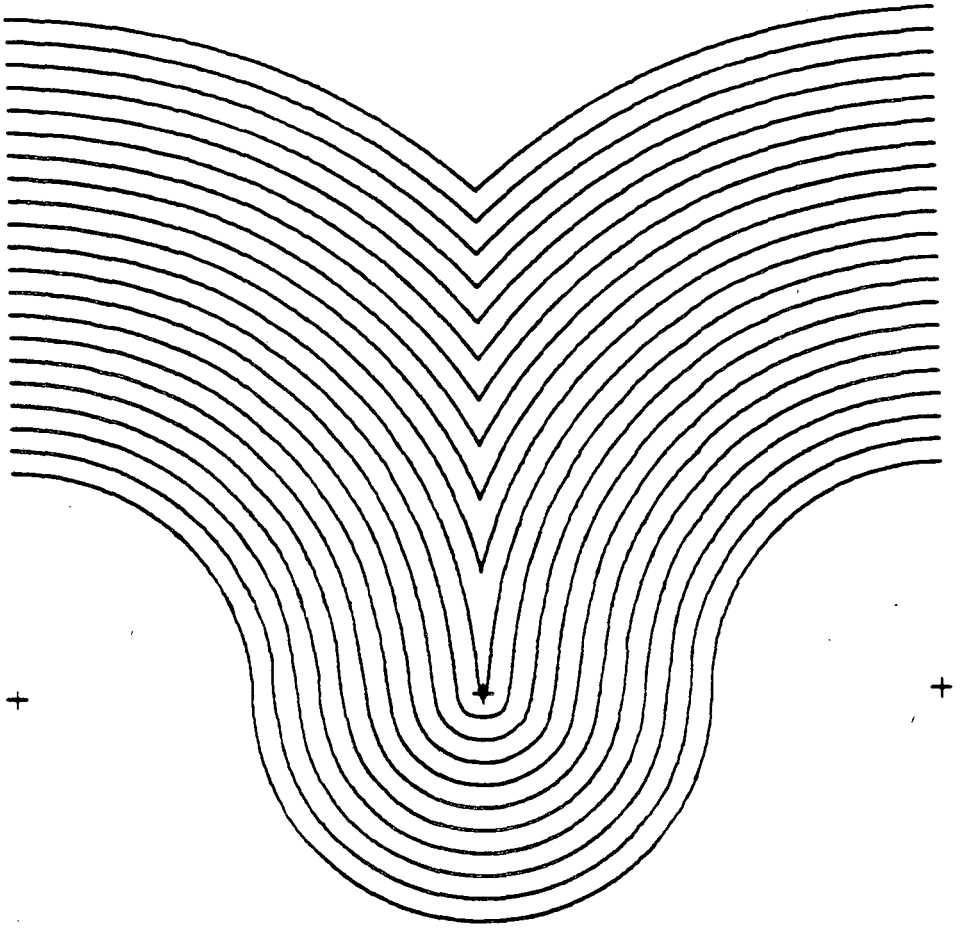


Figure 3-4. Demonstration of the moving-boundary algorithm for uniform current distribution on a semicircular electrode profile.

Demonstration of the Model

To illustrate how the model is used and to conduct a preliminary evaluation of its effectiveness, we apply it to a specific example. There are few published works containing both leveling performance measurements and polarization measurements on a given system at well defined conditions of mass transport. The case chosen for our example is coumarin in a Watts nickel bath as studied by Kruglikov, *et al.* (18). Specifically, we seek to simulate, using their measured polarization data shown in Figure 3-5, the leveling of a triangular groove, 50 microns in depth and with an included angle of 60 degrees. The bath contains 300 g/l $NiSO_4 \cdot 7H_2O$, 15 g/l NaCl, and 30 g/l H_3BO_3, at pH 4.5 and at 30 °C. The average current density is 10 mA/cm^2. Figure 3-6 shows simulated electrode-profile histories for four different leveling conditions. In panel A, no coumarin is present. Panels B, C and D are at a bulk coumarin concentration of 0.68 millimolar, at three different rotating-disk speeds: 150, 360 and 900 rpm. Using the Levich equation (49),

$$N_A = 0.620\, D_A^{\frac{2}{3}}\, \nu^{-\frac{1}{6}}\, \omega^{\frac{1}{2}}\, c_A^{bulk} \tag{20}$$

and a value of 5.01×10^{-4} cm-s$^{-\frac{1}{2}}$ for the constant[7]

$$K = 0.620\, D_A^{\frac{2}{3}}\, \nu^{-\frac{1}{6}}, \tag{21}$$

the corresponding effective boundary-layer thicknesses,

$$\delta_N = \frac{D_A\, c_A^{bulk}}{N_A^{lim}}, \tag{22}$$

were calculated: 36.5, 23.6 and 14.9 μm. This distance is used in the model as the diffusion-layer thickness away from the triangular groove.

---

[7] K is not reported directly by Kruglikov, *et al.* but is available from their conversion of the data.

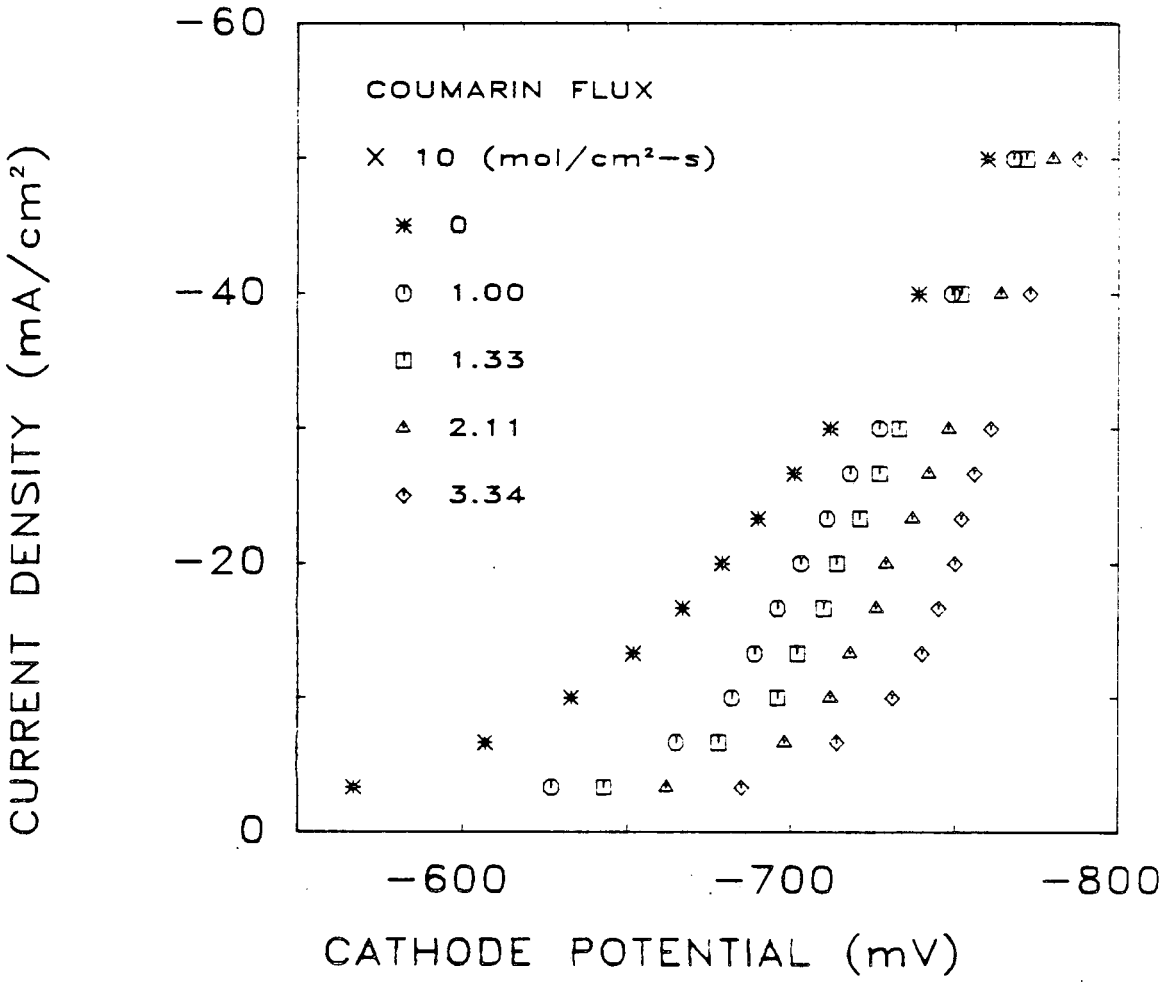


Figure 3-5. Polarization data of Kruglikov, *et al.* for coumarin in a Watts-nickel bath: current density *versus* cathode potential with leveling-agent flux, N_A, as a parameter.

Figure 3-2 gives the boundary-element nodal structure used for the simulation of Figure 3-6D. The domain extends far beyond the groove, approximating a semi-infinite expanse. Since the concentration-field solution becomes trivial several boundary-layer thicknesses beyond the groove, we truncate this domain, and simply impose the value of flux at the "cutoff" point over the remainder of the electrode surface. Dual nodes are employed at all sharp corners of the boundary. A total of 76 nodes make up the potential domain boundary, with 37 of these on the electrode surface; the concentration domain boundary contains 44 nodes, 19 of these on the electrode.

Each simulation pictured was carried out for 100 time steps with each step depositing an average thickness 0.01 times the original groove depth. From the starting profile (bottom) every fifth time step is shown.

The polarization data used in the model are plotted in Figure 3-5. Each marker type corresponds to a different limiting flux of coumarin, N_A^{lim}. Some of the points represent coinciding data from different combinations of rotation speed and coumarin concentration but corresponding to the same value of limiting flux, N_A^{lim}. Each point shown is used to construct the bilinear interpolation function, $\tilde{\eta}(i, N_A)$, used in the electrode boundary condition for all four of the simulations of Figure 3-6. Kruglikov, *et al.* made the measurements on a 0.6-cm rotating-disk electrode. Temperature, bath concentration, and current density were the same as for the leveling performance measurements. Each point was recorded after 1 to 2 minutes of steady-state electrolysis at the prescribed current density and rotation speed.

Panel A, the case with no leveler present, shows, as expected, the poor leveling performance associated with geometric leveling. Current density is at all times uniform to within 5 percent. Panels B, C and D show rising leveling performance with succes-

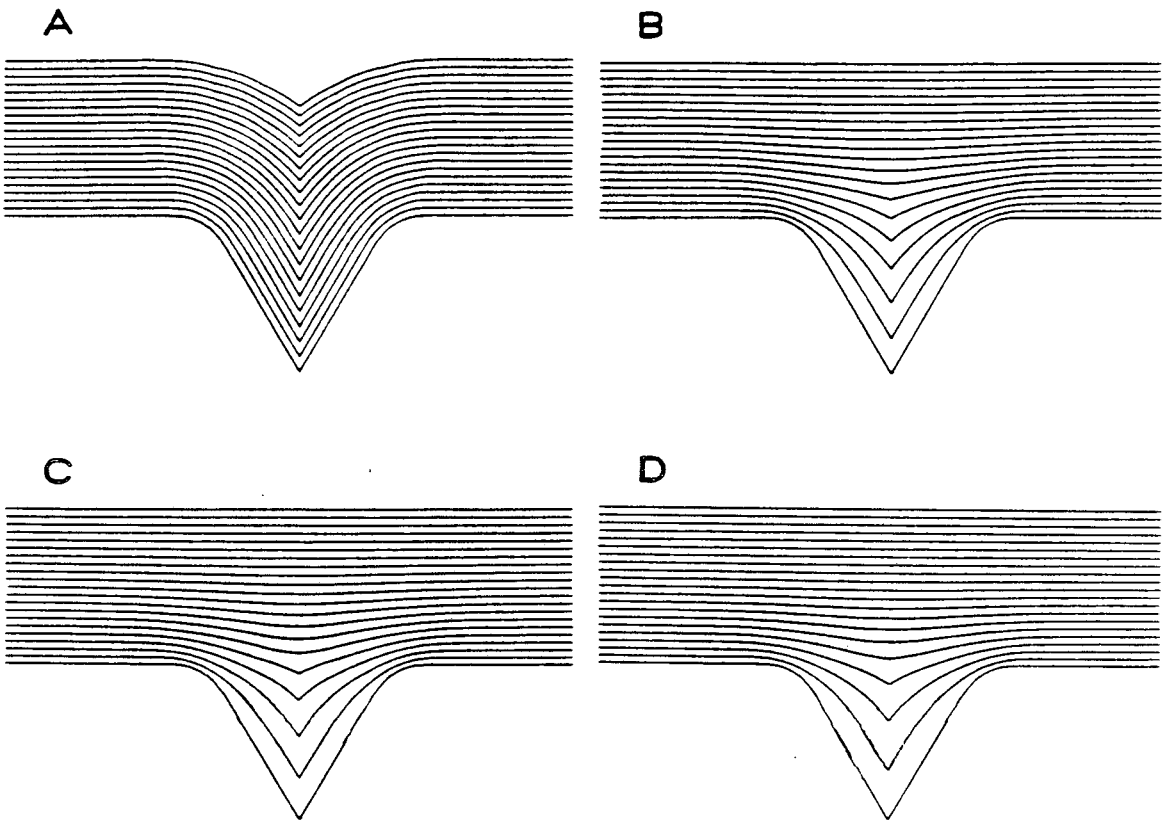


Figure 3-6. Simulated leveling of a triangular groove in a Watts-nickel bath with coumarin (i_{AVE} =10 mA/cm^2): A — no coumarin present; B, C and D — c_A^{bulk} = 0.68 mM at rotation speeds 150, 360 and 900 rpm.

sively decreasing average boundary-layer thickness. This trend is easily explained for the triangular-groove geometry since the local flux of inhibitor at the angular recess is fixed at zero, while the average flux increases; wider variations in flux N_A promote stronger leveling.

Figure 3-7 shows the decay of feature amplitude (groove depth) with time for the four cases pictured in Figure 3-6. Below 90-percent decay, each curve of the semilogarithmic plot is slightly sigmoidal in shape, but roughly in agreement with the linear dependence (exponential decay) reported by various experimenters (18,50). Beyond 90 percent, there is a departure from exponential decay, a deceleration of leveling. We have not seen any experimental data in this amplitude range for comparison.

The model's predictions are compared to the leveling measurements by Kruglikov, *et al.* in Figure 3-8. Leveling performance is reported as the percent decrease in groove depth after depositing 5 μm far from the groove. The literature data were taken at three rotation speeds (150, 360 and 900 rpm) over a range of coumarin concentration (0.3 to 2.7 millimolar). The simulations are for varying rotation speed at a concentration of 0.68 millimolar coumarin.

The model's prediction significantly exceeds the measured leveling performance over the range studied. (Simulations for N_A higher than 3.3×10^{-9} mol/cm^2–s require polarization data beyond the range provided by Kruglikov, *et al.*) The deviation is greater at higher coumarin flux. There are several possible explanations for the disparity. It may be that, in a groove as deep as 50 microns, there is some circulation of the electrolyte. This could substantially raise the local flux inside the notch above that predicted on the basis of diffusion alone, lessening the flux nonuniformity responsible for leveling. If this were the case, it would mean that the model should not be applied

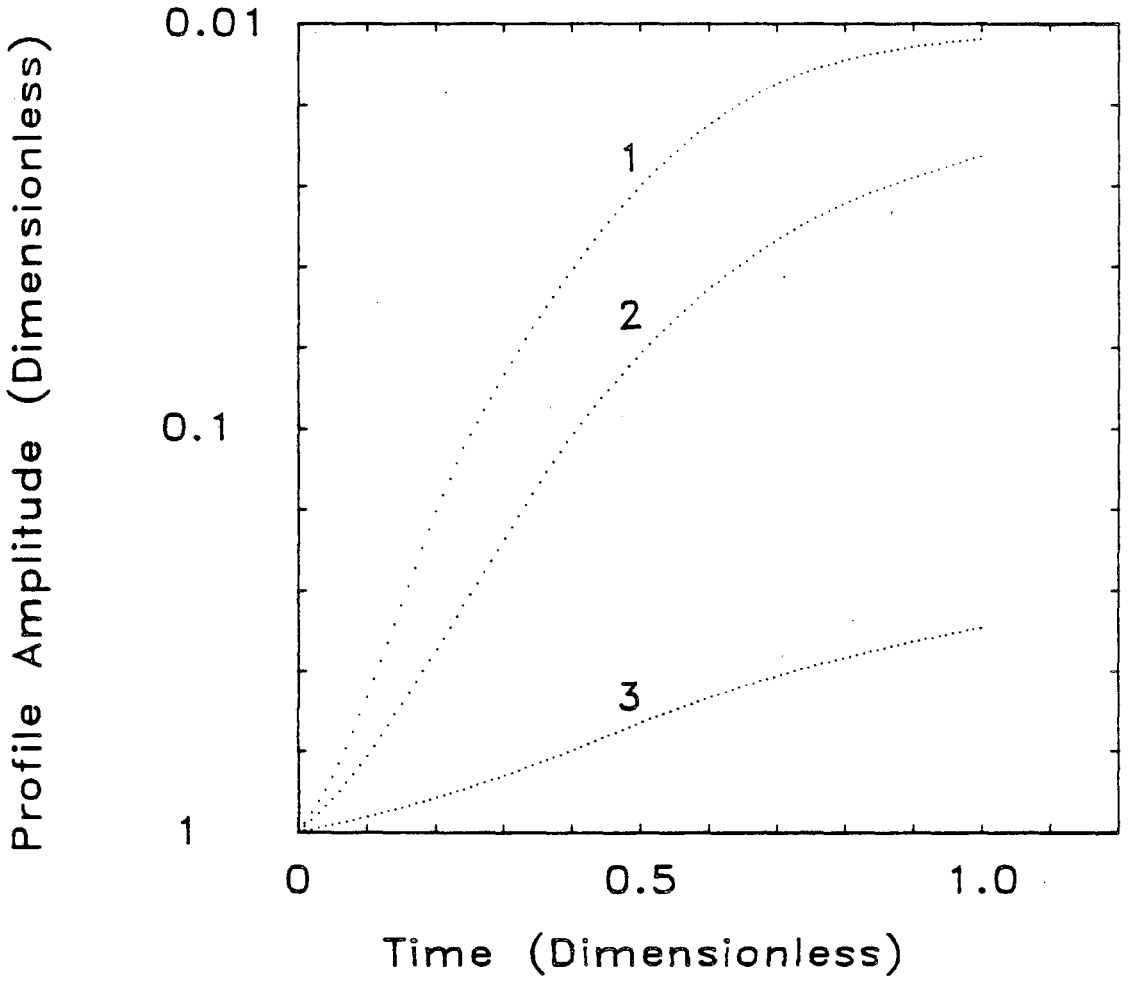


Figure 3-7. Decay in profile amplitude with time for the simulations of Figure 3-6: 1 and 2 — c_A^{bulk} = 0.68 mM, 900 and 150 rpm (Figs. 3-6D and 3-6B); 3 — no coumarin present (Fig. 3-6A).

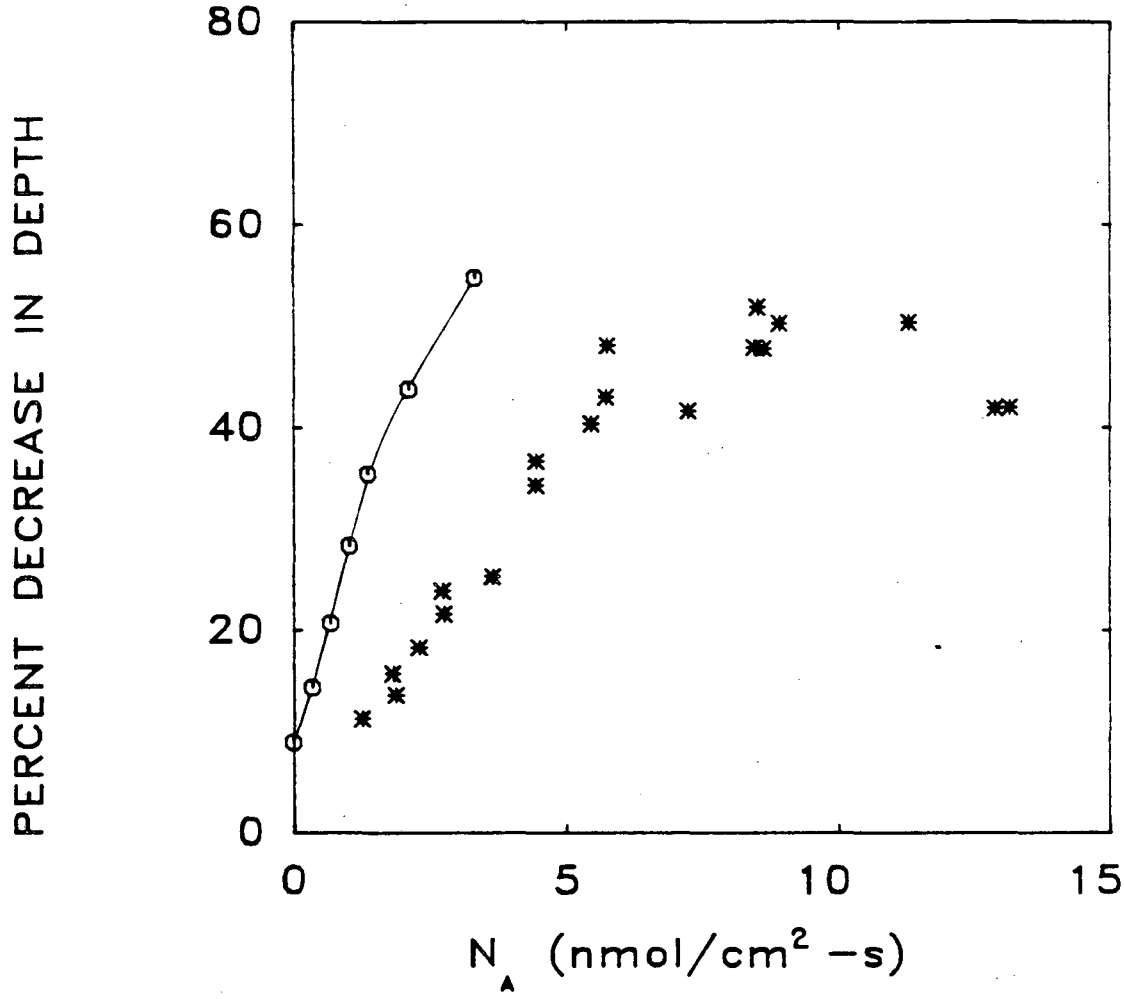


Figure 3-8. Comparison of leveling performance predicted by the model (connected circles) with measurements by Kruglikov, *et al.* (asterisks).

to such large features, or that its treatment of convective diffusion should be improved. Another possibility is that the polarization data used were not entirely compensated for ohmic resistance. Or, the surfaces of the two electrodes (one for polarization measurements and the other for profilometry studies) may not have been prepared in exactly the same way. Finally, the flat portion of the electrode may have been different from the groove walls, either in surface finish or because of the influence of crystallographic orientation on kinetics. It is not advisable to draw major conclusions about a model's value from a single comparison to experiment. The authors are confident that further tests of the model will produce better agreement, and we are certain that such comparative studies, and the refinements in the model that they suggest, will lead to a better understanding of leveling phenomena.

Conclusion

A model is advanced for simulation of leveling in electrodeposition based on the detailed dependence of surface overpotential on current density and leveling-agent flux. Using polarization data from one published study (18), the model predicts leveling that agrees with qualitative expectations, but exceeds the measured leveling performance of the experimental study. For conclusive evaluation and ultimate refinement of the model, further well defined experimental measurements of both polarization behavior and leveling performance will be required. The model promises to serve as a predictive tool in applications of electrodeposition, and as a conceptual device for evaluation and advancement of our present understanding of leveling.

References

1. O. Kardos and K. G. Foulke, "Advances in Electrochemistry and Electrochemical Engineering," P. Delahay and C. W. Tobias, Eds., vol. 2, John Wiley & Sons, New York, 1962.

2. S. S. Kruglikov, T. A. Smirnova, "Proceedings of the 8th Congress of the International Union for Electrodeposition and Surface Finishing," Basel, 1972, pp 105-109, Forester-Verlag, Zurich, 1973.

3. S. S. Kruglikov, N. T. Kudriavtsev, G. F. Vorobiova and A. Ya. Antonov, *Electrochimica Acta,* **10,** 253 (1965).

4. O. Kardos, Proceedings of "Surface 66," Basel, 1966, p. 62; *Galvanotechnik und Oberflachenschutz,* **8**, 161, 185 (1967).

5. E. B. Leffler and H. Leidheiser Jr., *Plating,* **44**, 388 (1957).

6. A. R. Despic and K. I. Popov in "Modern Aspects of Electrochemistry" No. 7, B. E. Conway and J. O'M. Bockris, Eds., pp 199-313, Plenum Press, New York, 1972.

7. D. G. Foulke and O. Kardos, *Proc. Amer. Electroplat. Soc.,* **43**, 172 (1956).

8. O. Kardos, *Proc. Amer. Electroplat. Soc.,* **43**, 181 (1956).

9. S. A. Watson and J. Edwards, *Trans. Inst. Metal Finishing,* **34**, 167 (1957).

10. S. A. Watson *Trans. Inst. Metal Finishing,* **37**, 144 (1960).

11. J. Edwards, *Trans. Inst. Metal Finishing,* **39**, 33, 45, 52 (1962).

12. N. Ibl, Proceedings of "Surface 66," Basel, 1966 p. 48; Galvanotchnik und Oberflachenshutz **7**: 256 (1966).

13. S. S. Kruglikov, N. T. Kudryavtsev, R. P. Sobolev, and A. Ya. Antonov, *Plating,* p. 78, January, 1966.

14. S. S. Kruglikov and L. M. Antipova, *Protection of Metals,* **6**, 81 (1970), (translated from *Zashchita Metallov,* **6**, 81 (1970)).

15. S. S. Kruglikov, N. T. Kudriavtsev, A. Ya. Antonov and A. V. Dribinski, *Trans. Inst. Metal Finishing,* **42**, 129 (1964).

16. T. Zak, *Trans. Inst. Metal Finishing,* **49**, 220 (1971).

17. Von G. Dreben and H. Wagner, *Metalloberflache,* **32**, 334 (1978).

18. S. S. Kruglikov, N. T. Kudryavtsev, G. I. Medvedev, and T. M. Izmailova, *Protection of Metals,* **8**, 668 (1972), (translated from *Zashchita Metallov,* **8**, 733 (1972)).

19. S. I. Krichmar, *Zh. fiz. khim.* **39**, 602 (1965); *Electrokhimiya,* **1**, 858 (1965).

20. G. T. Rogers and K. J. Taylor, *Electrochimica Acta.,* **8**, 887 (1963).

21. Yu. K. Vyagis, A. I. Bodnevas, and Yu. Yu. Matulis, *Zashchita Metallov (Protection of Metals)* **1**, 359 (1965); **2**, 201 (1966).

22. S. S. Kruglikov, Yu. I. Sinyakov, and N. T. Kudriavtsev, Electrokhimia **2**, 100 (1966).

23. Ph. Javet., N. Ibl, and H. Hintermann, *Electrochimica Acta,* **12**, 781 (1967).

24. S. E. Beacom and B. J. Riley, Research Report, General Motors Inc., GMR 325 (1961); *Galvanotechnik und Oberflachenshutz,* **3**: 206 (1962).

25. S. E. Beacom and B. J. Riley, *J. Electrochem. Soc.,* **106,** 309 (1959).

26. S. E. Beacom and B. J. Riley, *J. Electrochem. Soc.,* **107,** 785 (1960).

27. C. C. Roth and H. Leidheiser, Jr., *J. Electrochem. Soc.,* **100**, 553 (1953).

28. G. T. Rogers and K. J. Taylor, *Electrochimica Acta.,* **11**, 1685 (1966).

29. G. T. Rogers and K. J. Taylor, *Electrochimica Acta.,* **13**, 109 (1968).

30. L. Mirkova, St. Rashkov and Chr. Nanev, *Surface Technology* **15**, 181 (1982).

31. Yu. E. Gerenrot, L. A. Vaisburd, R. G. Dubenko, I. M. Bazavova, and P. S. Pel'kis, *Protection of Metals,* **9**, 206 (1973), (translated from *Zashchita Metallov,* **9**, 219 (1973)).

32. J. Newman, "Electrochemical Systems," pp. 340-341, Prentice Hall, Englewood Cliffs, NJ (1973).

33. K. H. Huebner, "The Finite Element Method for Engineers," John Wiley & Sons, New York, 1975.

34. C. A. Brebbia and S. Walker, "Boundary Element Techniques in Engineering," pp. 25-53, Newnes-Butterworths, Boston (1980).

35. C. A. Brebbia, "The Boundary Element Method for Engineers," p. 58, Wiley & Sons, New York (1978).

36. J. Dukovic and C. W. Tobias, submitted to *J. Electrochem. Soc., May, 1986.*

37. G. A. Prentice and C. W. Tobias, *J. Electrochem. Soc.*, **129**, 78 (1982).

38. G. A. Prentice and C. W. Tobias, *J. Electrochem. Soc.*, **129**, 317 (1982).

39. G. A. Prentice, Ph.D. Dissertation, University of California, Berkeley, Lawrence Berkeley Laboratory, LBL-11694 (December 1980).

40. J. B. Riggs, R. H. Muller, and C. W. Tobias, *Electrochimica Acta,* **26**, 961 (1981).

41. J. B. Riggs, Ph.D. Dissertation, University of California, Berkeley (YEAR).

42. R. Alkire, T. Bergh, and R. L. Sani, *J. Electrochem. Soc.* **125**, 1981 (1978).

43. R. Sautebin, H. Froidvaux, and D. Landolt, *J. Electrochem. Soc.* **127**, 1096 (1980).

44. C. Clerc and D. Landolt, *Electrochimica Acta,* **29**, 787 (1984).

45. E. C. Hume III, W. M. Deen, and R. A. Brown, *J. Electrochem. Soc.*, **131**, 1251 (1984).

46. J. Deconinck and G. Maggetto, *J. Electrochem. Soc.*, **132**, 2960 (1985).

47. J. Deconinck, Ph.D. Thesis, Vrije Universiteit Brussel, Brussels, Belgium (1985).

48. A. P. Peskin, Ph.D. Dissertation, University of Colorado, Boulder, Colorado, (1985).

49. V. G. Levich, "Physicochemical Hydrodynamics," Prentice-Hall, Englewood Cliffs, N.J., 1962.

50. S. I. Krichmar and A. Ya. Pronskaya, *Protection of Metals*, **4**, 657 (1968), (translated from *Zashchita Metallov*, **4**, 745 (1968)).

LIST OF SYMBOLS

a	wavelength of sinusoidal profile, cm
Bj^*	species adsorbed to the electrode
c_A	concentration of leveling agent mole/cm^3
c_A^{bulk}	concentration of leveling agent in the bulk electrolyte, mole/cm^3
c_e	leveling-agent concentration at the electrode surface (nomenclature of Ibl (12)), mole/cm^3
D_A	diffusivity of leveling agent in electrolyte, cm^2/s
F	Faraday's constant, 96,487 C/equiv
H	amplitude of sinusoidal profile
i	current density, mA/cm^2
i_{AVE}	average current density away from the electrode surface, mA/cm^2
i_p , i_r	local current densities at 'peak' and 'recess' of electrode profile, mA/cm^2
k	constant characterizing profile geometry
K	Levich constant
L	characteristic length of electrode profile, cm
M	atomic mass of depositing metal, g/mol
n	number of electrons participating in the deposition reaction
N_A	flux of leveling agent, mole/cm^2–s
N_A^{lim}	limiting flux of leveling agent, mole/cm^2–s
N_M	flux of metal, mole/cm^2–s
P	parameter expressing leveling power, dimensionless
$\mathbf{n}$	unit vector normal to domain boundary, dimensionless
x	horizontal distance, cm
y	vertical distance, cm
$\Gamma_{B_j^*}$	surface concentration of adsorbed species j, mole/cm^2
δ_N	equivalent boundary-layer thickness, cm

δ_N^{ave}	average equivalent boundary-layer thickness, cm
∇^{2*}	Laplacian operator nondimensionalized by the characteristic length L
κ	electrolyte conductivity, $\text{ohm}^{-1}\text{cm}^{-1}$
η	surface overpotential, mV
$\tilde{\eta}$	interpolated surface overpotential, mV
ν	kinematic viscosity, cm^2/s
ϕ	potential, mV
ρ	density of depositing metal, g/cm^3
ω	rotating-disk speed, radian/s

Appendix A-1

Boundary-Element Code for Active-Passive Model

The following Fortran code was used in the calculations of Chapter 1. A sample input-data file is given at the end of the program listing. Important variables are defined in subroutine 'input' or else as they appear in the code.

```
      program becur

c     This program calculates the current distribution in an
c     electrochemical cell.  The cell configuration can be either
c     planar or axisymmetric.

c     Documentation:
c       Input variables and parameters are explained as they are
c       read in the ''input'' subroutine.  All other variables are
c       introduced as they appear in the program.

c     Polarization:
c       The model allows for polarization
c       of one electrode by any of four kinetic relations: 1) Butler-
c       Volmer kinetics, 2) linear kinetics, 3) Tafel kinetics, and
c       4) active-passive kinetics using a trapezoidal curve fit
c       the active-pasive polarization curve.

c     Sign Convention:
c       The open-circuit potential of the cell is effectively
c       subtracted out of the problem by referring all potenitials
c       to their open-circuit values.  The working electrode (the
c       electrode at which we consider polarization) is always taken
c       to be at zero potential.  Under this system, the anode is
c       always the more positive electrode.  As always, current
c       and overpotential are positive at the anode and negative
c       at the cathode.  Current density at an electrode is
c       opposite in sign to the dot product of the potential
c       gradient and the inward-pointing surface normal vector.

c     Method:
c       Laplace's equation is solved by quadratic boundary elements.
c       Solution of the matrix equation is by Gauss elimination with
c       full pivoting; the forward-reduction step is carried out only
c       once.

c     Convergence:
c       The boundary conditions may be applied either as
c       essential or natural conditions (except for the active-passive
c       case which is natural only) so that the more stable iteration
c       scheme can be chosen for a given problem.

c     Library Subroutines:
c       The method requires evaluation of Eliptic Integrals.
c       As written, the code relies on the IMSL library for this.
c       (''link code, sys_imsls/lib'' on the LBL CSA cluster of VAX's)
c       Alternatively, the Elliptic integrals can be evaluated
c       within the code (albeit less accurately) using functions
c       EI1 and EI2.
```

```
      implicit double precision (a-h,o-z)

      common/nr1/ surf, delsurf
      common/piv/ ipivot(150), jpivot(150)
      common/set0/nonelec
      common/set1/nonods
      common/set2/coords(150,2)
      common/set2a/keyloc(150)
      common/set3/iaxism,rinner
      common/set4/a(150,150)
      common/set5/c(150,150)
      common/set6/x(150)
      common/set7/bc(150)
      common/set8/keybc(150),keyode(150)
      common/set10a/tol,fracfp,fracnr
      common/set10b/mctr,mctrmax,mswitch,mconsec
      common/set10c/nctr,nctrmax,ndivmax
      common/set10d/ncon,nconmax
      common/set10e/ntot,ntotmax
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set14/ikin
      common/set14a/xchcur,alphaa,alphac
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set21/ibctype
      common/set22/irestrt,ikeep
      common/set24/igrid,xmin,xmax,ymin,ymax

      dimension b(150), surf(150), surfnew(150), delsurf(150)

c     Read in all data and print it out if iprint = 1.
      call input

c     If you want a plot of the nodal structure, make it here.
c     Subroutine ''grid'' uses the graphics package ''DI3000.''
      if ( igrid . ge . 1 ) then
        call grid(nonods,coords,keyode,keyloc)
        if ( igrid . eq . 2 ) stop
        endif

c     See if this run is a restart ( irestrt = 1 ). If so,
c     skip the costly ''matrix'' and ''gauss1'' routines and
c     read in their products from the restart file, for003.dat.

10    if ( irestrt . eq . 1 ) then
```

```
        call rsread

        else

c       Formulate the matrix problem (once and for all.)
        call matrix(nonods,coords,keyloc,A,C,keybc,keyode)

c       Do forward reduction on the matrix (once and for all.)
        call gauss1(nonods,A,ipivot,jpivot)

        endif

c     If requested, write the contents of pivot vectors and a and
c     c matrices into a restart file, for008.dat.

      if ( ikeep . eq . 1 )   call rswrite

c     We now have the forward-reduced form of the system matrix
c     and we're ready start into the iteration cycle.

c     Initialize surf (either phi or its derivative at
c     the electrode surface).  Judicious starting values
c     were assigned to bc(i) in the input routine.
      do 100 i = 1, nonelec
        surf(i) = bc(i)
100     continue

c     Carry out the iterative scheme for convergence.
c     This involves calling the ``continue'' routine,
c     which in turn calls the ``converge'' routine.

      call continue

c     Tell the world what you've calculated.
      call output

      stop
      end



c     ******************* SUBROUTINE INPUT **************
      subroutine input

      implicit double precision (a-h,o-z)

c     This subroutine reads in all the input data and, if
c     desired, prints it all back out again as a check.
```

```
      common/set0/nonelec
      common/set1/nonods
      common/set2/coords(150,2)
      common/set2a/keyloc(150)
      common/set3/iaxism,rinner
      common/set7/bc(150)
      common/set8/keybc(150),keyode(150)
      common/set10a/tol,fracfp,fracnr
      common/set10b/mctr,mctrmax,mswitch,mconsec
      common/set10c/nctr,nctrmax,ndivmax
      common/set10d/ncon,nconmax
      common/set10e/ntot,ntotmax
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set14/ikin
      common/set14a/xchcur,alphaa,alphac
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set21/ibctype
      common/set22/irestrt,ikeep
      common/set24/igrid,xmin,xmax,ymin,ymax


c     *** First, give all run directions:

      read (1,300)
300   format(////)

c     Read iprint (print all data unless iprint equals zero),
c     ispotpr (values are spot-printed every 'intspot'
c     iterations if ispotpr = 1), and intspot.
c     Also read igrid, key for plotting grid.
c          igrid=0          run, no plot
c          igrid=1          run and plot
c          igrid=2          plot, no run
      read (1,301) iprint, ispotpr, intspot, igrid
301   format(///i1,9x,i1,9x,i3,10x,i1)

c     Is this run a restart of an earlier one?
c     ( yes: irestrt = 1;  no: irestrt = 0 )
c     Do we want to save matrix values and pivot keys
c     in a restart file?  ( yes: ikeep = 1;  no: ikeep = 0 )
      read (1,302) irestrt, ikeep
302   format(///i1,29x,i1)


c     *** Next, give instructions for convergence, etc.:
```

```
c       In which direction shall we iterate around the loop?
c       Read the value of ibctype (1: essential bc;  0: natural)
        read (1,303) ibctype
303     format(///i1)

c       To help convergence, we judiciously initialize the
c       electrode value (be it potential ''guessv'' in mV or
c       current density ''guessi'' in mA/cm2).
        read (1,304) guessv, guessi
304     format(///e10.3////e10.3)

c       The value of the relative convergence tolerance.
        read (1,305) tol
305     format(///e10.3)

c       The fraction of the new value taken in the relaxed
c       substitution, ''fracfp'' for the fixed-point
c       iteration, and ''fracnr'' for the
c       Newton-Raphson iteration:
        read (1,306) fracfp, fracnr
306     format(///f10.8,7x,f10.8)

c       The max allowed fixed-point iterations, ''mctrmax'',
c       the value of ''mctr'' at which we switch over to
c       Newton-Raphson, ''mswitch'' (less than ''mctrmax''):
c       and ''mconsec'' the number of consecutive decreases
c       in ''tmax'' required to switch to N.R.:
        read (1,3065) mctrmax, mswitch, mconsec
3065    format(///i4,7x,i4,7x,i4)

c       The number of Newton-Raphson iterations allowed, ''nctrmax,''
c       the number of diverging iterations before quitting, ''ndivmax,''
c       and the total allowed number of Newton-Raphson iterations:
        read (1,307) nctrmax, ndivmax, ntotmax
307     format(///i4,7x,i4,7x,i4)

c       Some continuation paramters:
c       The conservative starting value of delv, ''delv0'',
c       the initial increment in delv, ''deldelv'', (both in mV),
c       the number of continuation steps allowed, ''nconmax'',
c       and the maximum allowed increment, ''delpmax'':
        read (1,3073) delv0, deldelv, nconmax, delpmax
3073    format(///f8.1,7x,f8.3,7x,i4,7x,f8.1)

c       Initialize ''delv'' to ''delv0.''
        delv = delv0

c       *** Next, describe the electrochemical problem:
```

```
c     What is the applied voltage between the electrodes?
c     (The ultimate delv approached in the continuation scheme)
c     This is the counter-electrode potential minus the
c     working-electrode potential, MINUS the value of that same
c     potential difference that would prevail at open circuit.
c     (expressed in millivolts.)
c     (The ''working electrode'' is the one with the overpotential.)
      read (1,3075) nodelvf
3075  format(///i2///)
      read (1,3077) ( delvf(i), i = 1, nodelvf )
3077  format(8x,f8.1)

c     Which kinetic law shall we use?  1=Butler-Volmer;
c     2=linear; 3=Tafel; and 4=active-passive.
      read (1,308) ikin
308   format(///i1)

c     To characterize the kinetics in cases 1, 2 and 3,
c     we input three parameters:

c       Dimensionless exchange current density (Newman's J), (+)
        read (1,309) xchcur
309     format(///e10.3)

c       Ratio the transfer coefficients, alphaa and alphac.
        read (1,310) alphaa, alphac
310     format(///f7.4,10x,f7.4)

c     For active-passive runs, we need a nondimensionalized
c     polarization curve.  For convenience, we read in the
c     curve in mA/cm2 vs Volts, along with the characteristic
c     current density and voltage.

c       The number of cur-eta points for the trapezoidal fit.
        read (1,311) nopts
311     format(///i2)

c       The cur-eta pairs (mA/cm2 and mV)
        read (1,312)
312     format(///)
        read (1,313) ( curap(i), etaap(i), i = 1, nopts )
313     format(5x,f10.6,2x,f6.1)

c       The characteristic current density in mA/cm2, always
c       positive:  (This is conductivity times RT/F divided
c       by characteristic length)
        read (1,314) curchar
314     format(///e10.3)
```

```
c       The characteristic voltage (RT/F) in millivolts:
        read (1,315) vchar
315     format(///f7.2)


c     *** Lastly, all geometric description:

c     Read iaxism.  If this reads zero, the problem is not
c     axisymmetric.  A value of one or anything else indicates
c     axisymmetry.
      read (1,316) iaxism
316   format(///i1)

c     The number of nodes in the nodal structure.
      read (1,317) nonods
317   format(///i3)

c     The number of nodes on the electrode surface.
      read (1,318) nonelec
318   format(///i3)

c     Read in the scale factor for the model representation
c     of the domaim.  SCALE is the number of model length units
c     per dimensionless length unit.  Since we have
c     nondimensionalized the problem with respect to
c     ''charl,'' SCALE is ''charl'' expressed in model
c     units.  Generally, we take scale > 1 to get the
c     coordinates in round numbers.
      read (1,319) scale
319   format(///f7.1)

c     Vertices of smallest rectangle that encloses the domain
c     (for graphics use.)
      read (1,320) xmin, xmax
320   format(////f7.1,1x,f7.1)
      read (1,321) ymin, ymax
321   format(//f7.1,1x,f7.1)

c     The value of rinner.  This is the distance (in the
c     distance units of the model) from the innermost point
c     of the nodal structure to the centerline.  If the
c     problem is not axisymmetric, we don't need
c     rinner, but we must read a value anyhow.
      read (1,322) rinner
322   format(///f9.1)

c     Read the coordinates of each node in global order.
c     As a convention, we number counter clockwise beginning
c     with the electrode.  Also read in the keys for the
```

```
c     boundary condition type (''keybc''), the boundary-segment
c     type (''keyode'') and the node location (''keyloc'').
      read (1,323)
323   format(///)
      read (1,324) (coords(i,1),coords(i,2),keybc(i),keyode(i),
     $   keyloc(i),i=1,nonods)
324   format(8x,f9.1,1x,f9.1,4x,i2,2x,i2,2x,i2,4x)


c     Since this program is general to two different
c     iteration schemes, we set the values of keybc
c     here according to the value of ibctype:
c     keybc = 1 for ibctype = 1,  and keybc = 1
c     for ibctype = 1.  Thus it doesn't matter what
c     is read in for keybc1 on the electrode above.
      do 326 i = 1, nonelec
        keybc(i) = ibctype
326     continue

c     Initially fill the boundary-condition vector with zeros.
      do 330 i = 1, nonods
        bc(i) = 0.
330     continue

c     Set any node on the counter electrode to 'delv' (dimensionless)
      do 331 i = 1, nonods
        k = keyode(i)
        if ( k . eq . 2 ) then
          bc(i) = delv0 / vchar
          endif
331     continue

c     Initialize electrode values whether they are potentials
c     or gradients.  Later you might want to change these initial
c     values to aid convergence.
c     Also set keybc=1 for ibctype=1; keybc=0 for
c     ibctype=0 on the electrode.
      do 333 i=1,nonelec
        if ( ibctype . eq . 1 ) then
          bc(i)     =  guessv / vchar
          keybc(i)  =  1
        else
          bc(i)     = -guessi / ( curchar * scale )
          keybc(i)  =  0
        endif
333     continue


c     Set the values of the gaussian quadrature abscissas and
```

```
c       weighting factors.

c       Six-point gaussian quadrature coefficients:

        nogauss = 6

        abgauss(1) = -0.932469514203152
        abgauss(2) = -0.661209386466265
        abgauss(3) = -0.238619186083197
        abgauss(4) =  0.238619186083197
        abgauss(5) =  0.661209386466265
        abgauss(6) =  0.932469514203152

        wfgauss(1) =  0.171324492379170
        wfgauss(2) =  0.360761573048139
        wfgauss(3) =  0.467913934572691
        wfgauss(4) =  0.467913934572691
        wfgauss(5) =  0.360761573048139
        wfgauss(6) =  0.171324492379170

        return
        end



c       **************** SUBROUTINE CONTINUE *********************
c       This is the master routine for the iterative solution scheme.
c       To give the best chances for convergence, it uses first-order
c       continuation on the parameter ''p'': the ''converge'' routine
c       is called each time the value of ''p'' has been advanced.
c       Many problems will not require recourse to continuation;
c       these can be solved by a single call to the ''converge'' routine.

        subroutine continue

        implicit double precision (a-h,o-z)

        common/nr1/ surf
        common/set0/nonelec
        common/set1/nonods
        common/set4/a(150,150)
        common/set5/c(150,150)
        common/set6/x(150)
        common/set7/bc(150)
        common/set8/keybc(150),keyode(150)
        common/set10a/tol,fracfp,fracnr
        common/set10b/mctr,mctrmax,mswitch,mconsec
```

```
      common/set10c/nctr,nctrmax,ndivmax
      common/set10d/ncon,nconmax
      common/set10e/ntot,ntotmax
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/p, p0, nopf, pf(40), delp, delpmax
      common/set21/ibctype
      common/set30/fprime(150,150),ipivot4(150),jpivot4(150)

      dimension b(150), surf(150), surfp(150), surf0(150)
      dimension dxdp(150), dfdp(150), shift(150)
      dimension fprime0(150,150),ipivot0(150),jpivot0(150)


c     Within this routine, the continuation parameter ``delv''
c     is called ``p.'' (see common set15.)

c     Count the Newton-Raphson iterations:
      ntot = 0
c     Count the continuation steps:
      ncon = 1
c     This will go to unity when we've reached ``delvf:''
      kf = 0
      ipf = 1
c     Numerical perturbation:
      pert = 0.000001

c     If this is the very first continutation step, you are
c     not starting from a converged solution.  In this case,
c     jump down to line 250 where ``converge'' is called.
      if ( ncon . eq . 1 ) go to 250


c     ***** Top of the ``continuation loop'' *****
100   ncon = ncon + 1

c     Form ``df/dx'' and ``df/dp'' based on the
c     present converged solution.

c     Temporarily perturb bc:
      do 110 i = 1, nonods
        if ( keyode(i) . eq . 2 ) bc(i) = bc(i) + pert
110     continue

c     Evaluate perturbed surface values, ``surfp:''
      if ( ibctype . eq . 0 ) call natural(surf,surfp,0)
      if ( ibctype . eq . 1 ) call essential(surf,surfp)

c     Evaluate the derivative df/dp, knowing that f=0 (since
c     we have recently converged to a low tolerance).
```

```
      do 120 i = 1, nonelec
        fpert = surfp(i) - surf(i)
        dfdp(i) = fpert / pert
120     continue

c     Restore bc to its unperturbed value:
      do 130 i = 1, nonods
        if ( keyode(i) . eq . 2 )  bc(i) = bc(i) - pert
130     continue

c     Now obtain the derivative dx/dp by solving the
c     algebraic system ``FPRIME * dx/dp = df/dp''
c     using the solver ``gauss2.''  (Recall that we have
c     already forward-reduced FPRIME in the ``newton''
c     routine by calling ``gauss1.'')

      call gauss2(nonelec,fprime,dxdp,dfdp,ipivot4,jpivot4,0)

c     In case you diverge on this ``p'' step and need to backtrack,
c     make a 'copy' of ``surf''.  (The 'original' will be
c     modified each time by the ``converge'' routine.)
      do 150 i = 1, nonelec
        surf0(i)  =  surf(i)
150     continue

c     Now revise ``delp'' based on the number of
c     Newton-Raphson iterations for the last iteration:
      if ( nctr . lt . 3 )  delp  =  2.0  * delp
      if ( nctr . eq . 3 )  delp  =  1.00 * delp
      if ( nctr . gt . 3 )  delp  =  0.56 * delp
      if ( dabs(delp) . gt . dabs(delpmax) )  delp = delpmax

c     Advance ``p:''
      pold = p
      p = p + delp

c     Near the end, take only as big a step as you need:
      test = ( p - p0 ) / ( pf(ipf) - p0 )
      if ( test . ge . 1.0 ) then
        p  = pf(ipf)
        kf = 1
        delp = pf(ipf) - pold
        endif

      nbak = 0
c     ***** Top of the ``backtrack loop'' *****
200   keycon = 0

c     Quit if you've done too many total iterations:
```

```
      if ( ntot . ge . ntotmax ) then
        write (2,210) ntotmax
210     format('Exceeded ntotmax =',i4,1x,'iterations.')
        return
        endif

c     Quit if ``delp'' has gotten ridiculously small:
      if ( dabs(delp) . le . 0.1 ) then
        write (2,212)
212     format(/'continuation has stalled.  deldelv < 0.1'/)
        return
        endif

c     Set the ``surf'' vector to its original value for this
c     continuation step.  (It may have been changed
c     by calls to the ``converge'' routine.)
      do 215 i = 1, nonelec
        surf(i)  =  surf0(i)
215     continue

c     Determine ``shift,'' the change in ``surf'' for this step:
      do 220 i = 1, nonelec
        shift(i)  =  - dxdp(i) * delp / vchar
220     continue

c     Update ``surf'' and the boundary-conditon vector, ``bc'':
      do 230 i = 1, nonods
        if (keyode(i) . eq . 1)  surf(i) = surf(i) + shift(i)
        if (keyode(i) . eq . 2)    bc(i) = p / vchar
230     continue

c     Call the ``converge'' routine, which tries to produce
c     the converged solution (using Newton-Raphson) for
c     the new value of the continuation variable, ``p.''
250   keycon = 0
      call converge(keycon)

c     If this is the first continuation step and it doesn't
c     converge, you have to stop.
      if ( keycon . eq . 0 . and . ncon . eq . 1 ) then
        write (2,260)
260     format('The very first step never converged.')
        stop
        endif

c     If, on any other step, you failed to converge,
c     backtrack and take a smaller ``p'' step.
      if ( keycon . eq . 0 ) then
```

```
c       Count the number of times you have to backtrack, ``nbak.''
        nbak  =  nbak  +  1
        write (2,270) nbak
270     format(/'Backtracking',3x,i3,'th time')

c       Revise values of ``delp'' and ``p:''
        pold = p - delp
        frac  =  0.5
        delp  =  frac * delp
        p  =  pold  +  delp

c       Return to the top of the ``backtrack loop.''
        go to 200
        endif


c     If the ``converge'' routine does succeed, keep
c     going with the continuation scheme:

      if ( kf . eq . 1 . and . nbak . eq . 0 ) then
        write (2,280) pf(ipf)
280     format(/'You have your answer.  delv = ',f8.2)
        write (4,285) pf(ipf)
285     format('delv = ',f8.1)
        call plotfile
        kf = 0
        if ( ipf . eq . nopf ) return
        ipf = ipf + 1
        endif

      if ( ncon . gt . nconmax ) then
        write (2,290)
290     format(/'Exceeded nconmax iterations.')
        return
        endif

c     Return to the top of the ``continuation loop:''
      go to 100

      end



c     ***************** SUBROUTINE CONVERGE *********************
c     This iterative routine produces the converged solution
c     for a given value of the continuation parameter ``p.''

      subroutine converge(keycon)
```

```
      implicit double precision (a-h,o-z)

      common/nr1/ surf, delsurf
      common/piv/ ipivot(150), jpivot(150)
      common/set0/nonelec
      common/set1/nonods
      common/set2/coords(150,2)
      common/set2a/keyloc(150)
      common/set3/iaxism,rinner
      common/set4/a(150,150)
      common/set5/c(150,150)
      common/set6/x(150)
      common/set7/bc(150)
      common/set8/keybc(150),keyode(150)
      common/set10a/tol,fracfp,fracnr
      common/set10b/mctr,mctrmax,mswitch,mconsec
      common/set10c/nctr,nctrmax,ndivmax
      common/set10d/ncon,nconmax
      common/set10e/ntot,ntotmax
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set14/ikin
      common/set14a/xchcur,alphaa,alphac
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set21/ibctype
      common/set30/fprime(150,150),ipivot4(150),jpivot4(150)

      dimension b(150), surf(150), surfnew(150), delsurf(150)


c     Initialize parameters for iteration, convergence, divergence tests.

c     Number of iterations:
      mctr    = 0
      nctr    = 0
      kconsec = 0
c     Number of iterations for which ''tmax'' has gone UP minus
c     the number of times it has gone DOWN:
      ndiv    = 0
c     If tmax ever gets this high, quit:
      tmaxmax = 1000000.
c     Largest change in x on any node in a Newton-Raphson step:
      tmax    = 100.
      if ( mctrmax . eq . 0 ) then
        keyiter = 1
        else
```

```
        keyiter = 0
        endif

c     ***************** Start of Iteration ********************
c     Perform an iteration, i.e. operate on ``surf'' to
c     give ``surfnew.''

c     Heading for output:
120   if ( nctr . eq . 0 . and . mctr . eq . 0 ) then
        write (2,122) ncon
122     format(i4,'th continuation step:'//3x,'mctr'6x,
     $    'nctr',8x,'tmax',8x,'ncon',6x,'ntot')
        endif

c     If it's time, switch from fixed-point to Newton-Raphson:
      if ( mctr . ge . mswitch . and . nctr . eq . 0 ) then
        keyiter = 1
        write (2,125)
125     format('switching to N.R.')
        endif

c     Decide which type of iteration to do: fixed-point
c     or Newton-Raphson:

c     Choice of fixed-point iteration:
      if ( keyiter . eq . 0 ) then

        mctr = mctr + 1
        mtot = mtot + 1

c       If you've exceeded the max allowed iterations, say so and quit.
        if ( mctr . ge . mctrmax  ) then
          write (2,180) delv, mctr, nctr, deldelv
          write (2,130)
130       format ('You exceeded nctrmax iterations.')
          keycon = 0
          return
          endif

c       Call the ``fixedpoint'' routine to operate on the vector
c       ``surf'' returning a vector of corrections ``delsurf''
c       (the difference between ``surfnew'' and ``surf'')

        call fixedpoint

        endif


c     Choice of Newton-Raphson iteration:
```

```
      if ( keyiter . eq . 1 ) then

        nctr = nctr + 1
        ntot = ntot + 1

c       If you've exceeded the max allowed iterations, say so and quit.
        if ( nctr . ge . nctrmax  ) then
          write (2,180) delv, mctr, nctr, deldelv
          if ( nctrmax . eq . 0 ) then
              write (2,135)
135           format('divergence, no n.r. called for')
            else
              write (2,140)
140           format ('You exceeded nctrmax iterations.')
            endif
          keycon = 0
          return
          endif

c       Call the ``newton'' routine to operate on the vector ``surf'',
c       returning a vector of corrections ``delsurf.''
        call newton

        endif


c     Regardless of the iteration scheme used to obtain ``delsurf'',
c     obtain the corrected vector ``surfnew''.
      do 145 i = 1, nonelec
        surfnew(i) = surf(i) + delsurf(i)
145     continue


c     ******************* Substitution  **********************
c     Here we make the susbstitution (relaxed if elected, and
c     the weighting factor may be different for a fixed-point
c     iteration)
c     to produce ``surf'' for the next iteration.
c     At the same time, we find the biggest change in ``surf''
c     on the electrode for the convergence test.

      tmaxold = tmax
      tmax    = 0.
c     Find ``tmax,'' the largest relative change in ``surf'':
      do 150 i = 1, nonelec
        t = dabs ( surfnew(i) - surf(i) ) / dabs( surf(i) + 2.0e-6 )
        if ( t . gt . tmax ) then
          tmax = t
          endif
```

```
          if ( keyiter . eq . 0 ) then
              surf(i) = fracfp * surfnew(i) + (1.0 - fracfp) * surf(i)
            else
              surf(i) = fracnr * surfnew(i) + (1.0 - fracnr) * surf(i)
            endif
150       continue

c       As a diagnostic, print out tmax for each iteration:
        write (2,160) mctr, nctr, tmax, ncon, ntot
160     format(3x,i4,6x,i4,6x,g10.4,6x,i4,6x,i5)

c       *************** Convergence Test ****************
c       If you have converged, return to the ``continue'' routine.

        if ( tmax . lt . tol ) then
c         Integrate to obtain the total cell current:
          call curint(curtot)
          write (2,170)
170       format('converged')
          write (2,180) delv, mctr, nctr, deldelv, curtot
180       format('delv=',g14.8,3x,'mctr=',i3,3x,'nctr=',
     $      i3,3x,'deldelv=',g10.4/
     $      'if 1 unit = 1 cm, curtot in mA =',g14.8//)
          keycon = 1
c         call output
          return
          endif

c       Test for divergence two different ways:

c       1) If tmax exceeds tmaxmax then bail out.
        if ( tmax . ge . tmaxmax ) then
          write (2,180) delv, mctr, nctr, deldelv
          write (2,190)
190       format( /'divergence.  tmax exceeds tmaxmax.'/ )
          keycon = 0
          return
          endif

c       2) Keep a running tally of the pattern of convergence
c       in the Newton-Raphson iteration scheme:  If tmax goes
c       UP ndivmax times more than it goes DOWN over any
c       interval, announce this and quit.
        if ( keyiter . eq . 1 ) then
          if ( tmax . ge . tmaxold ) ndiv = ndiv + 1
          if ( tmax . lt . tmaxold ) ndiv = ndiv - 1
          if ( ndiv . le . 0       ) ndiv = 0
          if ( ndiv . ge . ndivmax ) then
            write (2,180) delv, mctr, nctr, deldelv
```

```
          write (2,195)
195       format( /'divergence.  ndiv exceeds ndivmax.'/ )
          keycon = 0
          return
          endif
        endif

c     If requested ( i.e. if mconsec < mctrmax )
c     switch to Newton-Raphson when tmax has gone down
c     ``mconsec'' consecutive times.
      if ( keyiter . eq . 0 ) then
        if ( tmax . lt . tmaxold ) then
            kconsec = kconsec + 1
          else
            kconsec = 0
          endif
        if ( kconsec . ge . mconsec ) then
          keyiter = 1
          write (2,200)
200       format('switching to N.R. - ten decreases in a row')
          endif
        endif


c     If you have neither converged nor shown signs
c     of divergence, do another iteration.
      go to 120

      end




c     ***************** SUBROUTINE FIXEDPOINT *********************
c     This routine simply goes through the cycle...

      subroutine fixedpoint

      implicit double precision (a-h,o-z)

      common/nr1/ x(150), delx(150)
      common/set0/nonelec
      common/set21/ibctype

      dimension xp(150), xpnew(150), b(150), f(150)

      do 5 i = 1, nonelec
        xp(i) = x(i)
```

```
5       continue
      keepoff = 0
      if ( ibctype . eq . 0 )  call natural(xp,xpnew,keepoff)
      if ( ibctype . eq . 1 )  call essential(xp,xpnew)

      do 10 i = 1, nonelec
        delx(i) = xpnew(i) - xp(i)
10      continue

      return
      end



c     ***************** SUBROUTINE NEWTON *****************
c     This routine does multivariable Newton-Raphson on the
c     function evaluated in either one of the routines
c     ``natural'' or ``essential'').  The scheme is
c     ``xnew = g(xold)'' which we solve by defining the
c     function ``f(x) = g(x) - x'', and then solving the
c     system ``FPRIME * delx = - f'' for the vector of
c     corrections, ``delx.''

      subroutine newton

      implicit double precision (a-h,o-z)

      common/nr1/ x(150), delx(150)
      common/piv/ ipivot(150), jpivot(150)
      common/set0/nonelec
      common/set21/ibctype
      common/set30/fprime(150,150),ipivot4(150),jpivot4(150)

      dimension xp(150), xpnew(150), b(150), f(150)

c     Set ``pert'' the value of the perturbation in x used
c     to evaluate the numerical derivative.

      pert = 0.0001

c   Create the jacobian matrix ``fprime'' in four steps:

c     1) Evaluate function of unperturbed set.
c     This may seem strange but we do this again in step 2 for
c     keeps; the only reason we do it here first is to fix the
c     value of the offset voltage in the gslow routine
c     ( if applicable ).  If this offset voltage is not
```

```
c     held fixed while we perturb the xp's, then the jacobian
c     will turn out singular.
c     ''keepoff'' is an argument of ''glsow'' which instructs
c     the gslow routine whether to KEEP the OFFset voltage
c     calculated during the last time through the routine,
c     or to calculate a new offset value for the present call.
      do 5 i = 1, nonelec
        xp(i) = x(i)
5       continue
      keepoff = 0
      if ( ibctype . eq . 0 )  call natural(xp,xpnew,keepoff)
      if ( ibctype . eq . 1 )  call essential(xp,xpnew)

c    2) Fill matrix with functional evaluations of perturbed x's.
      do 30 j = 1, nonelec
        do 10 i = 1, nonelec
          xp(i) = x(i)
          if ( i . eq . j ) xp(i) = xp(i) + pert
10        continue
        keepoff = 1
        if ( ibctype . eq . 0 )  call natural(xp,xpnew,keepoff)
        if ( ibctype . eq . 1 )  call essential(xp,xpnew)
        do 20 i = 1, nonelec
          fprime(i,j) = xpnew(i)
20        continue
30      continue

c    3) Evaluate function of unperturbed set.
      do 40 i = 1, nonelec
        xp(i) = x(i)
40      continue
      keepoff = 0
      if ( ibctype . eq . 0 )  call natural(xp,xpnew,keepoff)
      if ( ibctype . eq . 1 )  call essential(xp,xpnew)

c    4) Subtract vector ''xpnew'' from each column of the matrix.
c     Then divide each difference by pert.
c     Lastly, subtract unity from each diagonal element.
      do 70 j = 1, nonelec
        do 60 i = 1, nonelec
          fprime(i,j) = ( fprime(i,j) - xpnew(i) ) / pert
          if ( i . eq . j ) fprime(i,j) = fprime(i,j) - 1.0
60        continue
70      continue

c     Evaluate the vector ''f,''  ( f(x) = g(x) - x ).
c     (Recall that the last evaluation of g was for the
c     unperturbed vector x.)
c     Actually fill vector f with MINUS the value it should have
```

```
c       so that we can consider delx an additive correction.
        do 80 i = 1, nonelec
          f(i) = - ( xpnew(i) - x(i) )
80        continue

c       Solve the system ``FPRIME * delx = f'' using the solvers
c       ``gauss1'' and ``gauss2'' also used for the field problem.

        call gauss1(nonelec,fprime,ipivot4,jpivot4)
        call gauss2(nonelec,fprime,delx,f,ipivot4,jpivot4,0)

        return
        end



c       **************** SUBROUTINE ESSENTIAL ****************
c       This routine operates on a set of potential values
c       ``phi'' at the electrode and returns a new set
c       ``phinew'' by using phi as an essential boundary
c       condition in the field problem and then calculating
c       overpotentials (choice of 1) Butler-Volmer, 2) linear,
c       or 3) Tafel kinetics) corresponding to the
c       new current-density profile.

        subroutine essential(phi,phinew)
        implicit double precision (a-h,o-z)

        common/piv/ ipivot(150), jpivot(150)
        common/set0/nonelec
        common/set1/nonods
        common/set4/a(150,150)
        common/set5/c(150,150)
        common/set6/x(150)
        common/set7/bc(150)
        common/set12/scale
        common/set14/ikin
        common/set14a/xchcur,alphaa,alphac
        common/set14b/nopts,curap(40),curchar,etaap(40),vchar
        common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax

        dimension b(150)
        dimension phi(150), phinew(150)

c       Set bc(i).
        do 115 i=1,nonelec
          bc(i) = phi(i)
```

```
115     continue

c     Apply the BCs; make vector b.
      call makeb(nonods,c,bc,b)

c     Call routine ''gauss2'' to back-substitute and
c     get nodal derivatives.
      neumann = 0
      call gauss2(nonods,a,x,b,ipivot,jpivot,neumann)

c     Calculate the corresponding profile of electrode
c     potentials using the overpotential expression.
      do 130 i = 1, nonelec

c       Concentration overpotential: (we neglect it.)
        etac  =  0.

c       Activation overpotenital:

        cur  =  - x(i) * scale

c       Branch to the Kinetic Expression of choice:

c       1) Butler-Volmer kinetics
        if (ikin . eq . 1 ) then
          ratio  =  cur / xchcur
          etas   =  0.
          call bvkin(ratio,etas,alphaa,alphac)
          endif

c       2) Linear kinetics
        if ( ikin . eq . 2 ) then
          etas  =  cur / ( xchcur * ( alphaa + alphac ) )
          endif

c       3) Tafel kinetics:
        if ( ikin . eq . 3 ) then

c         This shouldn't happen in Tafel:
          if ( cur . eq . 0. ) then
            etas = 0.
            endif

c         Anode:
          if ( cur . gt . 0. ) then
            etas =   dlog(  cur/xchcur ) / alphaa
            endif

c         Cathode:
```

```
          if ( cur . lt . 0. ) then
            etas = - dlog( -cur/xchcur ) / alphac
            endif

          endif


c       Total overpotential:
        etat  =  etac +  etas

c       Calculate new surface values of phi ''phinew''.
        phinew(i) = - etat

130     continue

      return
      end



c     ****************** SUBROUTINE NATURAL ******************
      subroutine natural(grad,gradnew,keepoff)

c     This routine begins with an estimate for the phi derivative
c     ''grad'' at the electrode surface, and produces a new set of
c     derivatives ''gradnew'' by first solving
c     for phi at the polarized electrode and then using an
c     inverted kinetic expression (choice of Butler-Volmer, linear,
c     Tafel, or active-passive kinetics) to give current densities.

      implicit double precision (a-h,o-z)

      common/piv/ ipivot(150), jpivot(150)
      common/set0/nonelec
      common/set1/nonods
      common/set4/a(150,150)
      common/set5/c(150,150)
      common/set6/x(150)
      common/set7/bc(150)
      common/set8/keybc(150),keyode(150)
      common/set12/scale
      common/set14/ikin
      common/set14a/xchcur,alphaa,alphac
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax

      dimension b(150)
```

```
      dimension gradnew(150), grad(150)

c     Set bc(i).
      do 100 i = 1, nonelec
        bc(i)  =  grad(i)
100     continue

c     Apply the BCs.  Make vector b.
      call makeb(nonods,c,bc,b)

c     We enforce an essential condition at the
c     counterelectrode or elsewhere on the boundary.
      neumann = 0

c     Call routine ``gauss2'' to back-substitute and
c     get node values ``x'' for potential.
      call gauss2(nonods,a,x,b,ipivot,jpivot,neumann)

c     Vector ``x'' now contains electrode-surface values
c     of potential.  Use the electrode boundary condition
c     to calculate the new profile of gradients at the
c     electrode, ``gradnew.''

      do 110 i = 1, nonelec

c       Neglecting concentration overpotential, we have:
        etas  =   - x(i)

c       Branch to one of four kinetic expressions:
c         1) Butler-Volmer, 2) Linear,
c         3) Tafel, or 4) Active-Passive.

c       1) Butler-Volmer Kinetics:
        if ( ikin . eq . 1 ) then
          anod  =    dexp (   alphaa * etas )
          cath  =  - dexp ( - alphac * etas )
          cur   =  xchcur * ( anod + cath )
          endif

c       2) Linear Kinetics:
        if ( ikin . eq . 2 ) then
          cur  =  ( alphaa + alphac )  *  xchcur  *  etas
          endif

c       3) Tafel Kinetics
        if ( ikin . eq . 3 ) then

c         Shouldn't happen in Tafel:
          if ( etas . eq . 0. ) then
```

```
            cur  =  0.
            endif

c         Anode:
          if ( etas . ge . 0. ) then
            anod  =    dexp (   alphaa * etas )
            cur   =    xchcur  *  anod
            endif

c         Cathode:
          if ( etas . lt . 0. ) then
            cath  =    dexp ( - alphac * etas )
            cur   =  - xchcur  *  cath
            endif

          endif


c       4) Active-Passive Kinetics
        if ( ikin . eq . 4 ) then
          cur = polcurv(etas)
          endif

c       Express each current as a normal potential derivative and
c       pass it back into the calling routine as ''gradnew(i).''
        gradnew(i)  =  - cur / scale

110     continue

      return
      end




c     **************** SUBROUTINE MATRIX *******************
      subroutine matrix(nonods,coords,keyloc,A,C,keybc,keyode)

c     This routine does the following
c      1) Formulate matrix ''G''
c           ( in problem G q = H u ) stored in ''C''
c      2) Formulate matrix ''H'', stored in ''A''
c      3) Interchange columns to get coefficients of
c           unknowns all on the same side.

      implicit double precision (a-h,o-z)

      common/set3/iaxism,rinner
```

```
      common/set11/nogauss,wfgauss(12),abgauss(12)

      dimension coords(150,2), keyloc(150)
      dimension A(150,150), C(150,150)
      dimension keybc(150), keyode(150)

      external hgrand, ggrand


c     At each node j on the surface we calculate an integrated
c     local influence from each concentrated point charge i on
c     the boundary.

c     (For a given concentrated unit charge, each element on the
c     surface will feel an influence. Along each of these
c     elements, we carry out a surface integration
c     of the potential function and its normal derivative
c     and associate these values with the node(s)
c     located on the element. Thus we create matrices G(i,j)
c     and H(i,j) where i is the index of point sources and j
c     is the index of nodes at which the integrated local influence
c     is calculated.)
c     Matrix H is called A here, G is called C.
c     The BE equations are G q = H u.

      do 390 j=1,nonods

        key = keyloc(j)

c       Branch according to the local node number of the node j

        if ( key . eq . 0 ) go to 340

c       ********* Case of node j at the end of its element. *********

        n1 = j
        r1 = coords(n1,1) + rinner
        z1 = coords(n1,2)

c       Element ahead (clockwise) of node j:

        sign = 1.0

        n2 = j + 2
        if ( j . eq . nonods ) n2 = 2
        r2 = coords(n2,1) + rinner
        z2 = coords(n2,2)

        n3 = j + 1
```

```
      if ( j . eq . nonods ) n3 = 1
      r3 = coords(n3,1) + rinner
      z3 = coords(n3,2)

      do 310 i=1,nonods

        ri = coords(i,1) + rinner
        zi = coords(i,2)

        C(i,j) =  gquad(ggrand,nogauss,wfgauss,abgauss,
     u    key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3)
        A(i,j) =  sign * gquad(hgrand,nogauss,wfgauss,abgauss,
     u    key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3)

310     continue

c     Element behind (counterclockwise) node j

      sign =  -1.0

      n2 = j - 2
      if ( j . eq . 1 ) n2 = nonods - 1
      r2 = coords(n2,1) + rinner
      z2 = coords(n2,2)

      n3 = j - 1
      if ( j . eq . 1 ) n3 = nonods
      r3 = coords(n3,1) + rinner
      z3 = coords(n3,2)

      do 320 i = 1, nonods

        ri = coords(i,1) + rinner
        zi = coords(i,2)

        C(i,j) = C(i,j)  +
     u    gquad(ggrand,nogauss,wfgauss,abgauss,
     u    key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3)
        A(i,j) = A(i,j)  +
     u    sign * gquad(hgrand,nogauss,wfgauss,abgauss,
     u    key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3)

320     continue

c     End of end-node fork
      go to 390

c     *********** Case of node j as middle node. ************
```

```
340     n1 = j - 1
        r1 = coords(n1,1) + rinner
        z1 = coords(n1,2)

        n2 = j + 1
        r2 = coords(n2,1) + rinner
        z2 = coords(n2,2)

        n3 = j
        r3 = coords(n3,1) + rinner
        z3 = coords(n3,2)

        do 350 i = 1, nonods

          ri = coords(i,1) + rinner
          zi = coords(i,2)

          C(i,j) =  gquad(ggrand,nogauss,wfgauss,abgauss,
     u      key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3)
          A(i,j) =  gquad(hgrand,nogauss,wfgauss,abgauss,
     u      key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3)

350       continue

390     continue

c     Obtain the diagonal elements of H by satisfying the no-
c     flux-at-uniform-potential condition. (i.e. sum of H(i,j)
c     on a given row is zero)

      do 399 i = 1, nonods

        A(i,i) = 0.
        sum = 0.
        do 395 j=1,nonods
          sum = sum + A(i,j)
395       continue
        A(i,i) = - sum

399     continue

c     If you need to, save matrices G and H for later use in
c     evaluating internal potentials and currents.
c      do 420 i=1,nonods
c        do 415 j=1,nonods
c          G(i,j) = C(i,j)
c          H(i,j) = A(i,j)
c415       continue
c420     continue
```

```
c       Rearrange matrices G and H so that H contains only the
c       coefficients of unknowns.  This will be returned as ''A.''
        do 440 j=1,nonods
          key = keybc(j)
          if (key.eq.0) go to 440
          do 435 i=1,nonods
            hold    =   A(i,j)
            A(i,j)  =  -C(i,j)
            C(i,j)  =  -hold
435         continue
440       continue

        return
        end



c       ******************** SUBROUTINE MAKEB ********************
        subroutine makeb(nonods,C,bc,b)

        implicit double precision (a-h,o-z)

        dimension C(150,150), bc(150), b(150)

c       This routine does the tiny task of multiplying the vector
c       of boundary conditions ''bc'' by the constant coefficient matrix
c       ''C'' (formed by moving all coefficients of knowns to one side
c       of the BE equations) to form the vector ''b'' (which will enter
c       the matrix problem A x = b, solved later by subroutines
c       ''gauss1'' and ''gauss2.''

        do 610 i = 1, nonods
          b(i) = 0.
          do 605 j = 1, nonods
            b(i) = b(i) + C(i,j) * bc(j)
605         continue
610       continue

        return
        end



c       ***************** SUBROUTINE GAUSS1 ****************
        subroutine gauss1(n,a,ipivot,jpivot)

c       This routine does the first stage of Gauss Elimination
```

```
c     on the matrix a.  Subroutine ''gauss2'' will follow and
c     do backsubstitution with the vector b to give the vector
c     of unknowns x.

      implicit double precision (a-h,o-z)

      dimension a(150,150)
      dimension  ipivot(150), jpivot(150)

c     One at a time, we will select n-1 rows (index k)
c     and reduce elements of the matrix by subtracting multiples
c     of row k.  We do not take the rows in order, however,
c     but instead select each 'key row' by a criterion and then
c     use full pivoting to  1) move the chosen row into the key
c     position (i=k) and  2) move the largest element in the key
c     row into the 'key column' (j=k).  The criterion for selecting
c     the key row is that the ratio of its largest element to its
c     second-largest element be the greatest among all the rows that
c     have not been already chosen.  A record of the row exchanges
c     is made in vector ''ipivot,'' a record of column exchanges in
c     vector ''jpivot,'' and a record of row operations in the
c     elements of a that have been zeroed.

c     Initialize the bookkeeping vectors
      do 505 i = 1, n
        ipivot(i) = i
        jpivot(i) = i
505     continue

c     For each of n-1 rows do the following:
      nm1 = n - 1
      do 590 k = 1, nm1

c       Search the unreduced portion of the matrix.  Find
c       the largest element in the row having the largest
c       ratio of biggest to second-biggest elements
c       (in absolute value).  Keep track of locations.

        quomax = 0.
c       Search each row.
        do 530 i = k, n

          big1 = 0.
          big2 = 0.
          j1 = k
c         Look at each element on row i.
          do 520 j = k, n

            if ( dabs( a(i,j) ) . ge . big1 ) go to 510
```

```
            if ( dabs( a(i,j) ) . lt . big2 ) go to 520
            big2 = dabs( a(i,j) )
            go to 520

510         big2 = big1
            big1 = dabs( a(i,j) )
            j1 = j

c           End search on a given row.
520         continue

          quo12  =   big1 / ( big2  + 1.0e-20 )
          if ( quo12 . lt . quomax ) go to 530
          quomax = quo12
          ipiv = i
          jpiv = j1

c         End search among rows.
530       continue

c       We are finished searching.
c       Record the location of the pivot element for later
c       use in the back substitution routine, ''gauss2.''
        ipivot(k) = ipiv
        jpivot(k) = jpiv

c       Perform the pivot.

c       Row pivot.  Do not move the history elements.
        do 540 j = k, n
          hold       =  a(ipiv,j)
          a(ipiv,j)  =  a(k,j)
          a(k,j)     =  hold
540       continue

c       Column pivot:
        do 550 i = 1, n
          hold       =  a(i,jpiv)
          a(i,jpiv)  =  a(i,k)
          a(i,k)     =  hold
550       continue

c       If the diagonal element of the new key row is nearly zero.
c       quit here and declare the matrix singular.

        diag = a(k,k)
        zero = 1.0e-6
        if ( dabs(diag) . le . zero ) then
          write (2,555)
```

```
555       format('The matrix is singular, old sport.')
          stop
          endif

c       Divide the key row by its diagonal element, except for
c       the diagonal element itself which is retained as is.
        kp1 = k + 1
        do 570 j = kp1, n
          a(k,j) = a(k,j) / diag
570       continue

c       Proceed with forward reduction.

        do 580 i = kp1, n

c         Reduction:
          do 575 j = kp1, n
            a(i,j) = a(i,j) - a(i,k) * a(k,j)
575         continue

c         Store the history of fwd reduction where zeros belong.
          a(i,k) = a(i,k) / diag

580       continue

590     continue

      return
      end



c     ****************** SUBROUTINE GAUSS2 ******************
      subroutine gauss2(n,a,x,b,ipivot,jpivot,neumann)

c     This is the companion routine to ''gauss1.'' Together, they
c     solve the matrix problem A x = b. They are separated into
c     two routines so that the forward reduction (gauss1) can be
c     carried out only once for multiple executions of the back
c     substitution. ''Gauss2'' picks up after the matrix has been
c     reduced to upper triangular form with the lower elements
c     carrying a record of the operations which must be done on the
c     vector b before doing the back substitution.
c     Also, vectors ipivot and jpivot contain records of the row
c     and column exchanges that were carried out in ''gauss1.''
c     Lastly, if we are dealing with a Neumann problem (neumann=1),
c     we prevent the arbitrary additive constant in the solution
c     from adversely influencing precision by setting x(n) = 0.0.
```

```
      implicit double precision (a-h,o-z)

      dimension a(150,150), b(150), x(150)
      dimension ipivot(150), jpivot(150)

c     Do the same series of manipulations on vector b that you did
c     on matrix A in gauss1.

      kmax = n - 1
      do 730 k = 1, kmax

        kp1 = k + 1
        ix = ipivot(k)

c       If no row exchange is called for, skip the exchange.
        if (ix.eq.k) go to 710

c       Perform the row exchange.
        hold  =  b(k)
        b(k)  =  b(ix)
        b(ix) =  hold

c       Do k+1st row reduction on vector ``b.''
710     do 720 i = kp1, n
          b(i) = b(i) - a(i,k) * b(k)
720       continue

c       Divide the ``key'' element of ``b'' by the corresponding
c       diagonal element of the matrix ``A.''
        b(k) = b(k) / a(k,k)

730     continue

c       Vector b now contains just what it would have if it had
c       been carried along with the forward reduction of matrix A.

c       Now do back substitution.  Vector x gets progressively
c       filled with the answer from bottom to top.

c       Recall that b(n) is the only element of vector
c       ``b'' which was not divided by the old diagonal
c       element of matrix a.  (If we have a neumann problem,
c       we'll set x(n) in the next section.)
        if ( neumann . ne . 1 )  x(n) = b(n) / a(n,n)

c       If this is a Neumann problem, we can only solve
c       for the field variable within an additive constant.
c       In this case we set x(n) to zero ( so as to retain
c       precision by keeping the additive constant from
```

```
c       turning out to be 1.e19 for example ).
        if ( neumann . eq . 1 ) x(n) = 0.

c       Remember that the diagonals of the matrix A are really
c       all ones, but we are storing different values there.
          do 750 kcompl = 2, n
          k = n + 1 - kcompl
          sum = b(k)
          kp1 = k + 1

          do 740 j = kp1, n
            sum = sum - a(k,j) * x(j)
740         continue

          x(k) = + sum
750       continue

c       Recalling that column pivoting was done during the forward
c       reduction in ''gauss2,'' we must now unscramble our answer
c       vector.  Procede in reverse of the order in which the
c       column exchanges were originally made.

        do 770 kcompl = 2, n
          k = n + 1 - kcompl
          m = jpivot(k)
          hold = x(m)
          x(m) = x(k)
          x(k) = hold
770       continue

        return
        end



c     ************** SUBROUTINE CURINT *****************
      subroutine curint(curtot)

      implicit double precision (a-h,o-z)

c     This routine integrates current density over the working
c     electrode surface (axisymmetric problems) to give
c     the total cell current ''curtot''.
      common/set0/nonelec
      common/set1/nonods
      common/set2/coords(150,2)
      common/set2a/keyloc(150)
      common/set3/iaxism,rinner
```

```
      common/set6/x(150)
      common/set7/bc(150)
      common/set8/keybc(150),keyode(150)
      common/set10a/tol,fracfp,fracnr
      common/set10b/mctr,mctrmax,mswitch,mconsec
      common/set10c/nctr,nctrmax,ndivmax
      common/set10d/ncon,nconmax
      common/set10e/ntot,ntotmax
      common/set12/scale
      common/set14/ikin
      common/set14a/xchcur,alphaa,alphac
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax

      dimension arcl(150)

c     Calculate the arc-length addresses of nodes along the
c     working-electrode surface.  Store these in vector
c     ''arcl''.  Also calculate the total current to the
c     anode surface, ''curtot.''

      narc   = 200
      narco2 = narc / 2
      noeelec = ( nonelec - 1 ) / 2
      arc    =  0.
      curtot =  0.
      do 20 i = 1, noeelec

        n1  =  2 * i  -  1
        n2  =  2 * i  +  1
        n3  =  2 * i

        r1  =  coords(n1,1)  +  rinner
        r2  =  coords(n2,1)  +  rinner
        r3  =  coords(n3,1)  +  rinner

        z1  =  coords(n1,2)
        z2  =  coords(n2,2)
        z3  =  coords(n3,2)

        cd1  =  - bc(n1)  *  curchar * scale
        cd2  =  - bc(n2)  *  curchar * scale
        cd3  =  - bc(n3)  *  curchar * scale

        ds =  2.0 / float(narc)

c       Divide each element up into ''narc'' segments and take the
c       arc length by adding up short tangents.
        do 15 k = 1, narc
```

```
        if ( k . eq . 1 )   arcl(n1) = arc

        s  =  - 1.0  +  2.0 * float( k - 1 ) / float(narc)

        p1 =  - 0.5  *  s  * ( 1.0 - s )
        p2 =    0.5  *  s  * ( 1.0 + s )
        p3 =   ( 1.0 - s ) * ( 1.0 + s )

        r    =  p1 * r1   +  p2 * r2   +  p3 * r3
        cdd  =  p1 * cd1  +  p2 * cd2  +  p3 * cd3

        dp1ds  =  s  -  0.5
        dp2ds  =  s  +  0.5
        dp3ds  =  - 2.0 * s

        drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
        dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

        dr  =  drds * ds
        dz  =  dzds * ds

        darc  =  dsqrt ( dr ** 2  +  dz ** 2 )
        arc  =  arc  +  darc

        pi = 3.14159265358979323846
        dcurtot = darc * 2.0 * pi * r * cdd
        curtot = curtot + dcurtot

        if ( k . eq . narco2 ) arcl(n3) = arc
        if ( k . eq . narc   ) arcl(n2) = arc

15      continue

20    continue

      return
      end



c     ************** SUBROUTINE PLOTFILE ******************
      subroutine plotfile

      implicit double precision (a-h,o-z)

c     This routine prints out the answer.
      common/set0/nonelec
```

```
      common/set1/nonods
      common/set2/coords(150,2)
      common/set2a/keyloc(150)
      common/set3/iaxism,rinner
      common/set6/x(150)
      common/set7/bc(150)
      common/set8/keybc(150),keyode(150)
      common/set10a/tol,fracfp,fracnr
      common/set10b/mctr,mctrmax,mswitch,mconsec
      common/set10c/nctr,nctrmax,ndivmax
      common/set10d/ncon,nconmax
      common/set10e/ntot,ntotmax
      common/set12/scale
      common/set14/ikin
      common/set14a/xchcur,alphaa,alphac
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax

      dimension arcl(150)

c     Calculate the arc-length addresses of nodes along the
c     working-electrode surface.  Store these in vector
c     ''arcl''.  Also calculate the total current to the
c     anode surface, ''curtot.''

      narc   = 200
      narco2 = narc / 2
      noeelec = ( nonelec - 1 ) / 2
      arc    =  0.
      curtot =  0.
      do 20 i = 1, noeelec

        n1  =  2 * i  -  1
        n2  =  2 * i  +  1
        n3  =  2 * i

        r1  =  coords(n1,1)  +  rinner
        r2  =  coords(n2,1)  +  rinner
        r3  =  coords(n3,1)  +  rinner

        z1  =  coords(n1,2)
        z2  =  coords(n2,2)
        z3  =  coords(n3,2)

        cd1  =  - bc(n1)  *  curchar * scale
        cd2  =  - bc(n2)  *  curchar * scale
        cd3  =  - bc(n3)  *  curchar * scale

        ds =  2.0 / float(narc)
```

```
c       Divide each element up into ''narc'' segments and take the
c       arc length by adding up short tangents.
        do 15 k = 1, narc

          if ( k . eq . 1 )   arcl(n1) = arc

          s  =  - 1.0  +  2.0 * float( k - 1 ) / float(narc)

          p1 =  - 0.5  *  s  * ( 1.0 - s )
          p2 =    0.5  *  s  * ( 1.0 + s )
          p3 =   ( 1.0 - s ) * ( 1.0 + s )

          r   =  p1 * r1   +  p2 * r2   +  p3 * r3
          cdd =  p1 * cd1  +  p2 * cd2  +  p3 * cd3

          dp1ds  =  s  -  0.5
          dp2ds  =  s  +  0.5
          dp3ds  =  - 2.0 * s

          drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
          dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

          dr  =  drds * ds
          dz  =  dzds * ds

          darc  =  dsqrt ( dr ** 2  +  dz ** 2 )
          arc  =  arc  +  darc

          pi = 3.14159265358979323846
          dcurtot = darc * 2.0 * pi * r * cdd
          curtot = curtot + dcurtot

          if ( k . eq . narco2 ) arcl(n3) = arc
          if ( k . eq . narc   ) arcl(n2) = arc

15        continue

20      continue


c     Create a special file (for004) for plotting idist:

        write (4,61)
61      format(/'title1: descriptive information'/)
        write (4,62)
62      format(/'title2:'/)
        write (4,63)
63      format(/'title3:'/)
        write (4,64)
```

```
64    format(/'title4:'/)
      write (4,65)
65    format(/'title5:'/)
      write (4,66)
66    format(/'title6:'/)
      write (4,67)
67    format(/'title7:'/)
      write (4,68)
68    format(/'title8:'/)

c     Pass these default values if only to mark where the number goes:
      radmax = 240.0
      curmax  = 50.0
      curmin  = 0.0
      curpeak = 38.35
      ilog = 0
      write (4,69) radmax, ilog, curmax, curmin, curpeak
69    format(/'radmax',2x,'ilog',5x,'curmax',8x,'curmin',
     $   8x,'curpeak'/1x,f6.2,7x,i1,5x,e8.2,5x,e8.2,5x,e9.3)

      noaxes = 0
      ifont = 3
      write (4,70) nonelec, scale, noaxes, blt, edt, ifont
70    format(/'nonelec',7x,'scale',7x,'noaxes',
     $   7x,'blt',7x,'edt',6x,'ifont'/2x,i4,7x,f8.2,7x,i1,7x,
     $   f7.3,7x,f7.3,5x,i2)

      write (4,80)
80    format(/'node',2x,'bc',2x,'phi,mV',6x,'i, pol',3x,
     $    'i,mA/cm2',3x,'arclength',3x,'r or y',2x,'x or z')

c     Convert potentials, ``pt,'' into millivolts; current densities,
c     ``cd,'' into milliamps per square centimeter:
      do 100 i = 1, nonods
        if ( keybc(i) . eq . 0 ) then
          pt  =    x(i)   *  vchar
          cd  =  - bc(i)  *  curchar * scale
         else
          pt  =    bc(i)  *  vchar
          cd  =  - x(i)   *  curchar * scale
         endif

c       In the case of active-passive kinetics,
c       write out the current calculated by the
c       polarization curve ``polcurv,'' ``c'' for comparison.
        c = 0.0
        if ( ikin . eq . 4 ) then
          if ( keyode(i) . eq . 1 ) then
            overpot =  - pt / vchar
```

```
            c  =  curchar * polcurv(overpot)
            endif
          endif

        write (2,110) i, keybc(i), pt, c, cd,
     $      arcl(i), coords(i,1), coords(i,2)
        write (4,110) i, keybc(i), pt, c, cd,
     $      arcl(i), coords(i,1), coords(i,2)

100     continue

110   format(i4,1x,i2,1x,g10.3,1x,g10.3,1x,g10.3,1x,
     $   f10.3,1x,f8.1,1x,f8.1)

      write (4,115) curtot
115   format(/'curtot, mA if 1 unit = 1 cm'/g10.4//)

      return
      end



c     ************** SUBROUTINE OUTPUT *****************
      subroutine output

      implicit double precision (a-h,o-z)

c     This routine prints out the answer.
      common/set0/nonelec
      common/set1/nonods
      common/set2/coords(150,2)
      common/set2a/keyloc(150)
      common/set3/iaxism,rinner
      common/set6/x(150)
      common/set7/bc(150)
      common/set8/keybc(150),keyode(150)
      common/set10a/tol,fracfp,fracnr
      common/set10b/mctr,mctrmax,mswitch,mconsec
      common/set10c/nctr,nctrmax,ndivmax
      common/set10d/ncon,nconmax
      common/set10e/ntot,ntotmax
      common/set12/scale
      common/set14/ikin
      common/set14a/xchcur,alphaa,alphac
      common/set14b/nopts,curap(40),curchar,etaap(40),vchar
      common/set15/delv,delv0,nodelvf,delvf(40),deldelv,delpmax
```

```
      dimension arcl(150)

c     Calculate the arc-length addresses of nodes along the
c     working-electrode surface.  Store these in vector
c     ''arcl''.  Also calculate the total current to the
c     anode surface, ''curtot.''

      narc   = 200
      narco2 = narc / 2
      noeelec = ( nonelec - 1 ) / 2
      arc    =  0.
      curtot =  0.
      do 20 i = 1, noeelec

        n1  =  2 * i  -  1
        n2  =  2 * i  +  1
        n3  =  2 * i

        r1  =  coords(n1,1)  +  rinner
        r2  =  coords(n2,1)  +  rinner
        r3  =  coords(n3,1)  +  rinner

        z1  =  coords(n1,2)
        z2  =  coords(n2,2)
        z3  =  coords(n3,2)

        cd1  =  - bc(n1)  *  curchar * scale
        cd2  =  - bc(n2)  *  curchar * scale
        cd3  =  - bc(n3)  *  curchar * scale

        ds =  2.0 / float(narc)

c       Divide each element up into ''narc'' segments and take the
c       arc length by adding up short tangents.
        do 15 k = 1, narc

          if ( k . eq . 1 )   arcl(n1) = arc

          s  =  - 1.0  +  2.0 * float( k - 1 ) / float(narc)

          p1 =  - 0.5  *  s  * ( 1.0 - s )
          p2 =    0.5  *  s  * ( 1.0 + s )
          p3 =   ( 1.0 - s ) * ( 1.0 + s )

          r    =  p1 * r1   +  p2 * r2   +  p3 * r3
          cdd  =  p1 * cd1  +  p2 * cd2  +  p3 * cd3

          dp1ds  =  s  -  0.5
          dp2ds  =  s  +  0.5
```

```
          dp3ds  =  - 2.0 * s

          drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
          dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

          dr  =  drds * ds
          dz  =  dzds * ds

          darc  =  dsqrt ( dr ** 2  +  dz ** 2 )
          arc  =  arc  +  darc

          pi = 3.14159265358979323846
          dcurtot = darc * 2.0 * pi * r * cdd
          curtot = curtot + dcurtot

          if ( k . eq . narco2 ) arcl(n3) = arc
          if ( k . eq . narc   ) arcl(n2) = arc

15        continue

20      continue

      write (2,25)
25    format(/'heres the vector full of answers')

      write (2,30)
30    format(/'node',2x,'bc',2x,'phi,mV',6x,'i, pol',3x,
     $     'i,mA/cm2',3x,'arclength',3x,'r or y',2x,'x or z')

c     Convert potentials, ``pt,'' into millivolts; current densities,
c     ``cd,'' into milliamps per square centimeter:
      do 100 i = 1, nonods
        if ( keybc(i) . eq . 0 ) then
          pt  =    x(i)   *  vchar
          cd  =  - bc(i)  *  curchar * scale
         else
          pt  =    bc(i)  *  vchar
          cd  =  - x(i)   *  curchar * scale
         endif

c       In the case of active-passive kinetics,
c       write out the current calculated by the
c       polarization curve ``polcurv,'' ``c'' for comparison.
        c = 0.0
        if ( ikin . eq . 4 ) then
          if ( keyode(i) . eq . 1 ) then
            overpot =  - pt / vchar
            c  =  curchar * polcurv(overpot)
            endif
```

```
          endif

        write (2,110) i, keybc(i), pt, c, cd,
     $      arcl(i), coords(i,1), coords(i,2)

100     continue

110   format(i4,1x,i2,1x,g10.3,1x,g10.3,1x,g10.3,1x,
     $   f10.3,1x,f8.1,1x,f8.1)

      write (2,120) nctr
120   format(/i5,1x,'iterations')

      return
      end



c     ************* FUNCTION GQUAD *************
c     This function integrates the function ``prom'' by gaussian
c     quadrature with ``nogauss'' points.
      function gquad(prom,nogauss,wfgauss,abgauss,key,iaxism,
     u   ri,zi,r1,z1,r2,z2,r3,z3)

        implicit double precision (a-h,o-z)
        dimension wfgauss(12), abgauss(12)

        sum = 0.
        do 810 igauss = 1, nogauss
          e    =  abgauss(igauss)
          wf   =  wfgauss(igauss)
          eval =  prom(key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3,e)
          sum  =  sum  +  eval * wf
810       continue
        gquad  =  sum
        return
        end



c     ************* FUNCTION GGRAND ****************
c     This function calculates a contribution to an
c     element of the matrix G.  It works for both the on- and
c     off-diagonal elements, differing only by a correction term
c     for the singularity.
```

```
      function ggrand(key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3,e)

      implicit double precision (a-h,o-z)
      double precision mmdelk

      ggrand = 0.
      small = 1.0e-5

c     If the next node is negligibly far away, return a zero
c     contribution to the matrix.
      dlength = dsqrt ( (r1-r3)*(r1-r3) + (z1-z3)*(z1-z3) )
      if ( dlength . le . small ) go to 890

      p1 =  - 0.5  *  e  * ( 1.0 - e )
      p2 =    0.5  *  e  * ( 1.0 + e )
      p3 =   ( 1.0 - e ) * ( 1.0 + e )

      r  =  p1 * r1  +  p2 * r2  +  p3 * r3
      z  =  p1 * z1  +  p2 * z2  +  p3 * z3

      drde = (e - 0.5) * r1  +  (e + 0.5) * r2  -  2.0 * e * r3
      dzde = (e - 0.5) * z1  +  (e + 0.5) * z2  -  2.0 * e * z3
      abjacob = dsqrt ( drde * drde  +  dzde * dzde )

c     Set the weighting function for integration along the element
c     ( basis function in quadratic interpolation ) according to
c     whether node j is an end node or a middle node on its element.
      if ( key . eq . 1 ) basis = p1
      if ( key . eq . 0 ) basis = p3

      pi = 3.14159265358979323846

c     Branch according to whether the problem is axisymmetric
c     (iaxism=1) or planar (iaxism=0)
      if ( iaxism . eq . 0 ) go to 840

c     ************** Case of Axisymmetric Problem **************

c     If r1 and r3 are both zero, than we are looking at
c     the centerline and we should make no contribution to
c     the system matrix.
      if ( dabs(r1) . le . small . and .
     $     dabs(r3) . le . small ) go to 890

      x = ( (r-ri)*(r-ri) + (z-zi)*(z-zi) ) /
     $    ( (r+ri)*(r+ri) + (z-zi)*(z-zi) )

c     Here we need the eliptic integral of x (m1 in abromowitz) of the
c     first kind.  Use the IMSL routine to get it to 16 digits.
```

```
      temp = mmdelk ( 3, x, ierk1 )
      temp = temp / dsqrt( (r+ri)*(r+ri) + (z-zi)*(z-zi) )
      ustarr = temp * r / pi

      ggrand = ustarr * basis * abjacob

      go to 870
c     End of axisymmetric branch


c     ****************** Case of Planar Problem ******************

840   dist = dsqrt ( (r-ri)*(r-ri) + (z-zi)*(z-zi) )
      ustar = - dlog ( dist ) / ( 2.0 * pi )
      ggrand = ustar * basis * abjacob

c     End of planar branch


c     ***************** Correction for Singularity ***************
c     See if i and j are the same point (diagonal element
c     or components of a dual node.) If so, add on a correction
c     which accounts for the singularity.
870   if ( key . eq . 1 ) rj = r1
      if ( key . eq . 1 ) zj = z1
      if ( key . eq . 0 ) rj = r3
      if ( key . eq . 0 ) zj = z3
      d = dsqrt ( (ri-rj)*(ri-rj) + (zi-zj)*(zi-zj) )
      if ( d . ge . small ) go to 890

c     The correction term for the singularity is the
c     same for axisymmetric and planar problems.

c     Branch according to the position of node j on the element.

      if ( key . eq . 0 ) go to 880

c     Case of end node:
        ubjacob = dsqrt (
     u     ( 1.5 * ( r3 - r1 ) - 0.5 * ( r2 - r3 ) ) *
     u     ( 1.5 * ( r3 - r1 ) - 0.5 * ( r2 - r3 ) ) +
     u     ( 1.5 * ( z3 - z1 ) - 0.5 * ( z2 - z3 ) ) *
     u     ( 1.5 * ( z3 - z1 ) - 0.5 * ( z2 - z3 ) ) )
        two = 2.
        cor = ( dlog(1. + e) + 1. - dlog(two) ) * ubjacob / (2. * pi)
        ggrand = ggrand + cor
        go to 890

c     Case of internal node:
```

```
880     ubjacob = 0.5 * dsqrt (
     u     ( r2 - r1 ) * ( r2 - r1 )  +
     u     ( z2 - z1 ) * ( z2 - z1 )  )
        cor = ( 1. + dlog( dabs(e) ) ) * ubjacob / ( 2.0 * pi )
        ggrand  =  ggrand  +  cor

890   return
      end



c     *************** FUNCTION HGRAND ****************
c     This function evaluates a contribution to each
c     element of matrix H.  We never use it for diagonal elements.

      function hgrand(key,iaxism,ri,zi,r1,z1,r2,z2,r3,z3,e)
      implicit double precision (a-h,o-z)
      double precision mmdelk, mmdele

      hgrand = 0.
      small = 1.0e-5

c     If the next node is negligibly far away, return a zero
c     contribution to the matrix.
      dlength = dsqrt ( (r1-r3)*(r1-r3) + (z1-z3)*(z1-z3) )
      if ( dlength . le . small ) go to 990

c     If nodes i and j have the same coordinates ( either i = j
c     or i and j make a dual node ) then the contribution to
c     the matrix will involve a singularity which we do not
c     handle here.  These contributions are taken care of in the
c     ''matrix'' subroutine where we satisfy the ''no-flux-at-uniform-
c     potential condition.''
      if ( key . eq . 1 ) rj = r1
      if ( key . eq . 1 ) zj = z1
      if ( key . eq . 0 ) rj = r3
      if ( key . eq . 0 ) zj = z3
      d = dsqrt ( (ri-rj)*(ri-rj) + (zi-zj)*(zi-zj) )
      if ( d . le . small ) go to 990

      p1 =  - 0.5  *  e  * ( 1.0 - e )
      p2 =    0.5  *  e  * ( 1.0 + e )
      p3 =   ( 1.0 - e ) * ( 1.0 + e )

      r  =  p1 * r1  +  p2 * r2  +  p3 * r3
      z  =  p1 * z1  +  p2 * z2  +  p3 * z3

      drde = (e - 0.5) * r1  +  (e + 0.5) * r2  -  2.0 * e * r3
```

```
      dzde = (e - 0.5) * z1  +  (e + 0.5) * z2  -  2.0 * e * z3

      abjacob = dsqrt ( drde * drde  +  dzde * dzde )

      dircosr =  - dzde / abjacob
      dircosz =    drde / abjacob

c     Set the weighting function for integration along the element
c     ( basis function in quadratic interpolation ) according to
c     whether node j is an end node or a middle node on its element.
      if ( key . eq . 1 ) basis = p1
      if ( key . eq . 0 ) basis = p3

      pi = 3.14159265358979323846

c     Branch according to whether the problem is axisymmetric
c     (iaxism=1) or planar (iaxism=0)
      if ( iaxism . eq . 0 ) go to 940

c     ************** Case of Axisymmetric Problem **************

c     If r1 and r3 both equal zero, then we are looking at the
c     centerline and we should make no contribution to the
c     system matrix.

      if ( dabs(r1) . le . small . and .
     u     dabs(r3) . le . small ) go to 990

      x = ( (r-ri)*(r-ri) + (z-zi)*(z-zi) ) /
     u    ( (r+ri)*(r+ri) + (z-zi)*(z-zi) )

      f1 = 1.0 / pi
      f1 = f1 / dsqrt ( (r+ri)*(r+ri) + (z-zi)*(z-zi) )
      t1 = mmdelk(3,x,ierk2) / (2.0 * r)
      t2 = mmdele(3,x,iere1) * ( r*r - ri*ri - (z-zi)*(z-zi) )
      t2 = t2 / ( 2.0 * r * ( (r-ri)*(r-ri) + (z-zi)*(z-zi) ) )
      t3 = mmdele(3,x,iere2)*(z-zi) / ((r-ri)*(r-ri)+(z-zi)*(z-zi))

      qstarr = - f1 * r * ( ( t1 + t2 ) * dircosr  +  t3 * dircosz )

      hgrand = qstarr * basis * abjacob

      go to 990
c     End of axisymmetric branch


c     ****************** Case of Planar Problem ******************
940   dist = dsqrt ( (r-ri)*(r-ri) + (z-zi)*(z-zi) )
      temp = ( (r-ri) * dircosr + (z-zi) * dircosz )
```

```
      qstar = - temp / ( 2.0 * pi * dist * dist )
      hgrand = qstar * basis * abjacob

c     End of planar branch

 990  return
      end


c     ************* FUNCTION EI1 ***************
c     This function approximates the complete eliptic integral of
c     the first kind of the complementary parameter x (called m1
c     by Abromowitz and Stegun.)
c     This is not used in this program unless the IMSL library
c     is unavailable

      function ei1(x)
        implicit double precision (a-h,o-z)

        p = ( ( ( (
     u  0.01451196212 ) * x  +
     u  0.03742563713 ) * x  +
     u  0.03590092383 ) * x  +
     u  0.09666344259 ) * x  +
     u  1.38629436112

        q = ( ( ( (
     u  0.00441787012 ) * x  +
     u  0.03328355346 ) * x  +
     u  0.06880248576 ) * x  +
     u  0.12498593597 ) * x  +
     u  0.50000000000

      ei1 = p - q * dlog(x)

      return
      end


c     ************* FUNCTION EI2 **************
c     This function estimates the value of the complete elliptic
c     integral of the second kind with complimentary parameter x.
c     This is not used in this program unless the IMSL library
c     is unavailable

      function ei2(x)
```

```
      implicit double precision (a-h,o-z)

      p = ( ( ( (
     u  0.01736506451 ) * x  +
     u  0.04757383546 ) * x  +
     u  0.06260601220 ) * x  +
     u  0.44325141463 ) * x  +
     u  1.00000000000

      q = ( ( ( (
     u  0.00526449639 ) * x  +
     u  0.04069697526 ) * x  +
     u  0.09200180037 ) * x  +
     u  0.24998368310 ) * x  +
     u  0.

      ei2 = p - q * dlog(x)

      return
      end



c     ***************** FUNCTION POLCURV *****************

c     This routine does a trapezoidal interpolation of
c     current-density/overpotential pairs (cur/eta) to
c     return a value of current ''polcurv'' corresponding
c     to the supplied value of dimensionless overpotential ''etaval.''
c     A set of ''nopts'' current-density/overpotential pairs
c     are passed in vectors ''curap'' and ''etaap.''
c     Pay attention to units:  ''etaval'' is a dimensionless
c     overpotential and ''polcurv'' is a dimensionless current
c     density.  However, curap and etaap are in mA/cm2 and
c     mV, so we temporarily convert to these real units inside
c     the function, and convert back before returning.

      function polcurv(eta)

        implicit double precision (a-h,o-z)

        common/set14b/nopts,curap(40),curchar,etaap(40),vchar

c       Convert ''eta'' to millivolts:
        etamv = eta * vchar

c       Corrective treatment of case (unrealistic) of subzero etaval:
c       (Just a convergence trick)
```

```
      if ( eta . le . 0. ) then
        polcurv  = - eta * 0.001
        return
        endif

c     Trapezoidal interpolation of current density:
      do 20 j = 2, nopts
        if ( etamv . le . etaap(j) ) then
          curma  = curap(j-1) + ( curap(j) - curap(j-1) ) *
     $      ( etamv - etaap(j-1) ) / ( etaap(j) - etaap(j-1) )

c         Nondimensionalize the current density before returning:
          polcurv  =  curma / curchar

          return
          endif
20      continue

      return
      end



c     ********************** SUBROUTINE BVKIN ************************
      subroutine bvkin(ratio,etas,alphaa,alphac)

      implicit double precision (a-h,o-z)

c     This routine solves for surface overpotential ''etas'' in the
c     Butler-Volmer kinetic expression.  ''ratio'' is the ratio of
c     the current density to the exchange-current density.
c     ''alphaa'' and ''alphac'' are the transfer coeficients.
c     ''etas'' has been made dimensionless by ''vchar.''

c     The Newton-Raphson algorithm:

c     Relative convergence tolerance:
      tol = 1.0e-15

c     Maximum number of iterations:
      nctrmax = 100

c     Initialize:
      nctr = 0

c     Initializations for the Tafel extremes:
      if (ratio.ge.1.0e+4)   etas =  dlog( ratio) / alphaa
      if (ratio.le.-1.0e+4)  etas = -dlog(-ratio) / alphac
```

```
10    nctr = nctr + 1

c     Function ''f'' will go to zero at convergence.
      anod = dexp (   alphaa * etas )
      cath = dexp ( - alphac * etas )
      f  =  anod  -  cath  -  ratio

c     We differentiate analytically:
      fprime  =  alphaa * anod  +  alphac * cath

c     Newton-Raphson increment:
      deletas  =  -  f  /  fprime

c     Don't let etas change too much between iterations:
      dmax =   1.0
      dmin = - 1.0
      if ( deletas . gt . dmax ) deletas = dmax
      if ( deletas . lt . dmin ) deletas = dmin

c     Newton-Raphson correction:
      etas  =  etas  +  deletas

c     Convergence:
      if ( dabs(deletas) . lt . tol ) then
        return
        endif

c     Divergence: (Too many iterations)
      if ( nctr . gt . nctrmax ) then
        write (2,20) etas
20      format ('too many iterations on bvkin.'/'etas='g8.2)
        stop
        endif

c     Keep iterating:
      go to 10

      end



c     ******************* SUBROUTINE RSREAD *********************
c     If this run is a restart ( irestrt = 1 ) we skip the costly
c     ''matrix'' and ''gauss1'' routines and merely read in
c     their products from the restart file, for003.dat.

      subroutine rsread
```

```
      implicit double precision (a-h,o-z)

      common/set1/nonods
      common/piv/ ipivot(150), jpivot(150)
      common/set4/a(150,150)
      common/set5/c(150,150)

      write (2,10)
10    format(/'This run was done from a restart file.'/)

c     Read from the restart file.

      read (3,20)
20    format(//////)

      read (3,30) ( ipivot(i), jpivot(i), i = 1, nonods )
30    format(10x,i4,6x,i4)

      read (3,40)
40    format(//)

      read (3,100) ( ( a(i,j), c(i,j), j=1,nonods ), i=1,nonods )
100   format(10x,e20.14,2x,e20.14)

      return
      end


c     ******************* SUBROUTINE RSWRITE *****************
c     If requested, write the contents of pivot vectors and a
c     and c matrices into a restart file, for008.dat.

      subroutine rswrite

      implicit double precision (a-h,o-z)

      common/set1/nonods
      common/piv/ ipivot(150), jpivot(150)
      common/set4/a(150,150)
      common/set5/c(150,150)

      write (2,10)
10    format(/'A restart file was created.')

      write (3,20)
20    format(//////)
```

```
      write (3,30) ( i, ipivot(i), jpivot(i), i = 1, nonods )
30    format(i4,6x,i4,6x,i4)

      write (3,40)
40    format(//)

      write (3,100) ( (i,j,a(i,j),c(i,j),j=1,nonods), i=1,nonods )
100   format(i4,1x,i4,1x,e20.14,2x,e20.14)

      write (3,110)
110   format(//)

      return
      end
```

Input-Data File

```
This is a file of input data for program ''becur''
for the nickel problem corresponding to the experiment.
Fine nodal structure.
-line-
-line-

iprint    ispotpr    intspot      igrid (0 run, 1 both, 2 plot )
i          i          iii          i
0          1          100          0

irestrt                         ikeep
i                                 i
0                                 1

ibctype ( 0 natural; 1 essential )
i
0

guessv (mV)       (only used for essential bc at electrode)
sd.dddEsee
-1.0  e+02

guessi (mA/cm2)  (only used for natural bc at electrode)
sd.dddEsee
 5.0  e-02

tol
sd.dddEsee
 1.000e-06

  fracfp            fracnr
d.ddddddddd       d.ddddddddd
0.2               1.0

mctrmax    mswitch    mconsec
iiii        iiii       iiii
   1           0         10

nctrmax    ndivmax    ntotmax
iiii        iiii       iiii
  10           3        500

delv0          deldelv       nconmax       delpmax
dddddd.d       dddddd.d        iiii        dddddd.d
    -5.0         -100.0        300         -2000.0

nodelf
```

```
ii
 1

delvf  ( phi(CE) - phi(WE) ) - same at open ckt.  (mV)
index   dddddd.d
 1        -1200.0

ikin (1 Butler-Volmer, 2 Linear, 3 Tafel, 4 passivating)
i
4

xchcur  (mA/cm2)
sd.dddEsee
 2.5  e-03

alphaa            alphac
dd.dddd           dd.ddd
 0.5               0.5

nopts   number of cur-eta pairs to fit polar. curve
ii      (applicable only for ikin=4)
30

cur-eta pairs  (mA/cm2, mV)
point     cur        eta
     ddd.dddddd  dddd.d
1       0.0          0.0
2       0.197       30.4
3       0.393       60.9
4       0.590       91.3
5       0.787      122.0
6       0.983      152.0
7       1.967      183.0
8       6.883      213.0
9      12.29       243.0
10     19.67       274.0
11     29.50       304.0
12     35.40       335.0
13     38.35       365.0
14     34.41       396.0
15     24.58       426.0
16     16.22       456.0
17      0.98       487.0
18      0.5        517.0
19      0.5       1371.0
20      1.86      1523.0
21      4.23      1599.0
22      9.83      1675.0
23     19.66      1765.0
```

```
24     19.66       1780.0
25      9.83       1787.0
26      6.88       1828.0
27      9.83       1876.0
28     19.66       1904.0
29     49.15       1942.0
30     74.71       1980.0

curchar
sd.dddEsee
 5.233e+00

vchar
dddd.dd
  24.08

iaxism
i
1

nonods
iii
113

nonelec
iii
 69

scale
ddddd.d
   20.0

vertices of smallest rectangle enclosing the domain
xmin      xmax
ddddd.d ddddd.d
    0.0   250.0
ymin      ymax
ddddd.d ddddd.d
    0.0    20.0

 rinner
ddddddd.d
     40.0

node coordinates                    keys
node       r or y    z or x       bc  ode loc
        ddddddd.d dddddddd.d      ii  ii  ii
1              0.0      240.0      0   1   1
2              0.0      237.5      0   1   0
```

3	0.0	235.0	0	1	1
4	0.0	232.5	0	1	0
5	0.0	230.0	0	1	1
6	0.0	227.5	0	1	0
7	0.0	225.0	0	1	1
8	0.0	222.5	0	1	0
9	0.0	220.0	0	1	1
10	0.0	217.5	0	1	0
11	0.0	215.0	0	1	1
12	0.0	212.5	0	1	0
13	0.0	210.0	0	1	1
14	0.0	207.5	0	1	0
15	0.0	205.0	0	1	1
16	0.0	202.5	0	1	0
17	0.0	200.0	0	1	1
18	0.0	197.5	0	1	0
19	0.0	195.0	0	1	1
20	0.0	192.5	0	1	0
21	0.0	190.0	0	1	1
22	0.0	187.5	0	1	0
23	0.0	185.0	0	1	1
24	0.0	182.5	0	1	0
25	0.0	180.0	0	1	1
26	0.0	177.5	0	1	0
27	0.0	175.0	0	1	1
28	0.0	172.5	0	1	0
29	0.0	170.0	0	1	1
30	0.0	167.5	0	1	0
31	0.0	165.0	0	1	1
32	0.0	162.5	0	1	0
33	0.0	160.0	0	1	1
34	0.0	157.5	0	1	0
35	0.0	155.0	0	1	1
36	0.0	152.5	0	1	0
37	0.0	150.0	0	1	1
38	0.0	147.5	0	1	0
39	0.0	145.0	0	1	1
40	0.0	142.5	0	1	0
41	0.0	140.0	0	1	1
42	0.0	135.0	0	1	0
43	0.0	130.0	0	1	1
44	0.0	125.0	0	1	0
45	0.0	120.0	0	1	1
46	0.0	115.0	0	1	0
47	0.0	110.0	0	1	1
48	0.0	105.0	0	1	0
49	0.0	100.0	0	1	1
50	0.0	95.0	0	1	0
51	0.0	90.0	0	1	1

52	0.0	85.0	0	1	0
53	0.0	80.0	0	1	1
54	0.0	75.0	0	1	0
55	0.0	70.0	0	1	1
56	0.0	65.0	0	1	0
57	0.0	60.0	0	1	1
58	0.0	55.0	0	1	0
59	0.0	50.0	0	1	1
60	0.0	45.0	0	1	0
61	0.0	40.0	0	1	1
62	0.0	35.0	0	1	0
63	0.0	30.0	0	1	1
64	0.0	25.0	0	1	0
65	0.0	20.0	0	1	1
66	0.0	15.0	0	1	0
67	0.0	10.0	0	1	1
68	0.0	5.0	0	1	0
69	0.0	0.0	0	1	1
70	0.0	0.0	0	0	1
71	5.0	0.0	0	0	0
72	10.0	0.0	0	0	1
73	15.0	0.0	0	0	0
74	20.0	0.0	0	0	1
75	20.0	0.0	0	0	1
76	20.0	10.0	0	0	0
77	20.0	20.0	0	0	1
78	20.0	30.0	0	0	0
79	20.0	40.0	0	0	1
80	20.0	50.0	0	0	0
81	20.0	60.0	0	0	1
82	20.0	70.0	0	0	0
83	20.0	80.0	0	0	1
84	20.0	90.0	0	0	0
85	20.0	100.0	0	0	1
86	20.0	110.0	0	0	0
87	20.0	120.0	0	0	1
88	20.0	130.0	0	0	0
89	20.0	140.0	0	0	1
90	20.0	150.0	0	0	0
91	20.0	160.0	0	0	1
92	20.0	170.0	0	0	0
93	20.0	180.0	0	0	1
94	20.0	190.0	0	0	0
95	20.0	200.0	0	0	1
96	20.0	207.5	0	0	0
97	20.0	215.0	0	0	1
98	20.0	222.2	0	0	0
99	20.0	230.0	0	0	1
100	20.0	235.0	0	0	0

101	20.0	240.0	0	0	1
102	20.0	245.0	0	0	0
103	20.0	250.0	0	0	1
104	20.0	250.0	1	2	1
105	15.0	250.0	1	2	0
106	10.0	250.0	1	2	1
107	5.0	250.0	1	2	0
108	0.0	250.0	1	2	1
109	0.0	250.0	0	0	1
110	0.0	247.5	0	0	0
111	0.0	245.0	0	0	1
112	0.0	242.5	0	0	0
113	0.0	240.0	0	0	1

Appendix A-2

Boundary-Element Code for Attached-Bubble Model

The following Fortran code was used in the calculations of Chapter 2. A sample input-data file is given at the end of the program listing. Important variables are defined in subroutine 'input' or as they appear in the code. Subroutines 'matrix,' 'makeb,' 'gauss1,' 'gauss2,' 'rsread' and 'rswrite' as well as functions 'gquad,' 'ei1,' 'ei2,' 'ggrand' and 'hgrand' are listed in Appendix A-1.

```
      program bebub

c     This program is intended to solve for the current
c     distribution in the vicinity of an electrolytic bubble
c     attached to an electrode.  Laplace's equation is to
c     be solved for two variables:
c         subscript                variable
c           1                 potential, phi
c           2                 concentration of dissolved gas, cg

c     We solve for the concentration over the diffusion boundary
c     layer (dbl) or ''domain 2,'' while phi is solved over a larger
c     region, ''domain 1'', comprising the dbl, with matching
c     conditions at the electrode.

c     The overpotential acting at the electrode surface comprises
c     activation overpotential (given by the Butler-Volmer
c     expression) and the ''incomplete'' concentration overpotential
c     (the remainder of the ''complete'' concentration overpotenial
c     is accounted for in the electrostatic potential drop across
c     the boundary layer. )

c     The program is general to axisymmetric and planar problems.

c     This is a boundary-element program using quadratic elements.
c     Be sure that, for both domains, the node numbering is done
c     counterclockwise beginning with the first electrode node.

      implicit double precision (a-h,o-z)

      common/nr1/ surf, delsurf
      common/piv1/ ipivot1(150), jpivot1(150)
      common/piv2/ ipivot2(150), jpivot2(150)
      common/set0/nonelec
      common/set1/nonods1,nonods2
      common/set2/coords1(150,2),coords2(150,2)
      common/set2a/keyloc1(150),keyloc2(150)
      common/set3/iaxism,rinner
      common/set41/a1(150,150)
      common/set42/a2(150,150)
      common/set51/c1(150,150)
      common/set52/c2(150,150)
      common/set61/x1(150)
      common/set62/x2(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set81/keybc1(150),keyode1(150)
      common/set82/keybc2(150),keyode2(150)
      common/set10/tol,param,paramnr,nctrmax,nmaxnr,nctr,nctrnr
```

```
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set13/iflag
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast
      common/set22/irestrt,ikeep
      common/set23/phiefar,phiede,blt,edt,phie(150)
      common/set24/igrid,xmin,xmax,ymin,ymax

      dimension b1(150), b2(150)
      dimension surf(150), surfnew(150), delsurf(150)

c     Begin by reading in all the data for the problem.
c     Call the input subroutine.

      call input

c     If requested, create a plot of the problem geometry.
c     Act according to the key ''igrid'':
c       igrid=0       run but no plot
c       igrid=1       run and plot
c       igrid=2       plot only ( no run )
      if ( igrid . lt . 1 ) go to 10
      call grid(nonods1,coords1,keyode1,keyloc1)
      call grid(nonods2,coords2,keyode2,keyloc2)
      if ( ixport . ne . 1 )
     $call grid(nonods2,coords2,keyode3,keyloc2)
      if ( igrid . eq . 2 ) stop

c     See if this run is a restart ( irestrt = 1 ).  If so, skip
c     the costly ''matrix'' and ''gauss1'' routines and merely
c     read in their products from the restart file, for003.dat.

10    if ( irestrt . eq . 0 ) go to 100
      call rsread
      go to 1003

c     Call subroutine ''matrix'' to set up each matrix problem
c     G q = H u (same for problems 2 and 3) and rearrange
c     to get A x = C bc, different now for each problem.
c     We form matrices A1, C1, A2 and C2.

100   call matrix(nonods1,coords1,keyloc1,a1,c1,keybc1,keyode1)
      call matrix(nonods2,coords2,keyloc2,a2,c2,keybc2,keyode2)
      if ( ixport . ne . 1 )
```

```
     ucall matrix(nonods2,coords2,keyloc2,a3,c3,keybc3,keyode3)

c     Send off both (all three) A matrices to be forward reduced
c     in subroutine ''Gauss1.'' The reduced matrices are retained
c     in their original storage locations along with a record of
c     row manipulations. Vectors ''ipivot'' and ''jpivot'' are
c     formed for each problem, as a record of row and column
c     exchanges carried out.

      call gauss1(nonods1,a1,ipivot1,jpivot1,iflag1)
      call gauss1(nonods2,a2,ipivot2,jpivot2,iflag2)
      if ( ixport . ne . 1 )
     ucall gauss1(nonods2,a3,ipivot3,jpivot3,iflag3)

c     If any matrix is singular, stop everything.
      if ( iflag1 . eq . 1 ) stop
      if ( iflag2 . eq . 1 ) stop
      if ( iflag3 . eq . 1 ) stop

c     If requested, write the contents of pivot vectors and a and
c     c matrices into a restart file, for008.dat.

1003  if ( ikeep . eq . 0 ) go to 1005
      call rswrite

c     Calculate ''phiefar'', the value of phi just off the electrode
c     surface, far from the bubble. This is the same potential
c     that would be calculated for a one-dimensional problem
c     with no bubble present. The routine ''onedim'' makes
c     this calculation.

1005  call onedim

c     We now have forward-reduced forms of both (all three) system
c     matrices and we're ready start into the iteration cycle.

c     Initialize surf, either phi or its derivative at the electrode
c     surface, according to the value of ifast.

c       Case of fast kinetics:
        if ( ifast . eq . 0 ) go to 118
        do 117 i = 1, nonelec
          surf(i) = bc1(i)
117       continue
        go to 120

c       Case of slow kinetics:
118     do 119 i = 1, nonelec
          surf(i) = epsilon / scale
```

```
119       continue

c     Initialize parameters for iteration, convergence,
c     divergence tests.
120   nctr    = 0
      nctrnr  = 0
      ndiv    = 0
      ndivmax = 50
      tmaxdiv = 1000.
      frac    = param
      tmax    = 100.
      tolnr   = tol * 2.
      iterkey = 1

c     **************** Start of Iteration ******************

c     See if we're at the point of changeover to N.R.
121   if ( nctr.eq.nctrmax . and . nctrnr.eq.0 ) go to 122
      if ( tmax.le.tolnr   . and . nctrnr.eq.0 ) go to 122
      go to 124

c     At the point of changeover to N.R. reset some parameters.
122   iterkey = 2
      ndivmax = 2
      ndiv = 0
      frac = paramnr

c     Branch to appropriate type of iteration.
124   if ( iterkey . eq . 2 ) go to 126

c     ********* Substitution Iteration *********
c     ''iterkey'' equals 1.

      nctr  = nctr  + 1
      keepoff = 0
      if ( ifast . eq . 0 )  call gslow(surf,surfnew,keepoff)

      if ( ifast . eq . 1 )  call gfast(surf,surfnew)
      go to 130

c     ********* Newton Raphson Iteration *******
c     ''iterkey'' equals 2.

126   nctrnr = nctrnr + 1
      frac = paramnr
      if ( nmaxnr . eq . 0 ) go to 172

c     Call subroutine ''nr'' to operate on the vector ''cur''
```

```
c       returning a vector of corrections ``delcur.''
        call nr

        do 128 i = 1, nonelec
          surfnew(i) = surf(i) + delsurf(i)
128       continue


c       ************* Convergence Test, Update ``surf'' ***********

c       For the convergence criterion, find the biggest change in
c       surf on the electrode.  Also, here is where you make the
c       relaxed substitution to form new guess surf for the next
c       iteration.
130     tmaxold = tmax
        tmax    = 0.
c       Find the largest absolute change in surf.
        do 138 i = 1, nonelec
          t = dabs( surfnew(i) - surf(i) )
          if ( t . gt . tmax ) tmax = t
          surf(i)  =  frac * surfnew(i)  +  ( 1.0 - frac ) * surf(i)
c         Impose some walls:
c         if ( ifast.eq.1 . and . surf(i).lt.wall ) surf(i) = wall
c         if ( ifast.eq.0 . and . surf(i).lt.0.      ) surf(i) = 0.
138       continue

c       Normalize tmax.
147     if ( ifast . eq . 1 )  tmax = dabs( tmax )
        if ( ifast . eq . 0 )  tmax = dabs( tmax * scale / epsilon )

c       As an diagnostic, print out tmax for each iteration.
c       if ( nctr . eq . 1 ) then
c         write (2,140)
c140        format(/2x,'nctr',3x,'nctrnr',3x,'tmax')
c         endif
c       write (2,142) nctr, nctrnr, tmax
c142    format(2x,i4,4x,i4,4x,g10.4)

c       If you have converged, call the output subroutine.
        if(tmax.gt.tol) go to 149
        write (2,146)
146     format(/'Of course it converged!'/)
        write (4,148)
148     format('Of course it converged!'/)
        go to 180

c       If you've exceeded the max allowed iterations, say so
c       and quit.
149     if ( nctrnr . ge . nmaxnr  ) go to 172
```

```
c   Test for divergence two different ways:

c   1) If tmax exceeds tmaxdiv then bail out.
      if ( tmax . lt . tmaxdiv ) go to 152
      write (2,150)
      write (4,150)
150   format( /'divergence.  tmax exceeds tmaxdiv.'/ )
      go to 180

c   2) Keep a running tally of the pattern of convergence.
c     If tmax goes UP ndivmax times more than it goes DOWN
c     over any interval, then announce this and quit.
152   if ( tmax . ge . tmaxold ) ndiv = ndiv + 1
      if ( tmax . lt . tmaxold ) ndiv = ndiv - 1
      if ( ndiv . le . 0       ) ndiv = 0
      if ( ndiv . le . ndivmax ) go to 156
      write (2,154)
      write (4,154)
154   format( /'divergence.  ndiv exceeds ndivmax.'/ )
      go to 180

c     If you have neither converged nor shown signs of
c     divergence, do another iteration.
156   go to 121

172   write (2,173)
      write (4,173)
173   format ('You exceeded nmaxnr iterations.')

180   call output

      stop
      end



c     ******************* SUBROUTINE INPUT ***********************
      subroutine input

      implicit double precision (a-h,o-z)

c     This subroutine reads in all the input data and, if desired,
c     prints it all back out again as a check.
      common/set0/nonelec
      common/set1/nonods1,nonods2
      common/set2/coords1(150,2),coords2(150,2)
```

```
      common/set2a/keyloc1(150),keyloc2(150)
      common/set3/iaxism,rinner
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set73/bc3(150)
      common/set81/keybc1(150),keyode1(150)
      common/set82/keybc2(150),keyode2(150)
      common/set83/keybc3(150),keyode3(150)
      common/set10/tol,param,paramnr,nctrmax,nmaxnr,nctr,nctrnr
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set13/iflag
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast
      common/set22/irestrt,ikeep
      common/set23/phiefar,phiede,blt,edt
      common/set24/igrid,xmin,xmax,ymin,ymax

c     Read in all the input.

      read (1,301)
301   format(////)

c     Read iprint (print all data unless iprint equals zero),
c     ispotpr (values are spot-printed every 'intspot' iterations
c     if ispotpr = 1), and intspot.
c     Also read igrid, key for plotting grid.
c          igrid=0        run, no plot
c          igrid=1        run and plot
c          igrid=2        plot, no run
      read (1,302) iprint, ispotpr, intspot, igrid
302   format(///i1,9x,i1,9x,i3,10x,i1)

c     Read iaxism.  If this reads zero, the problem is not
c     axisymmetric.  A value of one or anything else indicates
c     axisymmetry.
      read (1,303) iaxism
303   format(///i1)

c     Read nonods1, the number of nodes in the nodal structure for
c     the large domain ( over which phi is solved ).
      read (1,304) nonods1
304   format(///i3)

c     Read nonods2, the number of nodes in the nodal structure for
```

```
c       the diffusion boundary layer  ( the small domain over which
c       c and cg are solved ).
        read (1,305) nonods2
305     format(///i3)

c       Read nonelec, the number of nodes on the electrode surface.
        read (1,306) nonelec
306     format(///i3)

c       Read in the scale factor for the model representation
c       of the domaim.  SCALE is the number of model length units
c       per dimensionless length unit.  Since we have
c       nondimensionalized the problem with respect to
c       the bubble radius, SCALE is the bubble radius expressed
c       in model units.  Generally, we take scale > 1 to get
c       the coordinates in round numbers.
        read (1,307) scale
307     format(///f7.1)

c       Read blt, the Boundary-Layer Thickness, in bubble radii.
        read (1,308) blt
308     format(///f7.3)

c       Read edt, the Extended-Domain Thickness, in bubble radii.
        read (1,3085) edt
3085    format(///f7.3)

c       Read vertices of smallest rectangle that encloses the domain
c       (for graphics use.)
        read (1,309) xmin, xmax
309     format(////f7.1,1x,f7.1)
        read (1,310) ymin, ymax
310     format(//f7.1,1x,f7.1)

c       Read in the value of rinner.  This is the distance (in the
c       distance units of the model) from the innermost point of
c       the nodal structure to the centerline.  If the problem is
c       not axisymmetric, we don't need rinner, but we must read
c       a value anyhow.
        read (1,311) rinner
311     format(///f9.1)

c       Read the stoichiometric coefficients of the reacting
c       species, s1 and sg.  These are ''Newmanclature'', p 31.
        read (1,312) s1, sg
312     format(///f4.2,6x,f4.2)

c       Read enovrz2, ratio of number of electrons in the reaction
c       to the valence on the inert ion of the binary electrolyte.
```

```
      read (1,3125) enovrz2
3125  format(///f5.2)

c     Read ixport, a key signifying the transport model to be used
c     in this problem:
c         ixport                    transport model
c           1               binary, uniform conductivity
      read (1,313) ixport
313   format(///i1)

c     Read epsilon, the value gradient of phi far away.
      read (1,315) epsilon
315   format(///e10.3)

c     Read theta, the proportionality between normal gradients of
c     phi and cg at the electrode.
      read (1,316) theta
316   format(///e10.3)

c     Read omega, the proportionality between normal gradients of
c     phi and cg at the electrode.
      read (1,3165) omega
3165  format(///e10.3)

c     Is this run a restart of an earlier one?
c     ( yes: irestrt = 1;  no: irestrt = 0 )
c     Do we want to save matrix values and pivot keys
c     in a restart file?  ( yes: ikeep = 1;  no: ikeep = 0 )
      read (1,317) irestrt, ikeep
317   format(///i1,29x,i1)

c     Read ''speed'' the ratio of exchange current density,
c     taken at the bulk concentrations of ion and gas,
c     to the magnitude of the current density far from the bubble.
c     and the value of ifast ( 1 for fast kinetics, 0 for slow,
c     calling for the inverted loop.)
      read (1,318) speed, ifast
318   format(///e10.3,5x,i1)

c     Read the kinetic symmetry factor beta.
      read (1,319) beta
319   format(///f4.2)

c     Calculate exponents ql and qg for the kinetic expression,
c     according to Newman, p 174.
      if ( sl . gt . zero ) ql = 0.
      if ( sl . lt . zero ) ql = - sl
      if ( sg . gt . zero ) qg = 0.
      if ( sg . lt . zero ) qg = - sg
```

```
      el  =  ql  +  beta * sl
      eg  =  qg  +  beta * sg

c     Read the value of the relative convergence tolerance
      read (1,320) tol
320   format(///e10.3)

c     Read in the convergence parameters: ''param'', the
c     fraction of the new value used in the relaxed
c     substitution, and ''paramnr'' the fraction of the NR
c     correction term used.
      read (1,321) param, paramnr
321   format(///f10.8,4x,f7.5)

c     Read in the maximum number of iterations allowed:
c     ''nctrmax'' for relaxed substitution, and ''nmaxnr''
c     for newton raphson.
      read (1,324) nctrmax, nmaxnr
324   format(///i4,7x,i4)

c     Read the coordinates of each node in global order.
c     As a convention, we number counter clockwise beginning
c     with the electrode.  Also read in the keys for the boundary
c     condition type (''keybc'') and the boundary-segment type
c     (''keyode'').

c       First the coordinates of the extended domain:
        read (1,325)
325     format(///)
        read (1,326) (coords1(i,1),coords1(i,2),keybc1(i),
     $    keyode1(i),keyloc1(i),i=1,nonods1)
326     format(8x,f9.1,1x,f9.1,4x,i2,2x,i2,2x,i2,4x)

c       Then the coordinates of the boundary layer:
        read (1,3255)
3255    format(///)
        read (1,3265) (coords2(i,1),coords2(i,2),keybc2(i),
     $    keyode2(i),keyloc2(i),keybc3(i),keyode3(i),i=1,nonods2)
3265    format(8x,f9.1,1x,f9.1,4x,i2,2x,i2,2x,i2,4x,i2,2x,i2)

c     Since this program is general to two different iteration
c     schemes, we set the values of keybc1 here according to the
c     value of ifast.  We imply potential bc (keybc1 = 1) for
c     fast kinetic scheme (ifast = 1), and flux bc (keybc1 = 0)
c     for slow kinetic scheme (ifast = 0).  Thus it doesn't matter
c     what is read in for keybc1 on the electrode above.
      do 3269 i = 1, nonelec
        keybc1(i) = ifast
3269    continue
```

```
c     Initialize the boundary-condition vectors to zero.
      do 330 i = 1, nonods1
        bc1(i) = 0.
330     continue
      do 3305 i = 1, nonods2
        bc2(i) = 0.
        bc3(i) = 0.
3305    continue

c     Fix the the phi derivative at the top of the extended domain
c     at a value equal to minus epsilon ( minus because this is
c     an inward-pointing-surface-normal derivative. )
      do 331 i=1,nonods1
        if ( keyode1(i) . ne . 2 ) go to 331
        bc1(i) = - epsilon / scale
331     continue

c     Initialize the phi boundary condition on the electrode
c     (keyode1=1).  If ifast=1, we initialize phi to zero;  if
c     ifast=0, we initialize the derivative to epsilon/scale.
c     Also set keybc1=1 for ifast=1; keybc1=0 for ifast=0 on
c     the electrode.
      do 333 i=1,nonelec
        bc1(i) = 0.
        keybc1(i) = 1
        if ( ifast . eq . 1 ) go to 333
        bc1(i) = epsilon / scale
        keybc1(i) = 0
333     continue

c     Set c and cg to unity at the appropriate nodes.
      do 335 i=1,nonods2
        if ( keyode2(i) . ne . 2 ) go to 3355
        bc2(i) = 1.0
3355    if ( keyode3(i) . ne . 2 ) go to 335
        bc3(i) = 1.0
335     continue

c     At the electrode surface, initialize the normal concentration
c     derivatives to the values they will have far from bubble.
      do 344 i=1,nonods2
        if ( keyode2(i) . ne . 1 ) go to 3445
        bc2(i) =  omega * epsilon / scale
3445    if ( keyode3(i) . ne . 1 ) go to 344
        bc3(i) =  theta * epsilon / scale
344     continue

c     Set the values of the gaussian quadrature abscissas and
c     weighting factors.
```

```
c       Twelve-point gaussian quadrature coefficients:

        nogauss = 12

        abgauss(1)  =  - 0.981560634246719
        abgauss(2)  =  - 0.904117256370475
        abgauss(3)  =  - 0.769902674194305
        abgauss(4)  =  - 0.587317954286617
        abgauss(5)  =  - 0.367831498998180
        abgauss(6)  =  - 0.125233408511469
        abgauss(7)  =    0.125233408511469
        abgauss(8)  =    0.367831498998180
        abgauss(9)  =    0.587317954286617
        abgauss(10) =    0.769902674194305
        abgauss(11) =    0.904117256370475
        abgauss(12) =    0.981560634246719

        wfgauss(1)  =    0.047175336386512
        wfgauss(2)  =    0.106939325995318
        wfgauss(3)  =    0.160078328543346
        wfgauss(4)  =    0.203167426723066
        wfgauss(5)  =    0.233492536538355
        wfgauss(6)  =    0.249147045813403
        wfgauss(7)  =    0.249147045813403
        wfgauss(8)  =    0.233492536538355
        wfgauss(9)  =    0.203167426723066
        wfgauss(10) =    0.160078328543346
        wfgauss(11) =    0.106939325995318
        wfgauss(12) =    0.047175336386512

        return
        end


c       ***************** SUBROUTINE ONEDIM ****************
c       This routine solves the one dimensional problem which applies
c       radially far from the bubble. We calculate electrode-surface
c       values of c, cg, etac, etas and phie ( namely cefar, cgefar,
c       etacfar, etasfar and phiefar ). We print these out and return
c       the value of phiefar to the main routine. We go on to
c       calculate the value of potential at the edge of the extended
c       domain, ''phiede.'' One other odd job that gets done in this
c       routine is to set the potential at the edge of the extended
c       domain (bc1(i)) to the value ''phiede'' for transport cases
c       ixport = 1 or 3.
```

```
      subroutine onedim

      implicit double precision (a-h,o-z)

      common/set1/nonods1,nonods2
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast
      common/set23/phiefar,phiede,blt,edt
      common/set71/bc1(150)
      common/set81/keybc1(150),keyode1(150)

c     Concentrations at the electrode, far from the bubble:
      cfar  =  1.0  -  epsilon / theta
      if ( ixport . eq . 1 ) cfar = 1.0
      cgfar =  1.0  - blt * epsilon / omega

c     Concentration overpotential:
      etacfar  =  - s1 * dlog ( cfar )  -  sg * dlog ( cgfar )

c     Activation overpotential:
      etasfar  =   0.
      curfar   = - epsilon
      xchcur0  =   speed * dabs(curfar)
      xchcur   =   xchcur0 * ( cfar ** e1 ) * ( cgfar ** eg )
      ratio    =   curfar / xchcur
      call bvkin(ratio,etasfar,beta)

c     Total overpotential:
      etatfar  =  etacfar  +  etasfar

c     Value of potential just outside the diffuse double layer
c     refered to the potential of the electrode metal
      phiefar  =  - etatfar

c     Calculate the potential at the extended domain edge (e.d.e.)
c     This is only done for and only relevant to the two concentrated
c     transport cases ( ixport = 1 or 3 ).
      phiede  =  phiefar  +  epsilon * edt
      if ( ixport . eq . 2 ) phiede = 0.

c     Set bc1(i) at the edge of the extended domain to phiede.
c     iff a dirichlet b.c. is called for there.
      do 100 i=1,nonods1
        key = keyode1(i)
        kbc = keybc1(i)
        if ( key . ne . 2 ) go to 100
        if ( kbc . ne . 1 ) go to 100
```

```
        bc1(i) = phiede
100     continue

c     Print out some key values to help debug.
      write (2,110) cgfar, phiefar, phiede,etacfar,etasfar
110   format(//'cgfar'/g10.4//'phiefar'/g15.9//'phiede'/g15.9//
     u  'etacfar'/g10.4//'etasfar'/g10.4/)

      return
      end



c     ***************** SUBROUTINE NR ********************
c     This routine does multivariable Newton-Raphson on the
c     function evaluated in either one of the g subroutines
c     (``gfast'' or ``gslow'').

c     The scheme is ``xnew = g(xold)'' which we solve by
c     defining the function ``f(x) = g(x) - x'', and then
c     solving the system ``FPRIME * delx = - f'' for the
c     vector of corrections ``delx.''

      subroutine nr

      implicit double precision (a-h,o-z)

      common/nr1/ x(150), delx(150)
      common/piv1/ ipivot1(150), jpivot1(150)
      common/piv2/ ipivot2(150), jpivot2(150)
      common/set0/nonelec
      common/set1/nonods1,nonods2
      common/set41/a1(150,150)
      common/set42/a2(150,150)
      common/set51/c1(150,150)
      common/set52/c2(150,150)
      common/set61/x1(150)
      common/set62/x2(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set81/keybc1(150),keyode1(150)
      common/set12/scale
      common/set16/nctrsp,ispotpr,intspot
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast

      dimension xp(150), xpnew(150)
```

```
      dimension b1(150), b2(150)
      dimension fprime(150,150), f(150)
      dimension ipivot4(150), jpivot4(150)

c     Set ``pert'' the value of the perturbation in x used
c     to evaluate the numerical derivative.

      pert = 0.0001

c   Create the jacobian matrix ``fprime'' in three steps:

c     0) Evaluate function of unperturbed set.
c     This may seem strange but we do this again in step 2
c     for keeps; the only reason we do it here first is to
c     fix the value of the offset voltage in the gslow routine
c     (if applicable).  If this offset voltage is not held
c     fixed while we perturb the xp's, then the jacobian will
c     turn out singular.  ``keepoff'' is an argument of ``glsow''
c     that instructs the gslow routine whether to KEEP the
c     OFFset voltage calculated during the last time
c     through the routine, or to calculate a new offset value
c     for the present call.
      do 5 i = 1, nonelec
        xp(i) = x(i)
5       continue
      keepoff = 0
      if ( ifast . eq . 0 )  call gslow(xp,xpnew,keepoff)
      if ( ifast . eq . 1 )  call gfast(xp,xpnew)

c     1) fill matrix with functional evaluations of perturbed x's.
      do 30 j = 1, nonelec
        do 10 i = 1, nonelec
          xp(i) = x(i)
          if ( i . eq . j ) xp(i) = xp(i) + pert
10        continue
        keepoff = 1
        if ( ifast . eq . 0 )  call gslow(xp,xpnew,keepoff)
        if ( ifast . eq . 1 )  call gfast(xp,xpnew)
        do 20 i = 1, nonelec
          fprime(i,j) = xpnew(i)
20        continue
30      continue

c     2) Evaluate function of unperturbed set.
      do 40 i = 1, nonelec
        xp(i) = x(i)
40      continue
      keepoff = 0
      if ( ifast . eq . 0 )  call gslow(xp,xpnew,keepoff)
```

```
      if ( ifast . eq . 1 )  call gfast(xp,xpnew)

c     3) Subtract vector ''xpnew'' from each column of the matrix.
c     Then divide each difference by pert.
c     Lastly, subtract unity from each diagonal element.
      do 70 j = 1, nonelec
        do 60 i = 1, nonelec
          fprime(i,j) = ( fprime(i,j) - xpnew(i) ) / pert
          if ( i . eq . j ) fprime(i,j) = fprime(i,j) - 1.0
60        continue
70      continue

c     Evaluate the vector ''f,''  ( f(x) = g(x) - x ).
c     (Recall that the last evaluation of g was for the
c     unperturbed vector x.)
c     Actually fill vector f with MINUS the value it should have
c     so that we can consider delx an additive correction.
      do 80 i = 1, nonelec
        f(i) = - ( xpnew(i) - x(i) )
80      continue

c     Solve the system ''FPRIME * delx = f'' using the solvers
c     ''gauss1'' and ''gauss2'' also used for the field problem.

      call gauss1(nonelec,fprime,ipivot4,jpivot4,jacsing)
      call gauss2(nonelec,fprime,delx,f,ipivot4,jpivot4,0)

      return
      end



c     ****************** SUBROUTINE GFAST *********************
c     This routine operates on a set of potential values at the
c     electrode ''phi'' and returns a new set ''phinew,'' which
c     is generated by cycling through the coupled boundary-value
c     problem once.

      subroutine gfast(phi,phinew)
      implicit double precision (a-h,o-z)

      common/piv1/ ipivot1(150), jpivot1(150)
      common/piv2/ ipivot2(150), jpivot2(150)
      common/set0/nonelec
      common/set1/nonods1,nonods2
      common/set41/a1(150,150)
      common/set42/a2(150,150)
      common/set51/c1(150,150)
```

```
      common/set52/c2(150,150)
      common/set61/x1(150)
      common/set62/x2(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set81/keybc1(150),keyode1(150)
      common/set12/scale
      common/set16/nctrsp,ispotpr,intspot
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast
      common/set23/phiefar,phiede,blt,edt,phie(150)

      dimension b1(150), b2(150)
      dimension phi(150), phinew(150)

c     Set bc1(i).
      do 115 i=1,nonelec
        bc1(i) = phi(i)
115     continue

c     Apply the BCs; make vector b1.
      call makeb(nonods1,c1,bc1,b1)

c     Call subroutine ''gauss2'' to back-substitute and get
c     nodal derivatives.
      neumann = 0
      call gauss2(nonods1,a1,x1,b1,ipivot1,jpivot1,neumann)

c     Relate concentration derivatives to phi derivatives
c     at electrode.
      do 120 i = 1, nonelec
        bc2(i) = x1(i) / omega
        if ( ixport . ne . 1 )
     u  bc3(i) = x1(i) / theta
120     continue

c     Apply the BCs; make vectors b2 and b3.
      call makeb(nonods2,c2,bc2,b2)
      if ( ixport . ne . 1 )
     ucall makeb(nonods2,c3,bc3,b3)

c     Call subroutine ''gauss2'' to back-substitute and get
c     nodal concentrations.
      neumann = 0
      call gauss2(nonods2,a2,x2,b2,ipivot2,jpivot2,neumann)
      if ( ixport . ne . 1 )
     ucall gauss2(nonods2,a3,x3,b3,ipivot3,jpivot3,neumann)
```

```
      do 130 i = 1, nonelec

c       Concentration overpotential:

        cg = x2(i)
c       Set a wall:
        if ( cg . lt . 1.0 ) cg = 1.0

        c  = 1.0
        if ( ixport . ne . 1 ) c = x3(i)

        etac  =  - s1 * dlog ( c )  -  sg * dlog ( cg )

c       Activation overpotenital:

c       Local exchange current density depends on concentration:
        curfar   = - epsilon
        xchcur0  =   speed  *  dabs( curfar )
        xchcur   =   xchcur0 * ( c ** e1 ) * ( cg ** eg )

        cur  =  - x1(i) * scale
        ratio    =   cur / xchcur
        etas  =   0.
        call bvkin(ratio,etas,beta)

c       Total overpotential:
        etat  =  etac  +  etas

c       Calculate new surface values of phi ``phinew''.
        phinew(i) = - etat

130     continue


      return
      end



c     ***************** SUBROUTINE GSLOW ********************
      subroutine gslow(grad,gradnew,keepoff)

c     This routine begins with an estimate for phi derivative at
c     the electrode surface, and produces a new set of derivatives
c     by first solving for both phi and concentration at the
c     electrode and then using an inverted kinetic expression to
c     give current densities.
```

```
      implicit double precision (a-h,o-z)

      common/piv1/ ipivot1(150), jpivot1(150)
      common/piv2/ ipivot2(150), jpivot2(150)
      common/set0/nonelec
      common/set1/nonods1,nonods2
      common/set41/a1(150,150)
      common/set42/a2(150,150)
      common/set51/c1(150,150)
      common/set52/c2(150,150)
      common/set61/x1(150)
      common/set62/x2(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set81/keybc1(150),keyode1(150)
      common/set12/scale
      common/set16/nctrsp,ispotpr,intspot
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast
      common/set23/phiefar,phiede,blt,edt,phie(150)

      dimension b1(150), b2(150)
      dimension grad(150), gradnew(150), phisurf(150)
      dimension ghold(150), phold(150)

c     Set bc1(i), bc2(i), and bc3(i).
      do 115 i=1,nonelec
        bc1(i)  =  grad(i)
        bc2(i)  =  grad(i) / omega
        if ( ixport . ne . 1 )
     $  bc3(i)  =  grad(i) / theta
115     continue

c     Apply the BCs of each problem.  Make vectors b1 and b2.
      call makeb(nonods1,c1,bc1,b1)
      call makeb(nonods2,c2,bc2,b2)

c     Determine if field problem 1 is a Neumann problem and
c     set 'neumann' accordingly.  ('neumann = 1' means we have
c     a Neumann problem.)
      do 116 i = 1, nonods1
        neumann = 1
        key = keybc1(i)
        if ( key . eq . 1 ) neumann = 0
116     continue

c     Call subroutine ``gauss2'' to back-substitute and get
```

```
c     node values ``x'' for each field variable.
      call gauss2(nonods1,a1,x1,b1,ipivot1,jpivot1,neumann)
      call gauss2(nonods2,a2,x2,b2,ipivot2,jpivot2,0)
      if ( ixport . ne . 1 )
     $call gauss2(nonods2,a3,x3,b3,ipivot3,jpivot3,0)

c     We must correct the potentials calculated in ``gauss2''
c     by adding a constant ``offset'', because the reference
c     for these potentials is arbitrary in a neumann problem.
c     Another important consideration in a Neumann problem is
c     that the specified currents along the domain boundary
c     must satisfy an overall balance.  If they don't, there's
c     no way they can correspond to a solution to Laplace's
c     equation.  What we do is this:  begin with a uniform
c     current distribution on the electrode which satisfies the
c     overall balance.  Proceed through gauss2 and the overpotential
c     conditions to generate a new current distribution HAVING
c     GUESSED a value for ``offset.'' We iterate at this point:
c     trying values of offset until finally a current profile is
c     obtained which satisfies the overall balance.  (It is
c     reasonable to assume that only one value of offset will do
c     the trick )


c     For now we're treating a Neumann problem, ixport = 1.
c     if ( ixport  . eq . 2 ) go to 119
c     if ( ixport  . eq . 3 ) go to 119
c     if ( neumann . ne . 1 ) go to 119

c     Only necessarily true for transport case #1:
      cw   =  1.0

c     Do all this to get a first estimate of ``offset''.

c     ``cgw'', dissolved gas concentration at the wall:
      cgw  =  x2(nonelec)
      if ( cgw . lt . 1.0 ) cgw = 1.0

c     Calculate ``etasw'', activation overpotential at the wall:
      etasw    =   0.
      curw     =   - bc1(nonelec) * scale
      xchcur0  =   speed * dabs(epsilon)
      xchcur   =   xchcur0 * ( cw ** el ) * ( cgw ** eg )
      ratio    =   curw / xchcur
      call bvkin(ratio,etasw,beta)

c     Calculate ``etacw'', concentration overpotential at the wall:
      etacw  =  - sl * dlog ( cw )  -  sg * dlog ( cgw )
```

```
c       Calculate phiw, potential just outside the diffuse double
c       layer off of point w on the electrode surface:
        phimet  =  0.
        phiw    =  phimet  -  etacw  -  etasw

        if ( keepoff . eq . 0 )   offset  =  phiw  -  x1(nonelec)

c       Set up for Newton-Raphson
        pert  =  1.0e-5
        nnr = 0
        nnrmax = 1000
        tolnr  =  1.0e-10

        do 42 i = 1, nonelec
          phisurf(i) = x1(i)
42        continue

44      nnr = nnr + 1

        offsetp = offset + pert

        fp = extra(offsetp,gradnew,phisurf)
        f  = extra(offset, gradnew,phisurf)

        fprime = ( fp - f ) / pert

        deloff = - f / fprime

        call flag(pi)

        if ( deloff . gt .  0.01 ) deloff =  0.01
        if ( deloff . lt . -0.01 ) deloff = -0.01

        factor = 1.0
        offset  =  offset  +  factor * deloff

        if ( dabs(deloff) . le . tolnr ) go to 84

        if ( nnr . ge . nnrmax ) then
          write (2,74)
74        format('exceeded nnrmax iterations in offset nr')
          stop

        else
          go to 44

        endif

84      do 86 i = 1, nonods1
```

```
        x1(i)  =  x1(i)  +  offset
86      continue

      return
      end



c     ******************** FUNCTION EXTRA ****************
      function extra(offset,gradnew,phisurf)

      implicit double precision (a-h,o-z)

      common/set0/nonelec
      common/set1/nonods1,nonods2
      common/set2/coords1(150,2),coords2(150,2)
      common/set2a/keyloc1(150),keyloc2(150)
      common/set3/iaxism,rinner
      common/set61/x1(150)
      common/set62/x2(150)
      common/set63/x3(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set73/bc3(150)
      common/set81/keybc1(150),keyode1(150)
      common/set82/keybc2(150),keyode2(150)
      common/set83/keybc3(150),keyode3(150)
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast
      common/set23/phiefar,phiede,blt,edt,phie(150)

      dimension gradnew(150), grad(150), phisurf(150), phinew(150)

c     Adjust all the electrode potentials.
      do 118 i = 1, nonelec
        if ( keybc1(i) . ne . 0 ) go to 118
        phinew(i)  =  phisurf(i)  +  offset
118     continue

c     The next section of the program deals with the potential
c     matching condition at the electrode.

c     If we are in a dilute, binary electrolyte (ixport=2),
```

```
c       evaluate the potential at each electrode surface node,
c       using subroutines ''phieval'' and ''phistep'' along with
c       functions ''gquad'' and ''eline'' to do the necessary
c       line integral.
119     if ( ixport . eq . 2 ) call phieval

c       Use the overpotential condition to return a revised value
c       of current density at each electrode node.

125     do 130 i = 1, nonelec

c         Total overpotential:
          etatot  =  0.0  -  phinew(i)

c         Concentration overpotential:
          cg = x2(i)
c         Set a wall:
          if ( cg . lt . 1.0 ) cg = 1.0
          c  = 1.0
          if ( ixport . ne . 1 ) c = x3(i)
          etac  =  - s1 * dlog ( c )  -  sg * dlog ( cg )

c         Activation overpotenital by difference:
          etas   =   etatot  -  etac

c         Local exchange current density depends on concentration:
          curfar   = - epsilon
          xchcur0  =   speed  *  dabs( curfar )
          xchcur   =   xchcur0 * ( c ** e1 ) * ( cg ** eg )

c         Butler-Volmer kinetics:
          ratio =  dexp ( (1.0 - beta) * etas )  -  dexp (-beta * etas)
          cur   =  xchcur * ratio

c         Express these currents as normal derivatives and pass them
c         back into the calling routine as ''gradnew(i).''
          gradnew(i)  =  - cur / scale

130       continue


c       Evaluate ''extra'', the imbalance in current between
c       the electrodes.

c       We assume that the electrode is flat (z constant)
c       and we require rinner to be zero in these extended-
c       domain problems.
```

```
c       There are this many quadratic elements on the electrode:
        noels = ( nonelec - 1 ) / 2

c       Integrate.

        pi  =  3.14159265
        sumsum  =  0.
        rmax = 0.
        do 30 k = 1, noels

          n1  =  1  +  2 * k  -  2
          n2  =  1  +  2 * k
          n3  =  1  +  2 * k  -  1

          r1  =  coords1(n1,1)
          r2  =  coords1(n2,1)
          r3  =  coords1(n3,1)

          g1 = gradnew(n1)
          g2 = gradnew(n2)
          g3 = gradnew(n3)

c         Find rwall, the radius of the cylindrical insulating
c         boundary in model units.
          rwall = coords1(nonelec,1)

c         The actual integration by gaussian quadrature:
          sum  =  0.
          do 25 j = 1, nogauss

            s = abgauss(j)

            p1  =  0.5 * s * ( s - 1 )
            p2  =  0.5 * s * ( s + 1 )
            p3  =  ( 1.0 - s ) * ( 1.0 + s )

            rjaco = ( r1 + r2 - 2 * r3 ) * s  +  0.5 * (r2 - r1)
            rjaco = dabs ( rjaco )
            r  =  p1 * r1  +  p2 * r2  +  p3 * r3
            g  =  p1 * g1  +  p2 * g2  +  p3 * g3

            sum  =  sum  +  2.0 * pi * wfgauss(j) * g * r * rjaco

25          continue

          sumsum  =  sumsum  +  sum

30        continue
```

```
      top    = - ( epsilon / scale ) * pi * rwall ** 2
      bottom = sumsum

      extra  = top + bottom

      return
      end



c     ******************* SUBROUTINE PHIEVAL *******************
      subroutine phieval

      implicit double precision (a-h,o-z)

      common/set0/nonelec
      common/set61/x1(150)
      common/set62/x2(150)
      common/set18/epsilon,theta,omega,enovrz2
      common/set23/phiefar,phiede,blt,edt,phie(150)

c     This routine evaluates the value of phi at each electrode
c     node by doing a line integral beginning with the surface
c     point farthest from the bubble.  This routine calls the
c     routine ''phistep'' for each element on the electrode.

      noels  = ( nonelec - 1 ) / 2
      phie(noels) = phiefar

      do 80 i = 1, noels

c       Work backwards through electrode elements.
        k = noels + 1 - i

c       Local node numbering within element k:
        n1 = 2 * k  -  1
        n2 = 2 * k  +  1
        n3 = 2 * k

c       Calculate the potential changes from n2 to n3 (step1)
c       and from n3 to n1 ( step2 ) respectively:
        call phistep(k,step1,step2)

c       Step along and calculate the next two potentials.
        phie(n3)  =  phie(n2)  +  step1
        phie(n1)  =  phie(n3)  +  step2
```

```
80      continue

        return
        end



c       ******************* SUBROUTINE PHISTEP ********************
        subroutine phistep(k,step1,step2)

        implicit double precision (a-h,o-z)

        common/set0/nonelec
        common/set61/x1(150)
        common/set62/x2(150)
        common/set18/epsilon,theta,omega,enovrz2

        external eline

c       This routine calculates the step change in phi across a
c       given electrode-surface element from the nodal values of
c       concentration and phi.

        n1  =  2 * k  -  1
        n2  =  2 * k  +  1
        n3  =  2 * k

c       Coefficients of '' d phi / d e  =  Aphi * e  +  Bphi  '':
        Aphi  =  x1(n1)  +  x1(n2)  -  2.0 * x1(n3)
        Bphi  =  0.5 * ( x1(n2) - x1(n1) )

c       Coefficients of '' d con / d e  =  Acon * e  +  Bcon  '':
        Acon  =  x2(n1)  +  x2(n2)  -  2.0 * x2(n3)
        Bcon  =  0.5 * ( x2(n2) - x2(n1) )

c       Coefficients of '' con  =  a  +  b * e  +  c * e * e ''
        a  =  x2(n3)
        b  =  0.5 * (  x2(n2)  -  x2(n1)  )
        c  =  0.5 * (  x2(n1)  +  x2(n2)  -  2.0 * x2(n3)  )

        g  =  Aphi  -  ( theta + enovrz2 ) * Acon
        h  =  Bphi  -  ( theta + enovrz2 ) * Bcon

c       Calculate the potential change from n2 to n3 (step1):
        sign = 1.0
        step1  =  gquad(eline,nogauss,wfgauss,abgauss,idum,idum,
     u                dum1,dum2,sign,a,b,c,g,h)
```

```
c     Calculate the potential change from n3 to n1 (step2):
      sign = -1.0
      step2  =  gquad(eline,nogauss,wfgauss,abgauss,idum1,idum2,
     u              dum1,dum2,sign,a,b,c,g,h)

      return
      end



c     ******************* FUNCTION ELINE *************************
      function eline(idum1,idum2,dum1,dum2,sign,a,b,c,g,h,e)

      implicit double precision (a-h,o-z)

      s  =  sign * ( e + 1.0 ) / 2.0
      eline  =  - ( g * s  +  h ) / ( a  +  b * s  +  c * s * s )
c     We halve eline since the gquad routine stretches it
c     from -1 to 1.
      eline  =  0.5  *  eline

      return
      end



c     ************** SUBROUTINE OUTPUT ******************
      subroutine output

      implicit double precision (a-h,o-z)

c     This routine prints out the answer.
      common/set0/nonelec
      common/set1/nonods1,nonods2
      common/set2/coords1(150,2),coords2(150,2)
      common/set3/iaxism,rinner
      common/set61/x1(150)
      common/set62/x2(150)
      common/set63/x3(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set73/bc3(150)
      common/set81/keybc1(150),keyode1(150)
      common/set82/keybc2(150),keyode2(150)
      common/set83/keybc3(150),keyode3(150)
      common/set10/tol,param,paramnr,nctrmax,nmaxnr,nctr,nctrnr
```

```
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set18/epsilon,theta,omega,enovrz2
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set21/ixport,ifast
      common/set23/phiefar,phiede,blt,edt

c     Print out answers for phi.

      write (2,910)
910   format(/'heres the vector full of answers')

      write (2,920)
920   format(/'node',1x,'k1',2x,'i phi',3x,'phi',4x,
     u  'r or y',2x,'x or z')

c     Rearrange contents of vectors.
      do 930 i = 1, nonods1

c       Put all phi's in vector ``bc1'', all gradients in vector
c       ``x1.''
        if (keybc1(i).eq.1) go to 925
        hold   =  x1(i)
        x1(i)  =  bc1(i)
        bc1(i) =  hold

c       Convert all gradients to current densities normalized to
c       background current density.
925     x1(i) = - x1(i) * scale / epsilon

        write (2,940) i, keybc1(i), x1(i), bc1(i),
     u    coords1(i,1), coords1(i,2)
930     continue

940   format(i3,i2,1x,g14.8,1x,g14.8,2x,
     u  f8.1,1x,f8.1)


c     Print out the beef on the concentrations.

      write (2,9201)
9201  format(/'node',1x,'k1',2x,'i c',3x,'c',4x,
     u  'i cg',4x,'cg',4x,'r or y',2x,'x or z')

c     Rearrange contents of vectors.
      do 9301 i = 1, nonods2

c       Put all c's in vector ``bc2'', all gradients in
```

```
c       vector ``x2.''
        if (keybc2(i).eq.1) go to 9251
        hold   =  x2(i)
        x2(i)  =  bc2(i)
        bc2(i) =  hold

c       Put all cg's in ``bc3'' and all gradients in ``x3.''
9251    if (keybc3(i).eq.1) go to 9261
        hold   =  x3(i)
        x3(i)  =  bc3(i)
        bc3(i) =  hold

c       Convert all gradients to current densities normalized
c       to background current density.
9261    x2(i) = - x2(i) * scale * omega / epsilon
        x3(i) = - x3(i) * scale * theta / epsilon

        write (2,9401) i, keybc2(i), x2(i), bc2(i), x3(i), bc3(i),
     u    coords2(i,1), coords2(i,2)
9301     continue

9401   format(i3,i2,1x,g14.8,1x,g14.8,2x,g10.4,1x,g10.4,
     u  1x,f8.1,1x,f8.1)


      write (2,950) nctr, nctrnr
950   format(/'this many iterations on the relaxed
     $    substitution'/i5/ 'and this many on the
     $    newton-raphson'/i5/)


c     Calculate the dimensionless increment in resistance
c     due to the presence of the attached bubble.
      call delvolt(delv,delr,spacing,delrnrm)

      write (2,954) delv
954   format(/'dimensionless voltage increment'/g12.6)

      write (2,955) delr
955   format(/'dimensionless resistance increment'/g12.6)

      write (2,956) spacing
956   format(/'interbubble spacing in diameters'/f9.5)

      write (2,957) delrnrm
957   format(/'delrnrm = delr * spacing * spacing'/g12.6)
```

```
c     Create a special file with essential data for plotting idist.

      write (4,961)
961   format('title1: descriptive information'/)
      write (4,962)
962   format(/'title2:'/)
      write (4,963)
963   format(/'title3:'/)
      write (4,964)
964   format(/'title4:'/)
      write (4,965)
965   format(/'title5:'/)
      write (4,966)
966   format(/'title6:'/)
      write (4,967)
967   format(/'title7:'/)
      write (4,968)
968   format(/'title8:'/)

c     Pass these default values if only to mark where the
c     number goes.
      radmax = 2.0
      ilog = 0
      curmax = 3.0
      curmin = 0.0
      write (4,969) radmax, ilog, curmax, curmin
969   format(/'radmax',2x,'ilog',5x,'curmax',8x,'curmin'/
     $   2x,f5.2,7x,i1,5x,e8.2,5x,e8.2)

      noaxes = 0
      ifont = 1
      write (4,970) nonelec, scale, noaxes, blt, edt, ifont
970   format(/'nonelec',7x,'scale',7x,'noaxes',
     $   7x,'blt',7x,'edt',6x,'ifont'/2x,i4,7x,f8.2,7x,i1,7x,
     $   f7.3,7x,f7.3,5x,i2)

      write (4,980)
980   format(/'node',10x,'radius',10x,'cur')

      write (4,990) (i, coords1(i,1), x1(i), i=1,nonelec)
990   format(i4,10x,f8.1,10x,f10.4)

      write (4,954) delv
      write (4,955) delr
      write (4,956) spacing
      write (4,957) delrnrm

c     Calculate the Wagner number of this run, ignoring
c     concentration effects.
```

```
      ratio  =  - epsilon / ( dabs(epsilon) * speed )
      call bvkin(ratio,etasave,beta)
      an = ( 1.0 - beta ) * dexp( (1.0-beta) * etasave )
      ct =         beta   * dexp(     -beta  * etasave )
      wagner  =  dabs ( 1.0 / ( speed * epsilon * ( an + ct ) ) )

      write (4,995) wagner
995   format(/'wagner number'/g10.4)

      write (4,996) epsilon, speed, beta
996   format(/'epsilon or -delta'/g10.4//'speed or abs
     $   J/delta'/g10.4//'beta or alphac/sum'/g10.4)

      return
      end



c     ***************** SUBROUTINE DELVOLT *****************
c     This routine calculates the dimensionless increment in
c     voltage ''delv'' due to the presence of the bubble at
c     the surface.

c     We also calculate the dimensionless resistance increment
c     ''delr.'' To get the actual resistance increment due to
c     a regular, hexagonal array of such attached bubbles, simply
c     multiply by the bubble diameter and divide by the electrolyte
c     conductivity.

c     We also return a value ''delrnrm'':
c                 delrnrm == delr * s * s
c     where s is the interbubble spacing of the hexagonal array in
c     bubble diameters. This value will not depend as strongly as
c     ''delr'' does upon the interbubble spacing.

c     Notice that ''delr'' will be positive if the bubbles
c     increase the overall resistance, and vice versa (the case of
c     current enhancement by attached bubbles).

      subroutine delvolt(delv,delr,spacing,delrnrm)

      implicit double precision (a-h,o-z)

      common/set1/nonods1,nonods2
      common/set2/coords1(150,2),coords2(150,2)
      common/set3/iaxism,rinner
      common/set61/x1(150)
      common/set71/bc1(150)
```

```
      common/set81/keybc1(150),keyode1(150)
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set18/epsilon,theta,omega,enovrz2
      common/set21/ixport,ifast
      common/set23/phiefar,phiede,blt,edt

c     We assume that the extended-domain edge is flat
c     (z constant) and we require rinner to be zero in
c     these extended-domain problems.

c     Count the number of nodes ''nonede'' on the extended-domain
c     edge (ede) and keep track of the node number of the first
c     such node ''node1.''
      nonede = 0
      do 20 i = 1, nonods1
        key = keyode1(i)
        if ( key . ne . 2 ) go to 20
        nonede = nonede + 1
        if ( nonede . eq . 1 ) node1 = i
20      continue

c     There are this many quadratic elements on the e.d.e.:
      noels = ( nonede - 1 ) / 2

c     Integrate.
      sumsum = 0.
      rmax = 0.
      do 30 k = 1, noels

        n1 = node1 + 2 * k - 2
        n2 = node1 + 2 * k
        n3 = node1 + 2 * k - 1

        r1 = coords1(n1,1)
        r2 = coords1(n2,1)
        r3 = coords1(n3,1)

        u1 = bc1(n1)
        u2 = bc1(n2)
        u3 = bc1(n3)

c       Find rwall, the radius of the cylindrical insulating boundary
c       in model units.
        if ( r1 . gt . rwall ) rwall = r1
        if ( r3 . gt . rwall ) rwall = r3

c       The actual integration by gaussian quadrature:
        sum = 0.
```

```
      do 25 j = 1, nogauss

        s = abgauss(j)

        p1  =  0.5 * s * ( s - 1 )
        p2  =  0.5 * s * ( s + 1 )
        p3  =  ( 1.0 - s ) * ( 1.0 + s )

        rjaco = ( r1 + r2 - 2 * r3 ) * s  +  0.5 * (r2 - r1)
        rjaco = dabs ( rjaco )
        r  =  p1 * r1  +  p2 * r2  +  p3 * r3
        u  =  p1 * u1  +  p2 * u2  +  p3 * u3

        sum  =  sum  +  wfgauss(j) * ( u - phiede ) * r * rjaco

25      continue

      sumsum  =  sumsum  +  sum

30    continue

    delv     =   2.0  *  sumsum  /  ( rwall * rwall )
    delr     =   delv / epsilon

    pi  =  3.14159265
    afactor  =  dsqrt ( 2.0 * sqrt ( 3.0 ) / pi )
    spacing  =  rwall / ( scale * afactor )

    delrnrm  =  delr * spacing * spacing

    return
    end



c     ********************** SUBROUTINE BVKIN ************************
      subroutine bvkin(ratio,etas,beta)

      implicit double precision (a-h,o-z)

c     This routine solves for surface overpotential in the Butler-
c     Volmer kinetic expression.  ''ratio'' is the ratio of current
c     density to the concentration-dependent exchange-current density.
c     ''beta'' and ''en'' are from newman.  ''etas'' has been made
c     dimensionless by RT/F.

c     The Newton-Raphson algorythm:
```

```
      tol = 1.0e-15
      nctrmax = 100
      nctr = 0

      if (ratio.ge.1.0e+4)   etas =  dlog( ratio) / (1.0-beta)
      if (ratio.le.-1.0e+4)  etas = -dlog(-ratio) /    beta

10    nctr = nctr + 1

      anod = dexp (  ( 1.0 - beta ) * etas )
      cath = dexp (  -    beta      * etas )
      f  =  anod  -  cath  -  ratio

      fprime  =  ( 1.0 - beta ) * anod  +  beta * cath

      deletas  =  -  f  /  fprime

c     Don't let etas change too much between iterations.
      dmax =   1.0
      dmin = - 1.0
      if ( deletas . gt . dmax ) deletas = dmax
      if ( deletas . lt . dmin ) deletas = dmin

      etas  =  etas  +  deletas

      if ( dabs(deletas) . lt . tol ) go to 30
      if ( nctr . gt . nctrmax )      go to 20

      go to 10

20    write (2,21) etas
21    format ('too many iterations on bvkin.'/'etas='g8.2)
      stop

30    return
      end
```

Input-Data File

```
This file contains input data for the boundary-element
bubble problem featuring a concentration boundary layer within an
extended domain.
Contact angle of forty degrees.
diam / blt = 2

iprint                          grid plot key ( 0: run;
i         i         iii         i         1: both; 2: plot)
1         1         500         0

iaxism
i
1

nonods1, the number of nodes in the extended domain.
iii
 96

nonods2, the number of nodes in the diffusion boundary layer
iii
 74

nonelec, the number of nodes on electrode surface (dimension of NR)
iii
 27

scale ( the radius of the bubble in model units )
ddddd.d
  500.0

blt    (``boundary-layer thickness,'' in bubble radii)
ddd.ddd
  1.0

edt    (``extended-domain thickness,'' in bubble radii)
ddd.ddd     (The e.d. is the domain for the potential problem.)
  6.0

vertex coordinates of smallest box enclosing the extended domain
 xmin     xmax
ddddd.d ddddd.d
    0.0  1200.0
 ymin     ymax
ddddd.d ddddd.d
    0.0  3600.0

rinner          (axisymmetric offset, nodal-structure units)
```

```
dddddddd.d
      0.0

s1          sg    (stoichiomeric coefficients of active ion and gas)
d.dd        d.dd
1.0         0.5

enovrz2     Not needed in present version.
dd.dd
 1.0

ixport   transport case ( 1: binary,uniform conductivity;  2: dilute
i            binary 3: well-supported, dilute active ion )
1

epsilon     (the dimensionless derivative of phi normal to the
sd.dddEsee   electrode)  (base case is 2.051e-02)
 2.051e-02

theta       (dless concentration gradient/ dless phi gradient)
sd.dddEsee
 1.460e+00

omega       (dless dissolved-gas gradient/ dless phi gradient)
sd.dddEsee   (base case is -4.448e-05)
-4.448e-05

irestrt(1: read from restart) ikeep(1: write into restart) else 0.
i                             i
1                             0

speed      (exchange current density/current density far from bubble)
sd.dddEsee     i < ifast ( which iteration cycle? 1 gfast; 0 gslow )
 5.482E-01     0

beta        (kinetic parameter; this is fixed at 0.5 for ikinexp=3)
d.dd
0.43

tol         (relative hence dimensionless)
sd.dddEsee
 1.000e-10

relaxation factors for substitution and N.R. ( param and paramnr )
d.dddddddd    d.ddddd
0.01          1.0

max number of iterations for subst. and N.R. ( nctrmax and nmaxnr )
iiii        iiii
```

100 40

Extended-domain coords & keys.

Node	r or y dddddd.d	x or z dddddd.d	bc ii	phi ode ii	loc ii
1	321.4	0.0	0	1	1
2	336.0	0.0	0	1	0
3	350.0	0.0	0	1	1
4	365.0	0.0	0	1	0
5	380.0	0.0	0	1	1
6	395.0	0.0	0	1	0
7	410.0	0.0	0	1	1
8	427.0	0.0	0	1	0
9	445.0	0.0	0	1	1
10	465.0	0.0	0	1	0
11	485.0	0.0	0	1	1
12	505.0	0.0	0	1	0
13	525.0	0.0	0	1	1
14	550.0	0.0	0	1	0
15	575.0	0.0	0	1	1
16	600.0	0.0	0	1	0
17	625.0	0.0	0	1	1
18	650.0	0.0	0	1	0
19	675.0	0.0	0	1	1
20	700.0	0.0	0	1	0
21	750.0	0.0	0	1	1
22	800.0	0.0	0	1	0
23	850.0	0.0	0	1	1
24	900.0	0.0	0	1	0
25	950.0	0.0	0	1	1
26	1000.0	0.0	0	1	0
27	1050.1	0.0	0	1	1
28	1050.1	0.0	0	0	1
29	1050.1	50.0	0	0	0
30	1050.1	100.0	0	0	1
31	1050.1	150.0	0	0	0
32	1050.1	200.0	0	0	1
33	1050.1	250.0	0	0	0
34	1050.1	300.0	0	0	1
35	1050.1	400.0	0	0	0
36	1050.1	500.0	0	0	1
37	1050.1	600.0	0	0	0
38	1050.1	700.0	0	0	1
39	1050.1	800.0	0	0	0
40	1050.1	1000.0	0	0	1
41	1050.1	1250.0	0	0	0
42	1050.1	1500.0	0	0	1
43	1050.1	1750.0	0	0	0
44	1050.1	2000.0	0	0	1

45	1050.1	2250.0	0	0	0
46	1050.1	2500.0	0	0	1
47	1050.1	2750.0	0	0	0
48	1050.1	3000.0	0	0	1
49	1050.1	3000.0	0	2	1
50	750.0	3000.0	0	2	0
51	500.0	3000.0	0	2	1
52	250.0	3000.0	0	2	0
53	0.0	3000.0	0	2	1
54	0.0	883.0	0	0	1
55	73.4	877.6	0	0	0
56	122.3	867.8	0	0	1
57	170.0	853.2	0	0	0
58	216.0	833.9	0	0	1
59	259.9	810.2	0	0	0
60	280.9	796.7	0	0	1
61	301.2	782.1	0	0	0
62	320.7	766.6	0	0	1
63	339.5	750.1	0	0	0
64	357.4	732.7	0	0	1
65	374.4	714.5	0	0	0
66	390.4	695.4	0	0	1
67	405.5	675.5	0	0	0
68	419.6	654.9	0	0	1
69	432.7	633.6	0	0	0
70	444.6	611.7	0	0	1
71	455.5	589.3	0	0	0
72	465.2	566.3	0	0	1
73	475.7	533.2	0	0	0
74	486.1	500.0	0	0	1
75	491.9	472.8	0	0	0
76	496.1	445.7	0	0	1
77	498.6	420.9	0	0	0
78	499.8	395.9	0	0	1
79	499.9	371.0	0	0	0
80	498.6	346.1	0	0	1
81	496.2	321.2	0	0	0
82	492.5	296.5	0	0	1
83	487.5	272.1	0	0	0
84	481.4	247.9	0	0	1
85	474.1	224.0	0	0	0
86	465.5	200.6	0	0	1
87	455.8	177.6	0	0	0
88	445.0	155.1	0	0	1
89	433.1	133.2	0	0	0
90	420.1	111.9	0	0	1
91	406.0	91.3	0	0	0
92	391.0	71.4	0	0	1
93	374.9	52.2	0	0	0

94	358.0	33.9	0	0	1
95	340.1	16.5	0	0	0
96	321.4	0.0	0	0	1

Boundary-layer coords & keys.			gas conc			ion conc	
Node	r or y	x or z	bc	ode	loc	bc	ode
	dddddd.d	dddddd.d	ii	ii	ii	ii	ii
1	321.4	0.0	0	1	1	0	1
2	336.0	0.0	0	1	0	0	1
3	350.0	0.0	0	1	1	0	1
4	365.0	0.0	0	1	0	0	1
5	380.0	0.0	0	1	1	0	1
6	395.0	0.0	0	1	0	0	1
7	410.0	0.0	0	1	1	0	1
8	427.0	0.0	0	1	0	0	1
9	445.0	0.0	0	1	1	0	1
10	465.0	0.0	0	1	0	0	1
11	485.0	0.0	0	1	1	0	1
12	505.0	0.0	0	1	0	0	1
13	525.0	0.0	0	1	1	0	1
14	550.0	0.0	0	1	0	0	1
15	575.0	0.0	0	1	1	0	1
16	600.0	0.0	0	1	0	0	1
17	625.0	0.0	0	1	1	0	1
18	650.0	0.0	0	1	0	0	1
19	675.0	0.0	0	1	1	0	1
20	700.0	0.0	0	1	0	0	1
21	750.0	0.0	0	1	1	0	1
22	800.0	0.0	0	1	0	0	1
23	850.0	0.0	0	1	1	0	1
24	900.0	0.0	0	1	0	0	1
25	950.0	0.0	0	1	1	0	1
26	1000.0	0.0	0	1	0	0	1
27	1050.1	0.0	0	1	1	0	1
28	1050.1	0.0	0	0	1	0	0
29	1050.1	50.0	0	0	0	0	0
30	1050.1	100.0	0	0	1	0	0
31	1050.1	150.0	0	0	0	0	0
32	1050.1	200.0	0	0	1	0	0
33	1050.1	250.0	0	0	0	0	0
34	1050.1	300.0	0	0	1	0	0
35	1050.1	350.0	0	0	0	0	0
36	1050.1	400.0	0	0	1	0	0
37	1050.1	450.0	0	0	0	0	0
38	1050.1	500.0	0	0	1	0	0
39	1050.1	500.0	1	2	1	1	2
40	1000.0	500.0	1	2	0	1	2
41	950.0	500.0	1	2	1	1	2
42	900.0	500.0	1	2	0	1	2

43	850.0	500.0	1	2	1	1	2
44	800.0	500.0	1	2	0	1	2
45	750.0	500.0	1	2	1	1	2
46	700.0	500.0	1	2	0	1	2
47	650.0	500.0	1	2	1	1	2
48	600.0	500.0	1	2	0	1	2
49	550.0	500.0	1	2	1	1	2
50	515.0	500.0	1	2	0	1	2
51	486.1	500.0	1	2	1	1	2
52	486.1	500.0	0	0	1	0	0
53	491.9	472.8	1	2	0	0	0
54	496.1	445.7	1	2	1	0	0
55	498.6	420.9	1	2	0	0	0
56	499.8	395.9	1	2	1	0	0
57	499.9	371.0	1	2	0	0	0
58	498.6	346.1	1	2	1	0	0
59	496.2	321.2	1	2	0	0	0
60	492.5	296.5	1	2	1	0	0
61	487.5	272.1	1	2	0	0	0
62	481.4	247.9	1	2	1	0	0
63	474.1	224.0	1	2	0	0	0
64	465.5	200.6	1	2	1	0	0
65	455.8	177.6	1	2	0	0	0
66	445.0	155.1	1	2	1	0	0
67	433.1	133.2	1	2	0	0	0
68	420.1	111.9	1	2	1	0	0
69	406.0	91.3	1	2	0	0	0
70	391.0	71.4	1	2	1	0	0
71	374.9	52.2	1	2	0	0	0
72	358.0	33.9	1	2	1	0	0
73	340.1	16.5	1	2	0	0	0
74	321.4	0.0	1	2	1	0	0

Appendix A-3

Boundary-Element Code for Leveling Model

The following Fortran code was used in the calculations of Chapter 3. A sample input-data file is given at the end of the program listing. Important variables are defined in subroutine 'input' or as they appear in the code. Subroutines 'matrix,' 'makeb,' 'gauss1,' 'gauss2,' 'rsread' and 'rswrite' as well as functions 'gquad,' 'ei1,' 'ei2,' 'ggrand' and 'hgrand' are listed in Appendix A-1.

```
      program belev

c     This program is intended to solve for the current distribution
c     at an irregularly shaped electrode, and to simulate the evolution
c     of the electrode shape with time.

c     Laplace's equation is solved for two variables:
c         subscript              variable
c           1                 potential, phi
c           2                 concentration of leveling agent, con.
c     We solve for the concentration over the diffusion boundary
c     layer (dbl) or ``domain 2,'' while phi is solved over a larger
c   region, ``domain 1'', comprising the dbl, with matching conditions at
c     the electrode.

c     The overpotential acting at the electrode surface
c     is a linearized function of both local current density
c     and local flux of a leveling agent.

c     More comments to be elaborately furnished here later.

c     This program is general to axisymmetric and planar problems.

c     This is a boundary-element program using quadratic elements.
c     Be sure that, for both domains, the node numbering is done
c     counterclockwise beginning with the first electrode node.
c     Also, the node arrangement on the working electrode must be
c     identical for the two domains.

c     Vectors are always dimensionless;  scalars not necessarily.

      implicit double precision (a-h,o-z)

      common/nr1/ surf, delsurf
      common/piv1/ ipivot1(150), jpivot1(150)
      common/piv2/ ipivot2(150), jpivot2(150)
      common/set0/nonelec,nonelc2
      common/set1/nonods1,nonods2
      common/set2/coords1(150,2),coords2(150,2)
      common/set2a/keyloc1(150),keyloc2(150)
      common/set3/iaxism,rinner
      common/set41/a1(150,150)
      common/set42/a2(150,150)
      common/set51/c1(150,150)
      common/set52/c2(150,150)
      common/set61/x1(150)
      common/set62/x2(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
```

```
      common/set81/keybc1(150),keyode1(150)
      common/set82/keybc2(150),keyode2(150)
      common/set10/tol,param,paramnr,nctrmax,nmaxnr,nctr,nctrnr
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set13/iflag
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set18a/vchar,curchar,flxchar,conchar
      common/set18b/etabar,curbar,flxbar
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set22/irestrt,ikeep
      common/set24/igrid,xmin,xmax,ymin,ymax
      common/set25/tosca,nosteps,nstep

      dimension b1(150), b2(150)
      dimension surf(150), surfnew(150), delsurf(150)
      dimension space1(150), space2(150)

c     Begin by reading in all the data for the problem.
c     Call the input subroutine.

      call input

c     If requested, create a plot of the problem geometry.
c     Act according to the key ''igrid'':
c      igrid=0       run but no plot
c      igrid=1       run and plot
c      igrid=2       plot only ( no run )
      if ( igrid . lt . 1 ) go to 10
      call grid(nonods1,coords1,keyode1,keyloc1)
      call grid(nonods2,coords2,keyode2,keyloc2)
      if ( igrid . eq . 2 ) stop

c     See if this run is a restart ( irestrt = 1 ).  If so,
c     skip the costly ''matrix'' and ''gauss1'' routines and merely
c     read in their products from the restart file, for003.dat.

10    if ( irestrt . eq . 1 ) then
        call rsread
        go to 103
        endif

c     Call subroutine ''matrix'' to set up both matrix problems,
c     ''G q = H u'' and rearrange to give ''G q = H u'' for each.

c     Initialize some things:
```

```
90      nmin1d1 = 0
        nmin2d1 = 0
        nmin3d1 = 0
        nmin4d1 = 0
        nmin1d2 = 0
        nmin2d2 = 0
        nmin3d2 = 0
        nmin4d2 = 0

        nmax1d1 = 0
        nmax2d1 = 0
        nmax3d1 = 0
        nmax4d1 = 0
        nmax1d2 = 0
        nmax2d2 = 0
        nmax3d2 = 0
        nmax4d2 = 0

        gap1 = 0.
        gap2 = 0.

        do 95 k = 1, nonods
          space1(k) = 0.
          space2(k) = 0.
95        continue

c       Call the ''premove'' routine to set values that will be used
c       in each time step.
        call premove(coords1,keyode1,gap1,nmin1d1,nmax1d1,nmin2d1,
     $    nmax2d1,nmin3d1,nmax3d1,nmin4d1,nmax4d1,nonods1,space1)
        call premove(coords2,keyode2,gap2,nmin1d2,nmax1d2,nmin2d2,
     $    nmax2d2,nmin3d2,nmax3d2,nmin4d2,nmax4d2,nonods2,space2)

        nstep = 0
        call profile

        ***************** START OF TIME-STEP CYCLE *****************
100     nstep = nstep + 1
        call matrix(nonods1,coords1,keyloc1,a1,c1,keybc1,keyode1)
        call matrix(nonods2,coords2,keyloc2,a2,c2,keybc2,keyode2)

c       Send off both A matrices to be forward reduced in subroutine
c       ''Gauss1.'' The reduced matrices are retained in their original
c       storage locations along with a record of row manipulations.
c       Vectors ''ipivot'' and ''jpivot'' are formed for each problem,
c       as a record of row and column exchanges carried out.

        call gauss1(nonods1,a1,ipivot1,jpivot1,iflag1)
        call gauss1(nonods2,a2,ipivot2,jpivot2,iflag2)
```

```
c       If any matrix is singular, stop everything.
        if ( iflag1 . eq . 1 . or . iflag2 . eq . 1 ) then
          write (2,102) iflag1, iflag2
102       format('/singular element matrix')
          stop
          endif

c       If requested, write the contents of pivot vectors and a and
c       c matrices into a restart file, for008.dat.
103     if ( ikeep . eq . 1 ) then
          call rswrite
          endif

c       Set concentration of agent to zero at the electrode surface.
        do 104 k = 1, nonelc2
          bc2(k) = 0.
104       continue

c       Set concentration of agent to unity at the boundary-layer edge.
        do 105 k = 1, nonods2
          key = keyode2(k)
          if ( key . ne . 2 ) go to 105
          bc2(k) = 1.0
105       continue

c       Solve the agent concentration problem once and for all (for
c       this time step.)

c       Apply the BCs; make vector b2.
        call makeb(nonods2,c2,bc2,b2)

c       Solve for concentration gradient, x2.
        neumann = 0
        call gauss2(nonods2,a2,x2,b2,ipivot2,jpivot2,neumann)

c       Now start into the field-problem iteration cycle.

c       Initialize surf to phi at the electrode surface.
        do 110 i = 1, nonelec
          surf(i) = bc1(i)
110       continue

c       Initialize parameters for iteration, convergence.
c       divergence tests.
        nctr    = 0
        nctrnr  = 0
        ndiv    = 0
        ndivmax = 50
        tmaxdiv = 1000.
```

```
      frac    =  param
      tmax    =  100.
      tolnr   =  tol * 2.
      iterkey =  1

c     **************** Start of Iteration *********************

c     See if we're at the point of changeover to N.R.

120   if ( nctrnr . eq . 0 ) then
        if ( nctr.eq.nctrmax . or . tmax.le.tolnr ) then
c         At the point of changeover to N.R. reset
c         some parameters.
          iterkey = 2
          ndivmax = 2
          ndiv = 0
          frac = paramnr
          endif
        endif

c     Perform either of two types of iteration, ``1'' or ``2'':

c     1) ********* Substitution Iteration *********
      if ( iterkey . eq . 1 ) then
        nctr  = nctr  + 1
        call field(surf,surfnew)
        endif

c     2) ********* Newton Raphson Iteration *******
      if ( iterkey . eq . 2 ) then
        nctrnr = nctrnr + 1
        frac = paramnr
        if ( nmaxnr . eq . 0 ) then
          write (2,126)
126       format(/'You exceeded nctrmax substitution iterations;'/
     $      'no newton iterations called for.')
          stop
          endif
       Call subroutine ``nr'' to operate on the vector ``surf''
       returning a vector of corrections ``delsurf.''
        call nr(field)
        do 128 i = 1, nonelec
          surfnew(i) = surf(i) + delsurf(i)
128       continue
        endif


c     ************* Convergence Test, Update ``surf'' *************
```

```
c     For the convergence criterion, find the biggest change in surf
c     on the electrode.  Also, here is where you make the relaxed
c     substitution to form new guess surf for the next iteration.
      tmaxold = tmax
      tmax    = 0.
c     Find the largest absolute change in surf.
      do 130 i = 1, nonelec
        t = dabs( surfnew(i) - surf(i) )
        if ( t . gt . tmax ) tmax = t
        surf(i) = frac * surfnew(i) + ( 1.0 - frac ) * surf(i)
c       Impose some walls:
       if ( surf(i) . lt . wall ) surf(i) = wall
130     continue

c     Normalize tmax.
      tmax = dabs( tmax )

c     Do the following only on the first and last time
c     steps (so as to save file space):
      if ( nstep . eq . 1 . or . nstep . eq . nosteps ) then

c       As an diagnostic, print out tmax for each iteration.
c       Heading first:
        if (nctr.eq.1) then
          write (2,140)
140       format(/2x,'nctr',3x,'nctrnr',3x,'tmax')
          endif
c       Then the values:
        write (2,142) nctr, nctrnr, tmax
142     format(2x,i4,4x,i4,4x,g10.4)

        endif

c     If you have converged, brag about it and then go about
c     moving the boundary and jumping to the start of the next
c     time step:
      if ( tmax . le . tol ) then

        if ( nstep . eq . 1 . or . nstep . eq . nosteps ) then
          write (2,148)
148       format(/'Of course it converged!'/)
          call output
          endif

        do 148 i = 1, nonelec
          x1(i) = 200.0
148       continue

c       Move the boundary:
```

```
      call move1(coords1,gap1,nmin1d1,nmax1d1,nmin2d1,
     $   nmax2d1,nmin3d1,nmax3d1,nmin4d1,nmax4d1,space1)
      call move2(coords1,coords2,gap2,nmin1d2,nmax1d2,nmin2d2,
     $   nmax2d2,nmin3d2,nmax3d2,nmin4d2,nmax4d2,space2)

c     Record the electrode-node coordinates:
      call profile

c     Stop after the last time step:
      if ( nstep . ge . nosteps ) then
        write (2,150) nosteps
150     format(/'Finished',1x,i5,1x,'time steps'/)
        stop
        endif

c     ************** END OF TIME-STEP CYCLE ****************
      go to 100

c      endif


c    If you've exceeded the max allowed iterations, say so
c    and quit.
     if ( nctrnr . gt . nmaxnr  ) then
       write (2,160)
160    format ('You exceeded nmaxnr iterations.')
       call output
       stop
       endif

c    Test for divergence two different ways:

c    1) If tmax exceeds tmaxdiv then bail out:
     if ( tmax . ge . tmaxdiv ) then
       write (2,170)
170    format( /'Divergence:  tmax exceeds tmaxdiv.'/ )
       call output
       stop
       endif

c    2) Keep a running tally of the pattern of convergence:
c    If tmax goes UP ndivmax times more than it goes DOWN over any
c    interval, then announce this and quit.
     if ( tmax . ge . tmaxold ) ndiv = ndiv + 1
     if ( tmax . lt . tmaxold ) ndiv = ndiv - 1
     if ( ndiv . le . 0       ) ndiv = 0
     if ( ndiv . gt . ndivmax ) then
       write (2,180)
180    format( /'Divergence:  ndiv exceeds ndivmax.'/ )
```

```
        call output
        stop
        endif

c       If you have neither converged nor shown signs of
c       divergence, do another iteration (either newton-
c       raphson or relaxed substitution).
        go to 120

        end



c       ******************* SUBROUTINE INPUT ***********************
        subroutine input

        implicit double precision (a-h,o-z)
        double precision mark

c       This subroutine reads in all the input data.
        common/set0/nonelec,nonelc2
        common/set1/nonods1,nonods2
        common/set2/coords1(150,2),coords2(150,2)
        common/set2a/keyloc1(150),keyloc2(150)
        common/set3/iaxism,rinner
        common/set71/bc1(150)
        common/set72/bc2(150)
        common/set81/keybc1(150),keyode1(150)
        common/set82/keybc2(150),keyode2(150)
        common/set10/tol,param,paramnr,nctrmax,nmaxnr,nctr,nctrnr
        common/set11/nogauss,wfgauss(12),abgauss(12)
        common/set12/scale
        common/set13/iflag
        common/set16/nctrsp,ispotpr,intspot
        common/set17/tmax
        common/set18a/vchar,curchar,flxchar,conchar
        common/set18b/etabar,curbar,flxbar
        common/set18c/noflx,flxi(20),nocur,curj(20),etaij(20,20)
        common/set19/speed,beta
        common/set20/s1,sg,e1,eg
        common/set22/irestrt,ikeep
        common/set24/igrid,xmin,xmax,ymin,ymax
        common/set25/tosca,nosteps,nstep

c       Read in all the input data.
```

```
      read (1,301)
301   format(////)

c     Read iprint (print all data unless iprint equals zero),
c     ispotpr (values are spot-printed every 'intspot' iterations if
c     ispotpr = 1), and intspot.
c     Also read igrid, key for plotting grid.
c          igrid=0        run, no plot
c          igrid=1        run and plot
c          igrid=2        plot, no run
      read (1,302) iprint, ispotpr, intspot, igrid
302   format(///i1,9x,i1,9x,i3,10x,i1)

c     Read iaxism.  If this reads zero, the problem is not
c     axisymmetric.  A value of one or anything else indicates
c     axisymmetry.
      read (1,303) iaxism
303   format(///i1)

c     Read nonods1, the number of nodes in the nodal structure for
c     the large domain ( over which phi is solved ).
      read (1,304) nonods1
304   format(///i3)

c     Read nonods2, the number of nodes in the nodal structure for
c     the diffusion boundary layer  ( the small domain over which
c     c and cg are solved ).
      read (1,305) nonods2
305   format(///i3)

c     Read nonelec, the number of nodes on the electrode surface.
      read (1,306) nonelec
306   format(///i3)

c     Read nonelc2, the number of nodes on the electrode surface.
c     subtended by the concentration-problem domain.  This may
c     be specified as smaller than nonelec if, beyond a certain
c     point, there is no variation in agent flux expected.
      read (1,3065) nonelc2
3065  format(///i3)

c     Read in the scale factor for the model representation of the
c     domaim.  SCALE is the characteristic length of the problem
c     expressed in model length units.  Generally, we take
c     scale > 1 to get the coordinates in round numbers.
      read (1,307) scale
307   format(///f7.1)

c     Read vertices of smallest rectangle that encloses the domain
```

```
c       (for graphics use.)
        read (1,309) xmin, xmax
309     format(////f7.1,1x,f7.1)
        read (1,310) ymin, ymax
310     format(//f7.1,1x,f7.1)

c       Read in the value of rinner.  This is the distance (in
c       the distance units of the model) from the innermost point
c       of the nodal structure to the centerline.  If the problem
c       is not axisymmetric, we don't need rinner, but we must
c       read a value anyhow.
        read (1,311) rinner
311     format(///f9.1)

c       Read the stoichiometric coefficients of the reacting
c       species, s1 and sg.  These are ''Newmanclature'', p 31.
c       (Not used in the present version)
        read (1,312) s1, sg
312     format(///f4.2,6x,f4.2)

c       Read curbar, the uniform current density far from the
c       electrode, mA/cm2.
        read (1,315) curbar
315     format(///e10.3)

c       Read flxbar, the uniform inhibitor flux that would obtain
c       at a flat electrode under the same hydrodynamic conditions,
c       micromoles/cm2-s. (This is not used in the present version)
        read (1,316) flxbar
316     format(///e10.3)

c       ''etabar'', the uniform overpotenital that would obtain
c       at the surface of a flat electrode under the same hydrodynamic
c       conditions, mV.  (This is not used in the present version)
        read (1,3161) etabar
3161    format(///e10.3)

c       ''noflx'', the number of different ''flx'' values used
c       to construct the polarization expression.  (Number of
c       rows in matrix ''curij''.)
        read (1,3162) noflx
3162    format (///i2///)

c       ''flxi'', the vector of ''flx'' values used to construct
c       the polarization expression.  ''flx'' is leveling-agent
c       flux in micromoles/cm2-s, negative.
        do 3164 i = 1, noflx
          read (1,3163) flxi(i)
3163      format(6x,e10.3)
```

```
3164    continue

c       ''nocur'', the number of different ''cur'' values used
c       to construct the polarization expression.  (Number of
c       columns in matrix ''etaij''.)
        read (1,3165) nocur
3165    format (///i2///)

c       ''curj'', the vector of ''cur'' values used to construct
c       the polarization expression.  ''cur'' is current density
c       in mA/cm2, negative at a cathode.
        do 3167 j = 1, nocur
          read (1,3166) curj(j)
3166      format(6x,e10.3)
3167      continue

c       ''etaij'', matrix of surface overpotentials corresponding
c       to the ''cur-flx'' pairs above.  ''eta'' is surface
c       overpotential in millivolts, negative at a cathode.
        read (1,10)
10      format(/)
        do 50 j = 1, nocur
          read (1,20)
20        format(//)
          do 40 i = 1, noflx
            read (1,30) etaij(i,j)
30          format(6x,e10.3)
40          continue
50        continue

c       charl, characteristic length, centimeters
        read (1,3170) charl
3170    format(///e10.3)

c       vchar, the characteristic voltage ( RT/F ), millivolts
        read (1,3171) vchar
3171    format(///e10.3)

c       conchar, characteristic inhibitor concentration
c       (micromolar)
        read (1,3172) conchar
3172    format(///e10.3)

c       curchar, characteristic current density
c       ( vchar * kappa / charl ) mA/cm2.
        read (1,3173) curchar
3173    format(///e10.3)

c       flxchar, characteristic inhibitor flux
```

```
c       ( conchar * diffusivity / charl ), micromoles/cm2-s
        read (1,3174) flxchar
3174    format(///e10.3)

c       Is this run a restart of an earlier one?
c       ( yes: irestrt = 1;  no: irestrt = 0 )
c       Do we want to save matrix values and pivot keys
c       in a restart file?   ( yes: ikeep = 1;  no: ikeep = 0 )
        read (1,3175) irestrt, ikeep
3175    format(///i1,29x,i1)

c       Read the kinetic symmetry factor beta.
        read (1,319) beta
319     format(///f4.2)

c       Calculate exponents ql and qg for the kinetic expression,
c       according to Newman, p 174.
        if ( sl . gt . zero ) ql = 0.
        if ( sl . lt . zero ) ql = - sl
        if ( sg . gt . zero ) qg = 0.
        if ( sg . lt . zero ) qg = - sg
        el  =  ql  +  beta * sl
        eg  =  qg  +  beta * sg

c       Read the value of the relative convergence tolerance.
        read (1,320) tol
320     format(///e10.3)

c       Read in the convergence parameters: ``param'', the
c       fraction of the new value used in the relaxed substitution,
c       and ``paramnr'' the fraction of the NR correction term used.
        read (1,321) param, paramnr
321     format(///f10.8,4x,f7.5)

c       Read in the maximum number of iterations allowed: ``nctrmax''
c       for relaxed substitution, and ``nmaxnr'' for newton raphson.
        read (1,324) nctrmax, nmaxnr
324     format(///i4,7x,i4)

c       Read ``tosca'', the dless ratio of thickness step to
c       current density.
        read (1,3246) tosca
3246    format(///e10.3)

c       Read ``nosteps'', the number of time steps you want to use.
        read (1,3247) nosteps
3247    format(///i4)

c       Read the coordinates of each node in global order.
```

```
c     As a convention, we number counter clockwise beginning with
c     the electrode.  Also read in the keys for the boundary
c     condition type (''keybc'') and the boundary-segment
c     type (''keyode'').

c       First the coordinates of the extended domain:
        read (1,325)
325     format(///)
        read (1,326) (coords1(i,1),coords1(i,2),keybc1(i),
     $    keyode1(i),keyloc1(i),i=1,nonods1)
326     format(8x,f9.1,1x,f9.1,4x,i2,2x,i2,2x,i2,4x)

c       Then the coordinates of the boundary layer:
        read (1,3255)
3255    format(///)
        read (1,3265) (coords2(i,1),coords2(i,2),keybc2(i),
     $    keyode2(i),keyloc2(i),i=1,nonods2)
3265    format(8x,f9.1,1x,f9.1,4x,i2,2x,i2,2x,i2)

c     Initialize the boundary-condition vectors to zero.
      do 330 i = 1, nonods1
        bc1(i) = 0.
330     continue
      do 3305 i = 1, nonods2
        bc2(i) = 0.
3305    continue

c     Fix the the phi derivative at the top of the extended
c     domain at a value corresponding to the average current
c     density (according to ohm's law).  (MINUS curbar,
c     actually, because we're at the anode, so to speak.)
      do 331 i=1,nonods1
        if ( keyode1(i) . eq . 2 ) then
          bc1(i) = - ( - curbar / curchar ) / scale
          endif
331     continue

c     Initialize the phi boundary condition at the electrode.
c     ( eta = phimetal - phisolution  and phimetal is zero )
      do 333 i=1,nonelec
        bc1(i) = - etabar / vchar
333     continue

c     Set dimensionless agent concentration to unity at boundary-
c     layer edge and zero at the working-electrode surface:
      do 335 i=1, nonods2
        if ( keyode2(i) . eq . 2 )   bc2(i) = 1.0
        if ( keyode2(i) . eq . 1 )   bc2(i) = 0.0
335     continue
```

```
c       Six-point gaussian quadrature coefficients:

        nogauss = 6

        abgauss(1) = -0.932469514203152
        abgauss(2) = -0.661209386466265
        abgauss(3) = -0.238619186083197
        abgauss(4) =  0.238619186083197
        abgauss(5) =  0.661209386466265
        abgauss(6) =  0.932469514203152

        wfgauss(1) =  0.171324492379170
        wfgauss(2) =  0.360761573048139
        wfgauss(3) =  0.467913934572691
        wfgauss(4) =  0.467913934572691
        wfgauss(5) =  0.360761573048139
        wfgauss(6) =  0.171324492379170

        return
        end



c       ***************** SUBROUTINE NR ********************
c       This routine performs one iteration of the multivariable
c       Newton-Raphson method to solve the system ``f(x) = 0.''
c       The function ``f(x) is equal to ``g(x) - x'', and
c       ``g(x)'' is evaluated in subroutine ``subr(x,xnew)'' (a
c       dummy name for a subroutine that operates on vector
c       ``x'' and returns ``xnew = g(x)'' ).  Upon convergence,
c       vectors ``x'' and ``xnew'' will be identical.
c       We obtain corrections ``delx'' to vector ``x'' by solving
c       the matrix problem ``FPRIME * delx = - f.''

        subroutine nr(subr)

        implicit double precision (a-h,o-z)

        common/nr1/ x(150), delx(150)
        common/set0/nonelec

        dimension xp(150), xpnew(150)
        dimension fprime(150,150), f(150)
        dimension ipivot4(150), jpivot4(150)

c       Set ``pert'' the value of the perturbation in x used
```

```
c       to evaluate the numerical derivative.

        pert = 0.0000001
c       pert = 0.0001

c     Create the jacobian matrix ''fprime'' in three steps:

c       1) Evaluate function of unperturbed set.
c       This may seem strange but we do this again in step 2 for
c       keeps; the only reason we do it here first is to fix the
c       value of the offset voltage in the gslow routine ( if
c       applicable ).  If this offset voltage is not held fixed
c       while we perturb the xp's, then the jacobian will turn out
c       singular.
        do 5 i = 1, nonelec
          xp(i) = x(i)
5         continue
        call subr(xp,xpnew)

c       2) fill matrix with functional evaluations of perturbed x's.
        do 30 j = 1, nonelec
          do 10 i = 1, nonelec
            xp(i) = x(i)
            if ( i . eq . j ) xp(i) = xp(i) + pert
10          continue
          call subr(xp,xpnew)
          do 20 i = 1, nonelec
            fprime(i,j) = xpnew(i)
20          continue
30        continue

c       3) Evaluate function of unperturbed set.
        do 40 i = 1, nonelec
          xp(i) = x(i)
40        continue
        call subr(xp,xpnew)

c       4) Subtract vector ''xpnew'' from each column of the matrix.
c       Then divide each difference by pert.
c       Lastly, subtract unity from each diagonal element.
        do 70 j = 1, nonelec
          do 60 i = 1, nonelec
            fprime(i,j) = ( fprime(i,j) - xpnew(i) ) / pert
            if ( i . eq . j ) fprime(i,j) = fprime(i,j) - 1.0
60          continue
70        continue

c       Evaluate the vector ''f,''  ( f(x) = g(x) - x ).
c       (Recall that the last evaluation of g was for the
```

```
c       unperturbed vector x.)
c       Actually fill vector f with MINUS the value it should have
c       so that we can consider delx an additive correction.
        do 80 i = 1, nonelec
          f(i) = - ( xpnew(i) - x(i) )
80        continue

c       Solve the system ''FPRIME * delx = f'' using the solvers
c       ''gauss1'' and ''gauss2'' also used for the field problem.

        call gauss1(nonelec,fprime,ipivot4,jpivot4,jacsing)
        call gauss2(nonelec,fprime,delx,f,ipivot4,jpivot4,0)

        return
        end



c       ****************** SUBROUTINE FIELD *********************
c       This routine operates on a set of potential values at the
c       electrode ''phi'' and returns a new set ''phinew'' which
c       is generated by cycling through the boundary-value problem
c       once.

        subroutine field(phi,phinew)

        implicit double precision (a-h,o-z)

        common/piv1/ ipivot1(150), jpivot1(150)
        common/piv2/ ipivot2(150), jpivot2(150)
        common/set0/nonelec,nonelc2
        common/set1/nonods1,nonods2
        common/set2/coords1(150,2),coords2(150,2)
        common/set2a/keyloc1(150),keyloc2(150)
        common/set3/iaxism,rinner
        common/set41/a1(150,150)
        common/set42/a2(150,150)
        common/set51/c1(150,150)
        common/set52/c2(150,150)
        common/set61/x1(150)
        common/set62/x2(150)
        common/set71/bc1(150)
        common/set72/bc2(150)
        common/set81/keybc1(150),keyode1(150)
        common/set82/keybc2(150),keyode2(150)
        common/set10/tol,param,paramnr,nctrmax,nmaxnr,nctr,nctrnr
        common/set11/nogauss,wfgauss(12),abgauss(12)
```

```
      common/set12/scale
      common/set13/iflag
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set18a/vchar,curchar,flxchar,conchar
      common/set18b/etabar,curbar,flxbar
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
      common/set22/irestrt,ikeep
      common/set24/igrid,xmin,xmax,ymin,ymax
      common/set25/tosca,nosteps,nstep

      dimension b1(150), b2(150)
      dimension phi(150), phinew(150)

c     Set bc1(i).
      do 115 i = 1, nonelec
        bc1(i)  =  phi(i)
115     continue

c     Apply the boundary conditions; make vector b1.
      call makeb(nonods1,c1,bc1,b1)

c     Solve the field problem for the nodal derivatives (current
c     distribution) by calling ''gauss2'' to do back substitution.
      call gauss2(nonods1,a1,x1,b1,ipivot1,jpivot1,0)

c     Apply the overpotential condition to calculate a new
c     set of electrode potentials from the nodal derivatives.
      do 130 i = 1, nonelec

c       Activation overpotenital: (We neglect concentration
c       overpotential). We use the ''overpot'' routine to
c       get surface overpotential, ''eta,'' as a function of
c       ''x1'' (proportional to current density) and ''x2''
c       (the flux of inhibitor) at each electrode node.

        cur  =  - x1(i) * scale * curchar

c       On any working-electrode node not comprised by the
c       concentration domain, ''extend'' the concentration
c       solution by simply repeating the value of flux at
c       node ''nonelc2''.
        if ( i . gt . nonelc2 ) then
            i2 = nonelc2
          else
            i2 = i
          endif
```

```
        flx  =  - x2(i2) * scale * flxchar

        call overpot( cur, flx, eta )

c       Calculate new surface values of phi ``phinew''.
        phinew(i) = - eta / vchar

130     continue

      return
      end



c     ****************** SUBROUTINE OVERPOT *********************
c     This routine calculates the local surface overpotential,
c     ``eta'', corresponding to given values of ``cur'', (current
c     density in mA/cm2) and ``flx'' (the flux of inhibitor).

      subroutine overpot(cur,flx,eta)

      implicit double precision(a-h,o-z)
      double precision mark

      common/set18c/noflx,flxi(20),nocur,curj(20),etaij(20,20)
      dimension etai(20), etaj(20)

c     We have a rectangular grid of points ``etaij'' in
c     ``cur - flx'' space (all values in fully dimensional
c     form) among which we do bilinear interpolation in
c     two steps:

c     1)  Create vector ``etaj'' that represents the 'isoflux'
c     line at ``flx'':
      do 20 j = 1, nocur
        do 10 i = 1, noflx
          etai(i) = etaij(i,j)
10        continue
        etaj(j) = trap(flx, noflx, flxi, etai, 1)
20      continue

c     2)  Search along the trapezoidal fit of ``etaj'' for
c     ``cur'' and record the corresponding value of ``eta''.
      eta = trap(cur, nocur, curj, etaj, 2)

      return
      end
```

```
c       ****************** FUNCTION TRAP ************************
c       This interpolates trapezoidally among x-y pairs to return
c       a ''y'' value corresponding to the desired value of ''x''.
        function trap(x, nopairs, xvector, yvector, idim)

        implicit double precision (a-h,o-z)
        dimension xvector(20), yvector(20)

        irange = 0
        nosegs = nopairs - 1
        do 10 i = 1, nosegs

          n1 = i
          n2 = i + 1

          x1 = xvector(n1)
          x2 = xvector(n2)

          frac  =  ( x - x1 ) / ( x2 - x1 )

          if ( frac . ge . 0.0 . and . frac . lt . 1.0 ) then
            irange = 1
            y1 = yvector(n1)
            y2 = yvector(n2)
            y  =  y1  +  frac * ( y2 - y1 )

c           If you are doing linear interpolation in the first
c           direction, (idim=1) then treat ''out-of-range'' points
c           as follows:  if either of the two points is out of
c           range (the code for this is y = 0) then encode the
c           interpolate as out of range as well ( set y = 0 ).
            if ( idim . eq . 1 ) then
              if ( y1 . eq . 0.0 . or . y2 . eq . 0.0 ) y = 0.0
              endif

c           If you are doing linear interpolation in the second
c           dimension ( idim = 2 ) then don't allow any
c           interpolation using an ''out-of-range'' point (the
c           code for out-of-range is y = 0.0 ).
            if ( idim . eq . 2 ) then
              if ( y1 . eq . 0.0 . or . y2 . eq . 0.0 ) then
                write (2,7) x, x1, x2, y1, y2
7               format (/'Interpolating out of range in 2nd dimension'
     $            /'x, x1, x2, y1, y2'
     $            /e10.3,2x,e10.3,2x,e10.3,2x,e10.3,2x,e10.3)
                stop
                endif
              endif
```

```
          endif

10      continue

      if ( irange . eq . 0 ) then
        write (2,20) idim, x, xvector(1), xvector(nopairs)
20      format(/'Trap argument out of range.'
     $    /'idim',4x,'x',5x,'xvector1',5x,'xvectorn'
     $    /i2,2x,e10.3,5x,e10.3,5x,e10.3)
        stop
        endif

      trap = y

      return
      end


c     ************** SUBROUTINE OUTPUT *****************
      subroutine output

      implicit double precision (a-h,o-z)

c     This routine prints out the answer.
      common/set0/nonelec,nonelc2
      common/set1/nonods1,nonods2
      common/set2/coords1(150,2),coords2(150,2)
      common/set2a/keyloc1(150),keyloc2(150)
      common/set3/iaxism,rinner
      common/set61/x1(150)
      common/set62/x2(150)
      common/set71/bc1(150)
      common/set72/bc2(150)
      common/set81/keybc1(150),keyode1(150)
      common/set82/keybc2(150),keyode2(150)
      common/set10/tol,param,paramnr,nctrmax,nmaxnr,nctr,nctrnr
      common/set11/nogauss,wfgauss(12),abgauss(12)
      common/set12/scale
      common/set13/iflag
      common/set16/nctrsp,ispotpr,intspot
      common/set17/tmax
      common/set18a/vchar,curchar,flxchar,conchar
      common/set18b/etabar,curbar,flxbar
      common/set19/speed,beta
      common/set20/s1,sg,e1,eg
```

```
      common/set22/irestrt,ikeep
      common/set24/igrid,xmin,xmax,ymin,ymax
      common/set25/tosca,nosteps,nstep

c     Print out answers for phi.

      write (2,10)
10    format(/'heres the vector full of answers')

      write (2,20)
20    format(/'node',1x,'k1',8x,'cur',10x,'phi',8x,
     u  'r or y',2x,'x or z')

c     Rearrange contents of vectors.
      do 30 i = 1, nonods1

c       Put potentials in ``pt'' and all current
c       densities in ``cd''.  Apply ohm's law, and
c       ``redimensionalize'' the variables.
        if (keybc1(i).eq.0) then
            pt    =    x1(i) * vchar
            cd    = - bc1(i) * scale * curchar
          else
            pt    =    bc1(i) * vchar
            cd    = - x1(i) * scale * curchar
          endif

        write (2,40) i, keybc1(i), cd, pt,
     $    coords1(i,1), coords1(i,2)
30      continue

40    format(i3,i2,1x,g14.8,1x,g14.8,2x,
     $  f8.1,1x,f8.1)


c     Print out the beef on the concentrations.

      write (2,50)
50    format(/'node',1x,'k1',8x,'cur',10x,
     $  'conc',8x,'r or y',2x,'x or z')

c     Rearrange contents of vectors.
      do 60 i = 1, nonods2

c       Put potentials in ``pt'' and all current densities in ``cd''.
        if (keybc2(i).eq.0) then
            pt    =    x2(i) * conchar
            cd    = - bc2(i) * scale * flxchar
          else
```

```
          pt     =    bc2(i) * conchar
          cd     = - x2(i) * scale * flxchar
        endif

      write (2,70) i, keybc2(i), cd, pt,
     $   coords2(i,1), coords2(i,2)

60    continue

70   format(i3,1x,i2,1x,g14.8,1x,g14.8,2x,f8.1,1x,f8.1)

     write (2,80) nctr, nctrnr
80   format(/i5,1x,'iterations on the relaxed substitution and'/
     $   i5,1x,'iterations on the newton-raphson'/)

     return
     end



c    ****************** SUBROUTINE PROFILE ****************
c    This routine writes the coordinates of the electrode
c    nodes at each time step. We put this in a file all
c    by itself so that we can make a moving-boundary plot.

     subroutine profile

     implicit double precision (a-h,o-z)

     common/set0/nonelec,nonelc2
     common/set2/coords1(150,2),coords2(150,2)
     common/set24/igrid,xmin,xmax,ymin,ymax
     common/set25/tosca,nosteps,nstep

c    Put a heading in the file.
     if ( nstep . eq . 0 ) then
       write (3,10)
10     format ('This file contains coordinates of an electrode')
       write (3,11)
11     format ('profile at different instants in time.'/
     $     '                        c'/'title1'/'title2'/'title3'/
     $     'title4'/)

c      Write vertices of smallest rectangle that encloses
c      the domain (for graphics use.)
       write (3,12)
12     format('Coords of smallest rectangle enclosing domain')
c      ``xscale'' will be adjusted later; pass 50 now.
```

```
        xscale = 50.0
        write (3,13) xmin, xmax, xscale
13      format('xmin and xmax        xscale'
     $      /f7.1,1x,f7.1,10x,f7.1)
        write (3,14) ymin, ymax
14      format(/'ymin and ymax'/f7.1,1x,f7.1)

c       Write the number of nodes on the electrode.
        write (3,16) nonelec
16      format(//'number of electrode nodes'/i4)

c       Write the number of time steps and the number of
c       time steps taken in each plotting interval: (default
c       to unity, set here).
        interval = 1
        write (3,18) nosteps, interval
18      format(//'nosteps',5x,'interval'/i4,10x,i4)

        endif

      write (3,20) nstep
20    format(//i4,'th time step'/)

      write (3,30)
30    format('node',8x,'x coord',8x,'y coord')

      do 50 i = 1, nonelec
        write (3,40) i, coords1(i,1), coords1(i,2)
40      format(1x,i3,4x,f8.1,6x,f12.5)
50      continue

      return
      end



c     ****************** SUBROUTINE PREMOVE **************
      subroutine premove(coords,keyode,gap,nmin1,nmax1,nmin2,
     $   nmax2,nmin3,nmax3,nmin4,nmax4,nonods,space0)

      implicit double precision (a-h,o-z)

      dimension coords(150,2), space0(150), nonseg(10),
     $   keyode(150)

c     How many different segments make up the domain
c     boundary? Figure out this number, ``noseg'', from
```

```
c       ''keyode.'' Also count the number of nodes in each
c       segment and put this in a vector ''nonseg''.

c       Initialize the elements of nonseg zero.
        do 5 j = 1, 10
          nonseg(j) = 0.
5         continue

c       Calculate ''noseg'' and vector ''nonseg.''
        noseg = 1
        key = keyode(1)
        nonseg(1) = 1
        do 10 i = 2, nonods
          keyold = key
          key = keyode(i)
          if ( key . ne . keyold ) noseg = noseg + 1
          nonseg(noseg) = nonseg(noseg) + 1
10        continue


c       Record the node numbers of the first and last node
c       on each boundary. (We assume a simple four-boundary
c       arrangement.)

        nmin1 = 1
        nmax1 = nonseg(1)
        nmin2 = nonseg(1) + 1
        nmax2 = nonseg(1) + nonseg(2)
        nmin3 = nonseg(1) + nonseg(2) + 1
        nmax3 = nonseg(1) + nonseg(2) + nonseg(3)
        nmin4 = nonseg(1) + nonseg(2) + nonseg(3) + 1
        nmax4 = nonseg(1) + nonseg(2) + nonseg(3) + nonseg(4)

        noeelec =  ( nmax1 - 1 ) / 2
        narc    = 50
        narco2 = narc / 2

c       Calculate the relative spacing of nodes along
c       the working-electrode surface. Store this in
c       vector ''space0''.

        arc0   = 0.
        do 20 i = 1, noeelec

          n1  =  2 * i  -  1
          n2  =  2 * i  +  1
          n3  =  2 * i

          r1  =  coords(n1,1) .
```

```
      r2  =  coords(n2,1)
      r3  =  coords(n3,1)

      z1  =  coords(n1,2)
      z2  =  coords(n2,2)
      z3  =  coords(n3,2)

      ds =  2.0 / float(narc)

c     Divide each element up into ''narc'' segments and
c     take the arc length by adding up short tangents.
      do 15 k = 1, narc

        if ( k . eq . 1 )   space0(n1) = arc0

        s  =  - 1.0  +  2.0 * float( k - 1 ) / float(narc)

        dp1ds  =  s  -  0.5
        dp2ds  =  s  +  0.5
        dp3ds  =  - 2.0 * s

        drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
        dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

        dr  =  drds * ds
        dz  =  dzds * ds

        darc  =  dsqrt ( dr ** 2  +  dz ** 2 )

        arc0  =  arc0  +  darc

        if ( k . eq . narco2 ) space0(n3) = arc0
        if ( k . eq . narc   ) space0(n2) = arc0

15      continue

20    continue

c    Normalize the portion of vector ''space0'' that describes
c    the working electrode so that ''space0(nmax1) = 1.0''.

     do 22 i = 1, nmax1
       space0(i) = space0(i) / arc0
22     continue

c    Calculate the relative spacing of the second segment;
c    we take this to be a straight, vertical line:
     segment = coords(nmax2,2) - coords(nmin2,2)
     do 25 n = nmin2, nmax2
```

```
        space0(n) = ( coords(n,2) - coords(nmin2,2) ) / segment
25      continue

c     Calculate the relative spacing of the third segment;
c     we take this to be a straight, horizontal line:
      segment = coords(nmax3,1) - coords(nmin3,1)
      do 27 n = nmin3, nmax3
        space0(n) = ( coords(n,1) - coords(nmin3,1) ) / segment
27      continue

c     Calculate the relative spacing of the fourth segment;
c     we take this to be a straight, vertical line:
      segment = coords(nmax4,2) - coords(nmin4,2)
      do 30 n = nmin4, nmax4
        space0(n) = ( coords(n,2) - coords(nmin4,2) ) / segment
30      continue

c     Calculate the highest point on the original electrode surface.
      highest = 0.
      do 35 n = 1, nmax1
        try = coords(n,2)
        if ( try . gt . highest ) highest = try
35      continue

c     Calculate the vertical gap between this high point and the
c     ceiling of the domain.
      gap = coords(nmin3,2) - highest

      return
      end



c     ***************** SUBROUTINE MOVE1 ********************

c     This routine is the moving boundary part of the program.
c     It operates on the node coordinates ''coords'' and the local
c     current densites, ''x1'', and returns a new set of node
c     coordinates that have been displaced
c     according to Faraday's law for a small time step.
c     If this were as simple as it sounded, this routine wouldn't
c     be 1000 lines long...
c     GIVE MORE DETAILS.

      subroutine move1(coords,gap,nmin1,nmax1,nmin2,
     $    nmax2,nmin3,nmax3,nmin4,nmax4,space0)
```

```
      implicit double precision (a-h,o-z)

      common/set0/nonelec,nonelc2
      common/set12/scale
      common/set61/x1(150)
      common/set71/bc1(150)
      common/set81/keybc1(150),keyode1(150)
      common/set25/tosca,nosteps,nstep

      dimension coords(150,2), space0(150)
      dimension w(150), false(150,2)
      dimension iopen(150), bridge(150,2)

      noeelec =  ( nonelec - 1 ) / 2
      narc    = 1000
      narco2 = narc / 2
      rwall1 = coords(1,1)
      rwall2 = coords(nonelec,1)

c     There are four steps in this routine:

c     1) Faraday's Law:
c     Move each node normal to the surface according Faraday's
c     law.  Here we create more ''false'' nodes than we had
c     starting nodes because each point joining two elements
c     splits into two.  Some of these pairs coincide (if slope
c     was continuous across the element boundary), some overlap
c     (less than 180 degrees subtended through the electrolyte)
c     and some diverge (more than 180 degrees).

c     First put surface derivatives into vector ''w''.
      do 13 i = 1, nmax1
        key = keybc1(i)
        if ( key . eq . 0 ) then
            w(i) = bc1(i)
          else
            w(i) = x1(i)
          endif
c       Subversion:
        w(i) = w(i) / ( tosca * scale * scale )
13      continue


      do 20 i = 1, noeelec

        nf1  =  3 * i  -  1
        nf2  =  3 * i  +  1
        nf3  =  3 * i
```

```
      n1   =  2 * i  -  1
      n2   =  2 * i  +  1
      n3   =  2 * i

      r1  =  coords(n1,1)
      r2  =  coords(n2,1)
      r3  =  coords(n3,1)

      z1  =  coords(n1,2)
      z2  =  coords(n2,2)
      z3  =  coords(n3,2)

      do 15 k = 1, 3

        n   =  2 * ( i - 1 )  +  k
        nf  =  3 * ( i - 1 )  +  k  +  1

        dp1ds  =  s  -  0.5
        dp2ds  =  s  +  0.5
        dp3ds  =  - 2.0 * s

        drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
        dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

        drdn  = -dzds
        dzdn  =  drds

        denom  =  dsqrt ( drdn ** 2  +  dzdn ** 2 )
        rhat   =  drdn / denom
        zhat   =  dzdn / denom

c       Now move the node according to Faraday's law:
        rshift  =  tosca * scale**2 * w(n) * rhat
        zshift  =  tosca * scale**2 * w(n) * zhat

        false(nf,1)  =  coords(n,1)  +  rshift
        false(nf,2)  =  coords(n,2)  +  zshift

15      continue

20    continue

c     Now take care of first and last false nodes.

c     First node:
      n   =  1
      nf  =  1
      rhat = 0.0
      zhat = 1.0
```

```
      rshift  =  tosca * scale**2 * w(n) * rhat
      zshift  =  tosca * scale**2 * w(n) * zhat
      false(nf,1)  =  coords(n,1)  +  rshift
      false(nf,2)  =  coords(n,2)  +  zshift

c     Last node:
      n   =  nonelec
      nf  =  noeelec * 3  +  2
      rhat = 0.0
      zhat = 1.0
      rshift  =  tosca * scale**2 * w(n) * rhat
      zshift  =  tosca * scale**2 * w(n) * zhat
      false(nf,1)  =  coords(n,1)  +  rshift
      false(nf,2)  =  coords(n,2)  +  zshift


c     2) Fix overlaps:
c     Modify the ''false'' nodes by treating any overlapping
c     pairs.  Locate a single node between them that
c     approximately satisfies the time step for both
c     elements.  Also fix any overlap with
c     an insulator or a symmetry boundary.

c     Initialize vector ''iopen''.  Later this will be set
c     to unity at any node where there is a break in
c     the false surface.
      do 22 i = 1, nonelec
        iopen(i) = 0
22      continue

c     a) Check each inter-element joint for overlap.
c     ''nojnts'' is the number of joints, ''m'' is
c     the one we are on, and ''ijoint'' is the original
c     node number of that joint.

      nojnts = noeelec - 1
      do 30 m = 1, nojnts
        ijoint  =  2 * m  +  1
        nf1  =  3 * m  +  1
        nf2  =  3 * m  +  2

c     a1) average tangent vector at the element joint:

c       First element:
        n1  =  2 * m  -  1
        n2  =  2 * m  +  1
        n3  =  2 * m

        r1  =  coords(n1,1)
```

```
      r2  =  coords(n2,1)
      r3  =  coords(n3,1)

      z1  =  coords(n1,2)
      z2  =  coords(n2,2)
      z3  =  coords(n3,2)

      s   =  1.0

      dp1ds  =  s  -  0.5
      dp2ds  =  s  +  0.5
      dp3ds  =  - 2.0 * s

      drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
      dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

      denom  =  dsqrt ( drds ** 2  +  dzds ** 2 )
      rhat01   =  drds / denom
      zhat01   =  dzds / denom

c     Second element:
      n1  =  2 * (m + 1)  - 1
      n2  =  2 * (m + 1)  +  1
      n3  =  2 * (m + 1)

      r1  =  coords(n1,1)
      r2  =  coords(n2,1)
      r3  =  coords(n3,1)

      z1  =  coords(n1,2)
      z2  =  coords(n2,2)
      z3  =  coords(n3,2)

      s   =  -1.0

      dp1ds  =  s  -  0.5
      dp2ds  =  s  +  0.5
      dp3ds  =  - 2.0 * s

      drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
      dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

      denom  =  dsqrt ( drds ** 2  +  dzds ** 2 )
      rhat02   =  drds / denom
      zhat02   =  dzds / denom

c     Averaged unit tangent vector:

      rhatave  =  ( rhat01 + rhat02 ) / 2.0
```

```
      zhatave  =  ( zhat01 + zhat02 ) / 2.0

c     Vector pointing from false node 1 to false node 2:

      delr  =  false(nf2,1)  -  false(nf1,1)
      delz  =  false(nf2,2)  -  false(nf1,2)

c     Finally, test for overlap.  If the dot product of the
c     average tangent vector and the difference vector is
c     negative, we have overlap.

      dotprod  =  delr * rhatave  +  delz * zhatave
      if ( dotprod . lt . 0.0 ) then

        iopen(n1) = 0.

c       ''b'' is half the length of the difference vector
c       between the paired false points:
        b  =  0.5 * dsqrt( delr ** 2  +  delz ** 2 )

c       ''a'' is the length of each motion vector:
        delr = false(nf1,1) - coords(n1,1)
        delz = false(nf1,2) - coords(n1,2)
        a  =  dsqrt( delr ** 2  +  delz ** 2 )

c       Two times angle ''theta'' subtends the two motion vectors:
        theta = asin( b/a )

c       ''c'' is the length of the new common point:
        c  =  a  /  cos(theta)

c       Unit vector pointing to new point:
        delr1 = false(nf1,1) - coords(n1,1)
        delz1 = false(nf1,2) - coords(n1,2)
        delr2 = false(nf2,1) - coords(n1,1)
        delz2 = false(nf2,2) - coords(n1,2)
        delr3 = delr1 + delr2
        delz3 = delz1 + delz2
        denom = dsqrt( delr3 ** 2  +  delz3 ** 2 )
        unitr = delr3 / denom
        unitz = delz3 / denom

c       Point where the two false elements approximately intersect:
        shiftr = c * unitr
        shiftz = c * unitz
        false(nf1,1) = coords(n1,1) + shiftr
        false(nf1,2) = coords(n1,2) + shiftz
        false(nf2,1) = false(nf1,1)
        false(nf2,2) = false(nf1,2)
```

```
          elseif ( dotprod . eq . 0.0 ) then
            iopen(ijoint) = 0

          else
            iopen(ijoint) = 1

c           Determine the coordinates of the point halfway
c           between points nf1 and nf2 lying on a circular
c           arc centered at the original common node.  Enter
c           these in matrix ''bridge.''

c           Unit vector pointing to new point:
            delr1 = false(nf1,1) - coords(n1,1)
            delz1 = false(nf1,2) - coords(n1,2)
            delr2 = false(nf2,1) - coords(n1,1)
            delz2 = false(nf2,2) - coords(n1,2)
            delr3 = delr1 + delr2
            delz3 = delz1 + delz2
            denom = dsqrt( delr3 ** 2  +  delz3 ** 2 )
            unitr = delr3 / denom
            unitz = delz3 / denom

c           New point:
            dist = dsqrt( delr1 ** 2  +  delz1 ** 2 )
            deltar = dist * unitr
            deltaz = dist * unitz
            bridge(n1,1) = coords(n1,1) + deltar
            bridge(n1,2) = coords(n1,2) + deltaz

          endif

30      continue

c     While you're at it, calculate bridge points if
c     needed next to the walls:

c     Wall1:
      if ( false(2,1) . gt . rwall1 ) then
        iopen(1) = 1
        delr1 = false(1,1) - coords(1,1)
        delz1 = false(1,2) - coords(1,2)
        delr2 = false(2,1) - coords(1,1)
        delz2 = false(2,2) - coords(1,2)
        delr3 = delr1 + delr2
        delz3 = delz1 + delz2
        denom = dsqrt( delr3 ** 2  +  delz3 ** 2 )
        unitr = delr3 / denom
        unitz = delz3 / denom
```

```
        dist  = dsqrt( delr1 ** 2  +  delz1 ** 2 )
        deltar = dist * unitr
        deltaz = dist * unitz
        bridge(1,1) = coords(1,1) + deltar
        bridge(1,2) = coords(1,2) + deltaz
        endif

c     Wall2:
      nf = 3 * noeelec + 2
      nm = nf - 1
      if ( false(nm,1) . lt . rwall2 ) then
        iopen(nonelec) = 1
        delr1 = false(nm,1) - coords(nonelec,1)
        delz1 = false(nm,2) - coords(nonelec,2)
        delr2 = false(nf,1) - coords(nonelec,1)
        delz2 = false(nf,2) - coords(nonelec,2)
        delr3 = delr1 + delr2
        delz3 = delz1 + delz2
        denom = dsqrt( delr3 ** 2  +  delz3 ** 2 )
        unitr = delr3 / denom
        unitz = delz3 / denom
        dist  = dsqrt( delr1 ** 2  +  delz1 ** 2 )
        deltar = dist * unitr
        deltaz = dist * unitz
        bridge(nonelec,1) = coords(nonelec,1) + deltar
        bridge(nonelec,2) = coords(nonelec,2) + deltaz
        endif


c     3) Measure ''false'' surface:
c     Determine the total arclength of the false electrode
c     surface.  Do this numerically by stepping along in s.
c     When you reach a diverged pair of points, bridge them
c     with a circular arc centered at the common origin point
c     for the pair.

c     There are insulators or symmetry boundaries at r = rwall1
c     and r = rwall2.

c     ''inside'' equals zero until we inch inside rwall1.
c     Then it takes the value of 1.  After passing rwass2,
c     inside = 2.
      inside = 0

c     Initialize:
      arc1 = 0.

      do 40 i = 0, noeelec
```

```
      if ( i . ne . 0 ) then

        n1  =  3 * i  -  1
        n2  =  3 * i  +  1
        n3  =  3 * i

        r1  =  false(n1,1)
        r2  =  false(n2,1)
        r3  =  false(n3,1)

        z1  =  false(n1,2)
        z2  =  false(n2,2)
        z3  =  false(n3,2)

        ds =  2.0 / float(narc)

c       Divide each element up into ''narc'' segments and
c       take the arc length by adding up short tangents.
c       At each point, check and see if you have crossed
c       into the allowed range of r ( rwall1 < r < rwall2 ).
c       If so, measure for real.

        do 35 k = 1, narc

          s  =  - 1.0  +  2.0 * float( k - 1 ) / float(narc)

          p1  =  0.5 * s * ( s - 1 )
          p2  =  0.5 * s * ( s + 1 )
          p3  =  ( 1.0 - s ) * ( 1.0 + s )

          r  =  p1 * r1  +  p2 * r2  +  p3 * r3

          if ( r . ge . rwall1 . and . r . le . rwall2 ) inside = 1
          if ( r . lt . rwall1 ) inside = 0
          if ( r . gt . rwall2 ) inside = 2

          dp1ds  =  s  -  0.5
          dp2ds  =  s  +  0.5
          dp3ds  =  - 2.0 * s

          drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
          dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

          dr  =  drds * ds
          dz  =  dzds * ds

          darc  =  dsqrt ( dr ** 2  +  dz ** 2 )

c         Measure only if you're within the walls (inside=1):
```

```
c           If you are left of the left boundary, whether or not
c           you have been to the right or not, set (or reset)
c           arc1 to zero.
            if ( inside . eq . 1 ) then
                arc1 = arc1 + darc
              elseif ( inside . eq . 0 ) then
                arc1 = 0.
              endif


35          continue

          endif


c       Now measure any circular bridge that might exist
c       after this element.

        na1 = 3 * i + 1
        na2 = 3 * i + 2
        nb  = 2 * i + 1

        if ( iopen(nb) . eq . 1 ) then

          r1 = false(na1,1)
          r2 = false(na2,1)
          r3 = bridge(nb,1)

          z1 = false(na1,2)
          z2 = false(na2,2)
          z3 = bridge(nb,2)

          ds = 2.0 / float(narc)

c         Divide each bridge up into ''narc'' segments and take
c         the arc length by adding up short tangents. At each
c         point, check and see if you have crossed into the
c         allowed range of r ( rwall1 < r < rwall2 ). If so,
c         measure for real.

          do 37 k = 1, narc

            s = - 1.0 + 2.0 * float( k - 1 ) / float(narc)

            p1 = 0.5 * s * ( s - 1 )
            p2 = 0.5 * s * ( s + 1 )
            p3 = ( 1.0 - s ) * ( 1.0 + s )
```

```
            r  =  p1 * r1  +  p2 * r2  +  p3 * r3

            if ( r . ge . rwall1 . and . r . le . rwall2 ) inside = 1
            if ( r . lt . rwall1 ) inside = 0
            if ( r . gt . rwall2 ) inside = 2

            dp1ds  =  s  -  0.5
            dp2ds  =  s  +  0.5
            dp3ds  =  - 2.0 * s

            drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
            dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

            dr  =  drds * ds
            dz  =  dzds * ds

            darc  =  dsqrt ( dr ** 2  +  dz ** 2 )

c           Measure only if you're within the walls (inside=1):
c           If you are left of the left boundary, whether or not
c           you have been to the right or not, set (or reset)
c           arc1 to zero.
            if ( inside . eq . 1 ) then
                arc1  =  arc1  +  darc
              elseif ( inside . eq . 0 ) then
                arc1  =  0.
              endif

37          continue

          endif

40      continue

c     At this point, ``arc1'' is the length along the false
c     arc (within bounds) in model units.  We shorten it
c     slightly before doing step so that we're sure to
c     take care of the last point.
      arc1 = 0.9999999999 * arc1


c     4) Reposition nodes:
c     Step along the false surface again (increasing
c     ``arc2''), this time positioning the ``true''
c     nodes according to their original relative
c     spacing.  Place these in vector ``coords''.

c     ``inside'' equals zero until we inch inside rwall1.
c     Then it takes the value of 1.  After passing rwass2,
```

```
c       inside = 2.
        inside = 0

c       Initialize:
        tiny = 1.0e-13 * scale
        arc2 = tiny
        iseek = 1
        arcseek = 0.

c       Let ''iseek'' be the number of the node we are
c       trying to locate.  The distance along the false
c       surface to node k will be
c       ''arcseek  =  space0(iseek) * arc1''

        do 65 i = 0, noeelec

          if ( i . ne . 0 ) then

            n1  =  3 * i  -  1
            n2  =  3 * i  +  1
            n3  =  3 * i

            r1  =  false(n1,1)
            r2  =  false(n2,1)
            r3  =  false(n3,1)

            z1  =  false(n1,2)
            z2  =  false(n2,2)
            z3  =  false(n3,2)

            ds =  2.0 / float(narc)

c           Divide each element up into ''narc'' segments and
c           take the arc length by adding up short tangents.
c           At each point, check and see if you have crossed into the
c           allowed range of r ( rwall1 < r < rwall2 ).  If so,
c           measure for real.

            do 55 k = 1, narc

              s  =  - 1.0  +  2.0 * float( k - 1 ) / float(narc)

              p1  =  0.5 * s * ( s - 1 )
              p2  =  0.5 * s * ( s + 1 )
              p3  =  ( 1.0 - s ) * ( 1.0 + s )

              r  =  p1 * r1  +  p2 * r2  +  p3 * r3
              z  =  p1 * z1  +  p2 * z2  +  p3 * z3
```

```
      if ( r . ge . rwall1 . and . r . le . rwall2 ) inside = 1
      if ( r . lt . rwall1 ) inside = 0
      if ( r . gt . rwall2 ) inside = 2

      dp1ds  =  s  -  0.5
      dp2ds  =  s  +  0.5
      dp3ds  =  - 2.0 * s

      drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
      dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

      dr  =  drds * ds
      dz  =  dzds * ds

c     See if you've just crossed rwall1.  If so, then
c     back interpolate to lay down the first node:
      rnew  =  r + dr
      znew  =  z + dz
      if ( inside . eq . 0 . and . rnew . gt . rwall1 ) then
        back  =  - ( rnew - rwall1 ) / dr
        dsback  =   back * ds
        rint  =  rnew  +  drds * dsback
        zint  =  znew  +  dzds * dsback
        coords(iseek,1) = rint
        coords(iseek,2) = zint
        iseek = iseek + 1
        if (iseek . gt . nonelec ) go to 68
        arcseek = space0(iseek) * arc1
        endif


      darc  =  dsqrt ( dr ** 2  +  dz ** 2 )

c     Measure only if you're within the walls (inside=1):
c     If you are left of the left boundary, whether or not
c     you have been to the right or not, set (or reset)
c     arc2 to zero.
      if ( inside . eq . 0 ) then
        arc2  =  0.
      elseif ( inside . eq . 1 ) then
        arc2  =  arc2  +  darc

c       If you've just passed a node address, interpolate
c       back linearly and put a node there:
        if ( arc2 . gt . arcseek ) then
          back  =  - ( arc2 - arcseek ) / darc
          dsback  =   back * ds
          rint  =  r  +  drds * dsback
          zint  =  z  +  dzds * dsback
```

```
              coords(iseek,1) = rint
              coords(iseek,2) = zint
              iseek = iseek + 1
              if (iseek . gt . nonelec ) go to 68
              arcseek = space0(iseek) * arc1
              endif

            endif

55        continue

        endif


c       Now measure any circular bridge that might exist
c       after this element.

        na1  =  3 * i  +  1
        na2  =  3 * i  +  2
        nb   =  2 * i  +  1

        if ( iopen(nb) . eq . 1 ) then

          r1  =  false(na1,1)
          r2  =  false(na2,1)
          r3  =  bridge(nb,1)

          z1  =  false(na1,2)
          z2  =  false(na2,2)
          z3  =  bridge(nb,2)

          ds =  2.0 / float(narc)

c         Divide each bridge up into ''narc'' segments and
c         take the arc length by adding up short tangents.
c         At each point, check and see if you have crossed into the
c         allowed range of r ( rwall1 < r < rwall2 ).  If so,
c         measure for real.

          do 60 k = 1, narc

            s  =  - 1.0  +  2.0 * float( k - 1 ) / float(narc)

            p1  =  0.5 * s * ( s - 1 )
            p2  =  0.5 * s * ( s + 1 )
            p3  =  ( 1.0 - s ) * ( 1.0 + s )

            r  =  p1 * r1  +  p2 * r2  +  p3 * r3
            z  =  p1 * z1  +  p2 * z2  +  p3 * z3
```

```
      if ( r . ge . rwall1 . and . r . le . rwall2 ) inside = 1
      if ( r . lt . rwall1 ) inside = 0
      if ( r . gt . rwall2 ) inside = 2

      dp1ds  =  s  -  0.5
      dp2ds  =  s  +  0.5
      dp3ds  =  - 2.0 * s

      drds  =  r1 * dp1ds  +  r2 * dp2ds  +  r3 * dp3ds
      dzds  =  z1 * dp1ds  +  z2 * dp2ds  +  z3 * dp3ds

      dr  =  drds * ds
      dz  =  dzds * ds

c     See if you've just crossed rwall1.  If so, then
c     back interpolate to lay down the first node:
      rnew  =  r + dr
      znew  =  z + dz
      if ( inside . eq . 0 . and . rnew . gt . rwall1 ) then
        back  =  - ( rnew - rwall1 ) / dr
        dsback  =   back * ds
        rint  =  rnew  +  drds * dsback
        zint  =  znew  +  dzds * dsback
        coords(iseek,1) = rint
        coords(iseek,2) = zint
        iseek = iseek + 1
        if (iseek . gt . nonelec ) go to 68
        arcseek = space0(iseek) * arc1
        endif

      darc  =  dsqrt ( dr ** 2  +  dz ** 2 )

c     Measure only if you're within the walls (inside=1):
c     If you are left of the left boundary, whether or not
c     you have been to the right or not, set (or reset)
c     arc2 to zero.
      if ( inside . eq . 0 ) then
        arc2  =  0.
      elseif ( inside . eq . 1 ) then
        arc2  =  arc2  +  darc

c       If you've just passed a node address, put a node there:
        if ( arc2 . gt . arcseek ) then
          back  =  - ( arc2 - arcseek ) / darc
          dsback  =   back * ds
          rint  =  r  +  drds * dsback
          zint  =  z  +  dzds * dsback
          coords(iseek,1) = rint
```

```
                coords(iseek,2) = zint
                iseek = iseek + 1
                if (iseek . gt . nonelec ) go to 68
                arcseek = space0(iseek) * arc1
                endif

              endif

60          continue

          endif

65      continue

c     To allow for any error in the arc length measurements,
c     make sure that last electrode node is positioned on
c     the right-hand symmetry line:
68    nlast = 3 * noeelec + 2
      coords(nonelec,1) = false(nlast,1)


c     After the move, calculate the highest point on the new
c     electrode surface.

      highest = 0.
      do 70 n = 1, nmax1
        try = coords(n,2)
        if ( try . gt . highest ) highest = try
70      continue

c     Move the third boundary so that the vertical gap between
c     it and the highest point on the electrode is constant.

      do 80 n = nmin3, nmax3
        coords(n,2)  =  highest  +  gap
80      continue

c     As the corners are dual nodes, make sure they all get moved.
      coords(nmax2,2) = coords(nmin3,2)
      coords(nmin4,2) = coords(nmax3,2)
      coords(nmax4,2) = coords(nmin1,2)
      coords(nmin2,2) = coords(nmax1,2)

c     Now reposition the nodes on the symmetry boundaries (2 & 4).

      segment = coords(nmax2,2) - coords(nmin2,2)
      do 90 n = nmin2, nmax2
        coords(n,2) = coords(nmin2,2)  +  space0(n) * segment
90      continue
```

```
      segment = coords(nmax4,2) - coords(nmin4,2)
      do 100 n = nmin4, nmax4
        coords(n,2) = coords(nmin4,2)  +  space0(n) * segment
100     continue

      return
      end



c     ***************** SUBROUTINE MOVE2 ********************
c     This routine moves the nodes of the second domain,
c     vector coords2, after the first-domain nodes, coords1,
c     have already been moved.

      subroutine move2(coords1,coords2,gap,nmin1,nmax1,nmin2,
     $   nmax2,nmin3,nmax3,nmin4,nmax4,space0)

      implicit double precision (a-h,o-z)

      dimension coords1(150,2), coords2(150,2), space0(150)

c     Since the two domains share (part of) the electrode
c     boundary, simply equate those node coordinates:
      do 10 i = 1, nmax1
        coords2(i,1) = coords1(i,1)
        coords2(i,2) = coords1(i,2)
10      continue

c     Dual node at start of electrode:
      coords2(nmax4,1) = coords2(nmin1,1)
      coords2(nmax4,2) = coords2(nmin1,2)

c     Dual node at end of electrode:
      coords2(nmin2,1) = coords2(nmax1,1)
      coords2(nmin2,2) = coords2(nmax1,2)

c     R coordinate of top-right dual node:
      coords2(nmax2,1) = coords2(nmin2,1)
      coords2(nmin3,1) = coords2(nmin2,1)

c     R coordinate of top-left dual node:
      coords2(nmax3,1) = coords2(nmin1,1)
      coords2(nmin4,1) = coords2(nmin1,1)

c     Calculate the highest point on the new electrode surface:
      highest = 0.
```

```
      do 70 n = 1, nmax1
        try = coords2(n,2)
        if ( try . gt . highest ) highest = try
70      continue

c     Move the third (top) boundary so that the vertical
c     gap between it and the highest point on the electrode
c     is constant; Also stretch these nodes evenly in the
c     r direction:
      segment = coords2(nmax3,1) - coords2(nmin3,1)
      do 80 n = nmin3, nmax3
        coords2(n,2) = highest + gap
        coords2(n,1) = coords2(nmin3,1) + space0(n) * segment
80      continue

c     Z coordinates of top dual nodes:
      coords2(nmax2,2) = coords2(nmin3,2)
      coords2(nmin4,2) = coords2(nmax3,2)

c     Stretch the right symmetry boundary (boundary 2):
      segment = coords2(nmax2,2) - coords2(nmin2,2)
      do 90 n = nmin2, nmax2
        coords2(n,2) = coords2(nmin2,2)  +  space0(n) * segment
        coords2(n,1) = coords2(nmax1,1)
90      continue

c     Stretch the left symmetry boundary (boundary 4):
      segment = coords2(nmax4,2) - coords2(nmin4,2)
      do 100 n = nmin4, nmax4
        coords2(n,2) = coords2(nmin4,2)  +  space0(n) * segment
        coords2(n,1) = coords2(nmin1,1)
100     continue

      return
      end
```

Input-Data File

```
This file contains input data for a boundary-element
leveling problem.
Second set of kinetic data taken from Nickel-Coumarin
polarization curves.
Notch problem from Kruglikov paper.

iprint                                grid plot key (0: run;
i          i          iii             i              1: both;  2: plot)
1          1          500             0

iaxism
i
0

nonods1, the number of nodes in the extended domain.
iii
 66

nonods2, the number of nodes in the boundary layer.
iii
 66

nonelec, number of nodes on 'trode surface (dimension of NR)
iii
 27

nonelc2, number of nodes the 'trode surface (dimension of NR)
iii
 27

scale (model units/d-less unit)
ddddd.d
50000.0

vertex coords of smallest box enclosing domain
 xmin     xmax
ddddd.d ddddd.d
    0.0 20000.0
 ymin     ymax
ddddd.d ddddd.d
    0.0 99999.9

rinner       (nodal-structure units)
ddddddd.d
      0.0

s1         sg     (not applicable to this version)
```

```
d.dd        d.dd
1.0         0.5

curbar (mA/cm2)  (important)
sd.dAdEsee
-1.0  e+01

flxbar (micromoles/cm2-s) (Of no importance in this version)
sd.dddEsee
-2.69 e-03

etabar (millivolts) (Only a reference in this version)
sd.dddEsee
-7.32 e+02

noflx
ii
 2

flxi (micromol/cm2-s)
i       sd.dddEsee
1       +1.0  e+02
2       -1.0  e+02

nocur
ii
 2

        curj (mA/cm2)
j       sd.dddEsee
1       +1.0  e+10
2       -1.0  e+10

Matrix ''etaij'':

column for j = 1 ( cur = curj(1) )
i       sd.dddEsee
1       +5.0  e+01
2       +5.0  e+01

column for j = 2 ( cur = curj(2) )
i       sd.dddEsee
1       -5.0  e+01
2       -5.0  e+01

charl (centimeters)  (Not really used in this version)
sd.dddEsee
 5.0  e-03
```

```
vchar (millivolts)  (Answer doesn't really depend on this)
sd.dddEsee
 2.61 e+01

conchar (micromoles/liter) (Same as bulk concentration)
sd.dddEsee
 4.3  e+02

curchar (mA/cm2)  (This is a key input parameter)
sd.dddEsee
 2.24 e+11 (set for geometric leveling)

fixchar (micromol/cm2-s))  (Another key input parameter)
sd.dddEsee  (set now for geometric leveling)
 9.95 e-06

irestrt(1: read from restart) ikeep(1: write into restart;
i                             i            else 0.)
0                             0

beta  (kinetic symmetry parameter, not used in this version)
d.dd
0.5

tol          (relative hence dimensionless)
sd.dddEsee
 1.000e-10

relaxation factors for substitution and N.R. (param & paramnr)
d.ddddddd     d.ddddd
0.0005        1.0

max number of iterations for subst. and N.R. (nctrmax & nmaxnr)
iiii        iiii
   0          20

tosca  (dless ratio of layer-thickness step to current density)
sd.dddEsee     was 2.418e-1 before setting to 2.5e-1
 2.260e+09

nosteps  ( number of time steps you want to take )
iiii
 100

Extended domain coords and keys.  psi
Node       r or x    z or y       bc  ode loc
        ddddddd.d ddddddd.d      ii  ii  ii
  1           0.0   20000.0       1   1   1
  2        1950.9   19807.9       1   1   0
```

3	3826.8	19238.8	1	1	1
4	5555.7	18314.7	1	1	0
5	7071.1	17071.1	1	1	1
6	8314.7	15555.7	1	1	0
7	9238.8	13826.8	1	1	1
8	9807.9	11950.9	1	1	0
9	10000.0	10000.0	1	1	1
10	10192.1	8050.0	1	1	0
11	10761.2	6173.2	1	1	1
12	11685.3	4444.3	1	1	0
13	12928.9	2928.9	1	1	1
14	13656.1	2269.9	1	1	0
15	14444.3	1685.3	1	1	1
16	15286.0	1180.8	1	1	0
17	16173.2	761.2	1	1	1
18	16631.1	584.6	1	1	0
19	17097.2	430.6	1	1	1
20	17570.2	299.7	1	1	0
21	18049.1	192.1	1	1	1
22	18532.7	108.2	1	1	0
23	19019.8	48.2	1	1	1
24	19264.4	27.1	1	1	0
24	19509.3	12.0	1	1	1
25	19754.6	3.0	1	1	0
25	20000.0	0.0	1	1	1
26	20000.0	0.0	0	0	1
26	20000.0	250.0	0	0	0
27	20000.0	500.0	0	0	1
27	20000.0	750.0	0	0	0
28	20000.0	1000.0	0	0	1
29	20000.0	1500.0	0	0	0
30	20000.0	2000.0	0	0	1
31	20000.0	2500.0	0	0	0
32	20000.0	3500.0	0	0	1
33	20000.0	4500.0	0	0	0
34	20000.0	6000.0	0	0	1
35	20000.0	7500.0	0	0	0
36	20000.0	10000.0	0	0	1
37	20000.0	15000.0	0	0	0
38	20000.0	20000.0	0	0	1
39	20000.0	25000.0	0	0	0
40	20000.0	30000.0	0	0	1
41	20000.0	40000.0	0	0	0
42	20000.0	50000.0	0	0	1
43	20000.0	62500.0	0	0	0
44	20000.0	75000.0	0	0	1
45	20000.0	87500.0	0	0	0
46	20000.0	100000.0	0	0	1
47	20000.0	100000.0	0	2	1

48	10000.0	100000.0	0	2	0
49	0.0	100000.0	0	2	1
50	0.0	100000.0	0	0	1
51	0.0	87500.0	0	0	0
52	0.0	75000.0	0	0	1
53	0.0	62500.0	0	0	0
54	0.0	50000.0	0	0	1
55	0.0	40000.0	0	0	0
56	0.0	30000.0	0	0	1
57	0.0	25000.0	0	0	0
58	0.0	20000.0	0	0	1
59	0.0	17500.0	0	0	0
60	0.0	15000.0	0	0	1
61	0.0	12500.0	0	0	0
62	0.0	10000.0	0	0	1

Boundary-layer coords and keys.

Node	r or x	z or y	bc	agent ode	loc
	dddddd.d	dddddd.d	ii	ii	ii
1	0.0	20000.0	1	1	1
2	1950.9	19807.9	1	1	0
3	3826.8	19238.8	1	1	1
4	5555.7	18314.7	1	1	0
5	7071.1	17071.1	1	1	1
6	8314.7	15555.7	1	1	0
7	9238.8	13826.8	1	1	1
8	9807.9	11950.9	1	1	0
9	10000.0	10000.0	1	1	1
10	10192.1	8050.0	1	1	0
11	10761.2	6173.2	1	1	1
12	11685.3	4444.3	1	1	0
13	12928.9	2928.9	1	1	1
14	13656.1	2269.9	1	1	0
15	14444.3	1685.3	1	1	1
16	15286.0	1180.8	1	1	0
17	16173.2	761.2	1	1	1
18	16631.1	584.6	1	1	0
19	17097.2	430.6	1	1	1
20	17570.2	299.7	1	1	0
21	18049.1	192.1	1	1	1
22	18532.7	108.2	1	1	0
23	19019.8	48.2	1	1	1
24	19264.4	27.1	1	1	0
24	19509.3	12.0	1	1	1
25	19754.6	3.0	1	1	0
25	20000.0	0.0	1	1	1
26	20000.0	0.0	0	0	1
26	20000.0	250.0	0	0	0
27	20000.0	500.0	0	0	1

27	20000.0	750.0	0	0	0
28	20000.0	1000.0	0	0	1
29	20000.0	1500.0	0	0	0
30	20000.0	2000.0	0	0	1
31	20000.0	2500.0	0	0	0
32	20000.0	3500.0	0	0	1
33	20000.0	4500.0	0	0	0
34	20000.0	6000.0	0	0	1
35	20000.0	7500.0	0	0	0
36	20000.0	10000.0	0	0	1
37	20000.0	15000.0	0	0	0
38	20000.0	20000.0	0	0	1
39	20000.0	25000.0	0	0	0
40	20000.0	30000.0	0	0	1
41	20000.0	40000.0	0	0	0
42	20000.0	50000.0	0	0	1
43	20000.0	62500.0	0	0	0
44	20000.0	75000.0	0	0	1
45	20000.0	87500.0	0	0	0
46	20000.0	100000.0	0	0	1
47	20000.0	100000.0	1	2	1
48	10000.0	100000.0	1	2	0
49	0.0	100000.0	1	2	1
50	0.0	100000.0	0	0	1
51	0.0	87500.0	0	0	0
52	0.0	75000.0	0	0	1
53	0.0	62500.0	0	0	0
54	0.0	50000.0	0	0	1
55	0.0	40000.0	0	0	0
56	0.0	30000.0	0	0	1
57	0.0	25000.0	0	0	0
58	0.0	20000.0	0	0	1
59	0.0	17500.0	0	0	0
60	0.0	15000.0	0	0	1
61	0.0	12500.0	0	0	0
62	0.0	10000.0	0	0	1

Appendix B-1

Comparison of Finite-Element and Boundary-Element Methods

While all of the calculations reported in Chapters 1, 2 and 3, were done with the boundary-element method (BEM), some preliminary calculations were carried out with the finite-element method (FEM). The FEM code used to predict current distribution in the laboratory-scale active-passive cell of Chapter 1 is given in Appendix B-2. A nodal structure for this problem, containing 225 linear-triangular elements and 154 nodes, is shown in Figure B1-1A. Analogous calculations were performed with the boundary-element method (BEM), using the nodal structure of Figure B1-1B; this consists of 81 linear elements and 81 nodes, coinciding in location with the finite-element nodes that lie on the domain boundary. Figure B1-2 compares the current distributions calculated by the two techniques. It is emphasized that this isolated example cannot serve as a basis for general conclusions. However the comparison does serve to illustrate two inherent differences between the methods.

First, is is apparent that the FEM solution is a step profile, while BEM gives a continuous function, even though the two methods employ interpolation functions of the same order (linear). The reason for this is that in FEM, the piecewise-linear interpolation of potential must be *differentiated* to give current density, resulting in a constant value over each element. On the other hand, BEM uses the same order of interpolation for both the potential and its derivative. This distinction is especially relevant to electrochemical problems, in which the current density is the quantity of interest.

The second difference between the two curves is their shape: the FEM curve is shifted to the left and, in particular, features a lower current-density peak. Although

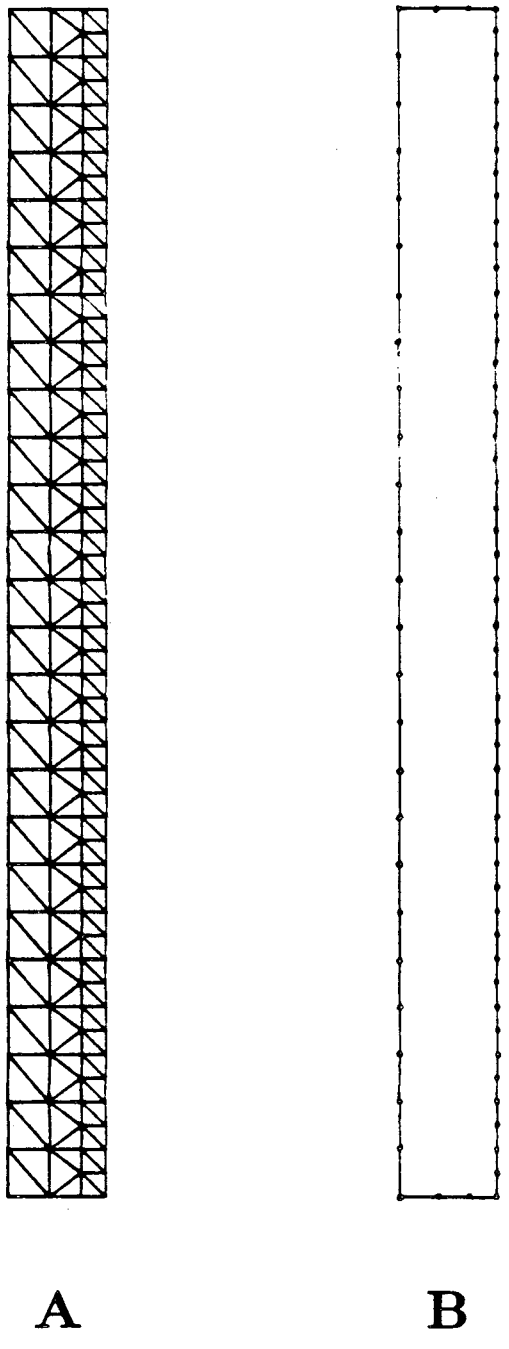


Figure B1-1. Nodal structures for a laboratory-scale active-passive cell: A—finite-element, and B—boundary-element

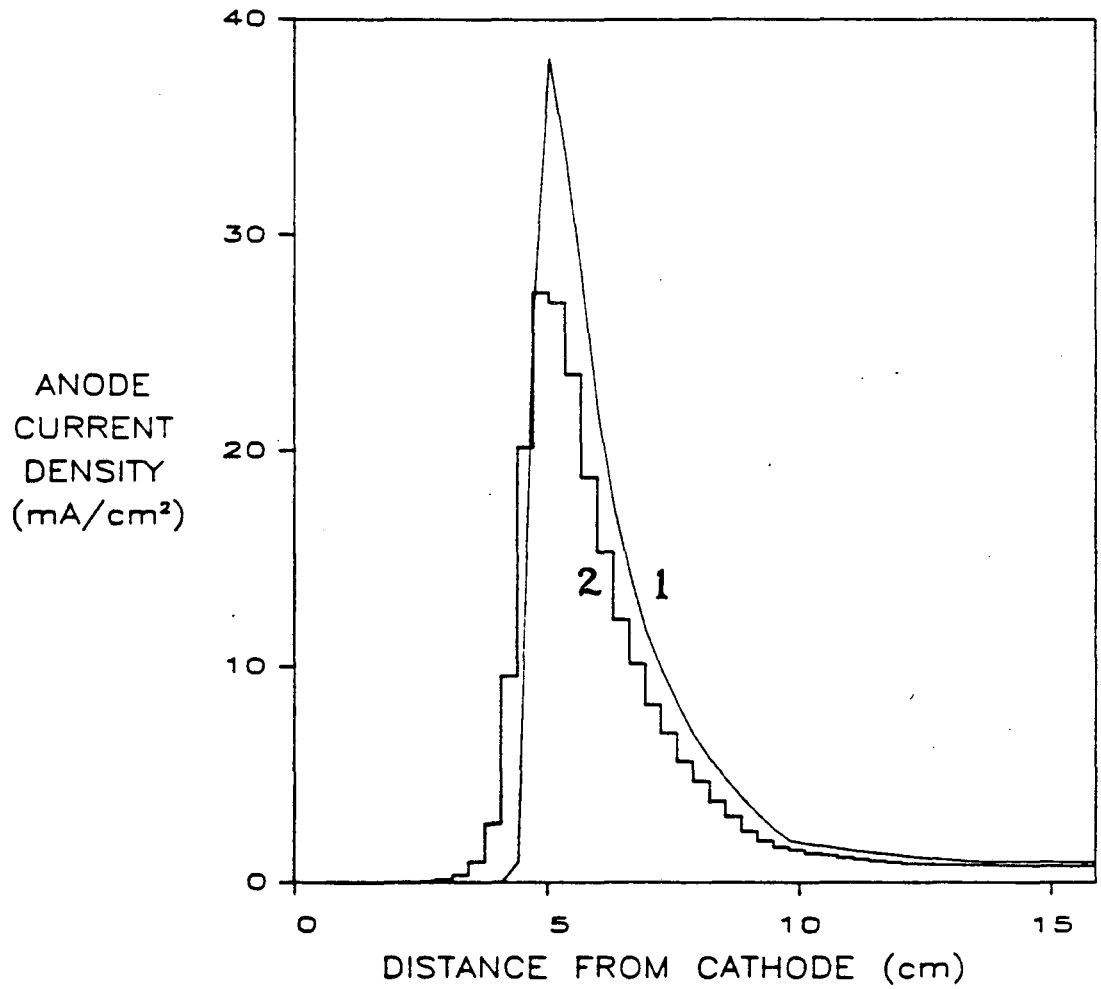


Figure B1-2. Comparison of current distributions calculated by: 1—boundary-element method, and 2—finite-element method

the exact solution is not known, we do know that its peak should correspond to the peak current density from the polarization curve, 38.8 mA/cm^2 in this example. This is satisfied much better by BEM than by FEM. The reason for this is the following. To obtain convergence, it is necessary to apply the electrode-overpotential relation as a *natural* boundary condition, *i.e.* to impose the potential *gradient* and calculate the surface potential by Laplace's equation. It is a feature of the finite-element method (both the Galerkin and variational formulations) that natural boundary conditions are not strictly enforced, *i.e.* the solution represents a compromise among the various constraints imposed (except for the *essential* boundary conditions, which are enforced exactly). Figure B1-3 illustrates this feature. The converged solution, (*i.e.* after the relative change in potential on an iteration has fallen below 10^{-5}) is shown by curve 2, but the potential-gradient profile that was *applied* on the last iteration to produce this solution is curve 3. (The two curves are related as follows: if one takes the value of surface potential corresponding to the converged solution of curve 2, and substitutes this into the overpotential expression, one obtains curve 3.) By contrast, essential boundary conditions are exactly enforced in BEM.

In summary, a limited comparison illustrates two advantages of BEM over FEM for the active-passive problem: higher-order interpolation of the potential gradient, and strict enforcement of natural boundary conditions.

Finally, an illustration is given of the comparative simplicity and ease of construction of BEM nodal structures. Figure B1-4 shows a portion of a finite-element nodal structure for the attached-bubble problem of Chapter 2. There is substantial grid refinement in the crevice region at the lower right, since high activity in the potential field is expected here. As no algorithm was available for automatic generation of this grid, it was constructed by hand, requiring considerable time. For comparison, a

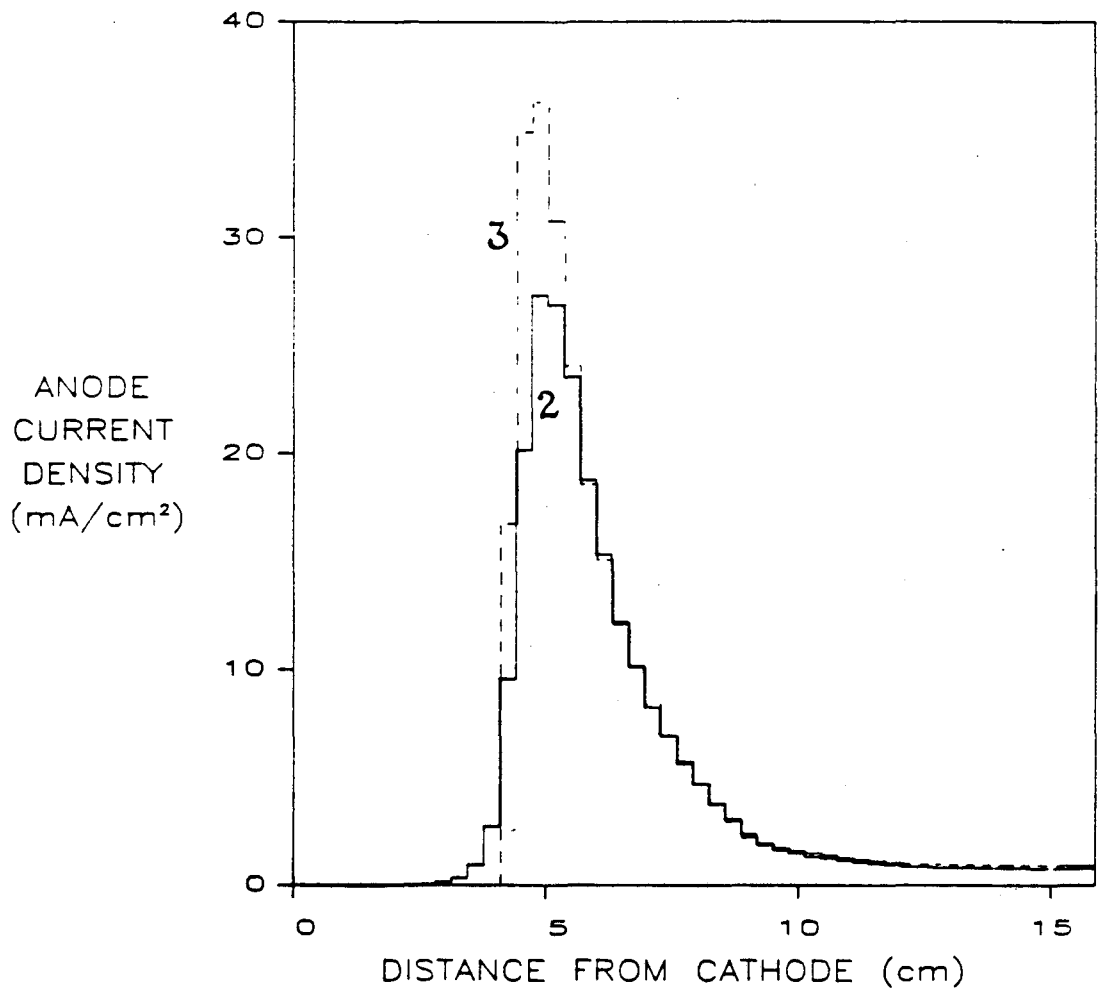


Figure B1-3. Comparison of finite-element current distributions corresponding to 2—converged solution, and 3—gradient boundary condition used for last iteration

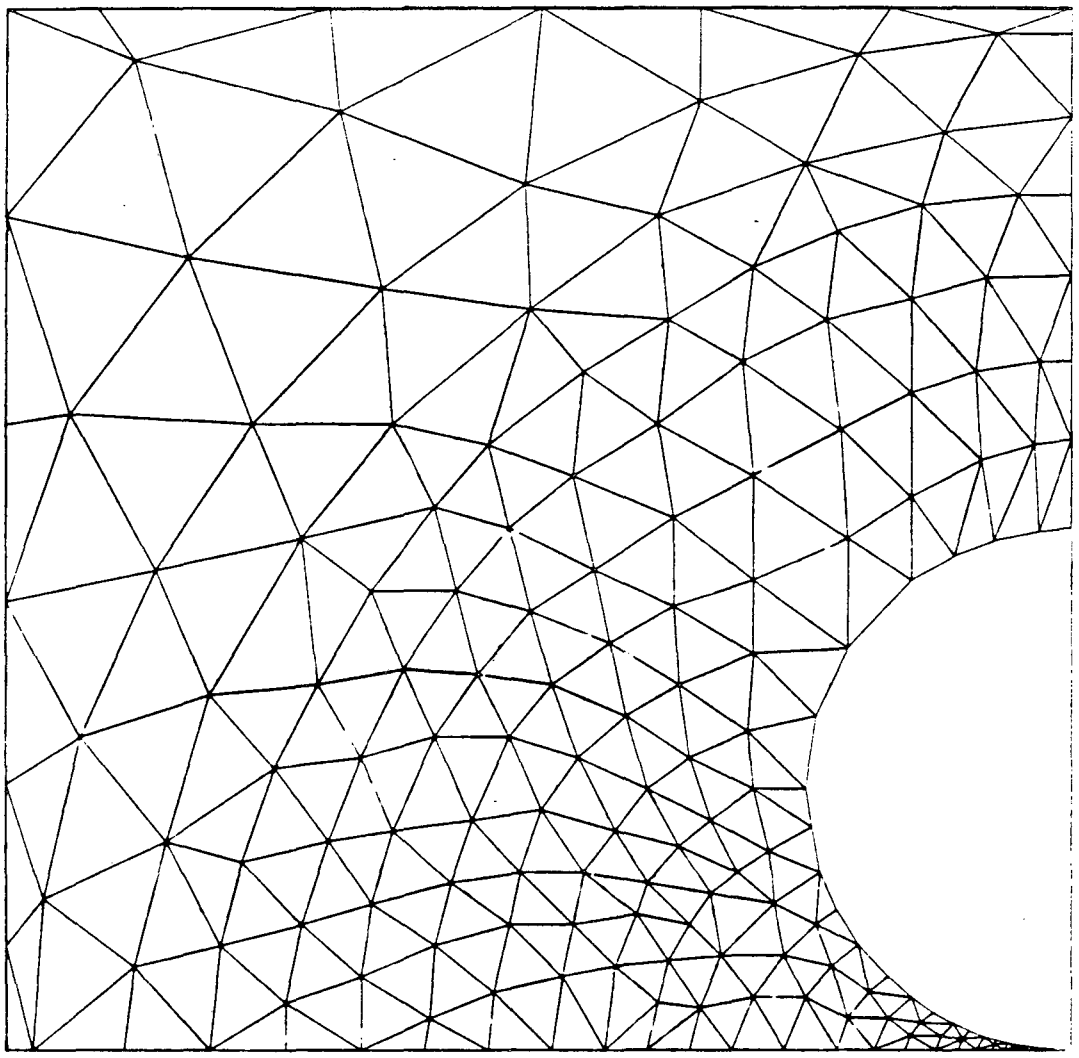


Figure B1-4. Portion of finite-element nodal structure for an attached bubble

boundary-element nodal structure used for the bubble work is shown in Figure B1-5. This particular example does not show the same degree of local refinement as Figure B1-4, but it is obvious that, even if it did, the boundary-element nodal structure would still be far less complicated.

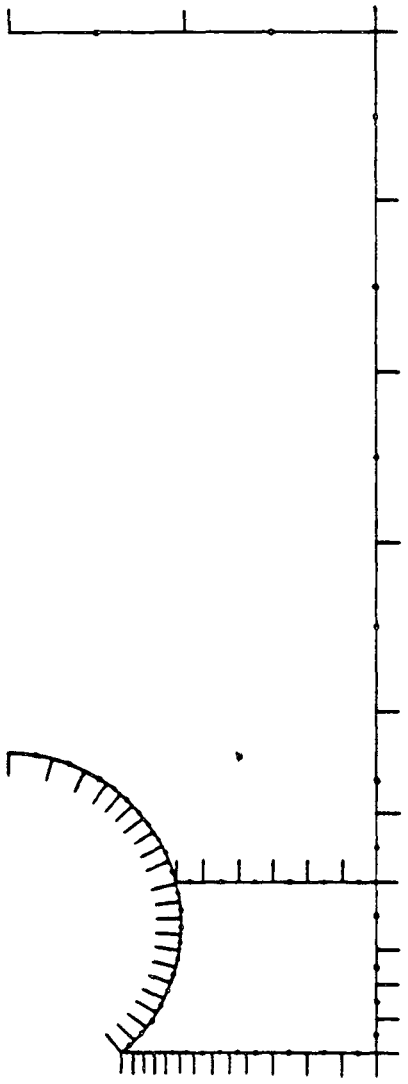


Figure B1-5. Boundary-element nodal structure for an attached bubble

Appendix B-2

Finite-Element Code for Active-Passive Model

The following Fortran program was used in preliminary calculations for the active-passive model described in Chapter 1A. A sample input-data file is given at the end of the program listing. Important variables are defined in subroutine 'input' or as they appear in the code.

```
      program fecur(input,output)

c     This program is for the active-passive problem, specifically
c     for the sample geometry used in the experiment.
c     This program is general to axisymmetric and planar problems.

c     Be sure that the local node numbering is done so that
c     1) order is counterclockwise, and 2) nodes 1 and 2 are on
c     the outside boundary (if there is one.)

      common//nonods, mbandw
      common/set1/nfixp(4),pfix(4), nofixp, noels
      common/set2/coords(154,2), ltopo(225,3), iaxism, rinner
      common/set3/big, diagq(154)
      common/set4/nolcur, cond, scale, lcur(48), cur(48)
      common/set5/nctr,nctrmax,tol,param
      common/set6/delvapp
      common/set7/noldrv, ldrv(48), drv(48)
      common/set9/nctrsp,ispotpr,intspot
      common/set11/noap,curap(20),etaap(20)
      common/set12/tmax
      common/matrix/q(154,8)
      common/vector/f(154)

c     Read in all data and do the grid generation.
      call input

c     Set up the stiffness matrix.
      call stiff

c     Forward reduce the stiffness matrix.
      call solve1

c     *********The cycle begins.********
      nctr = 0
      nctrsp = 0
110   nctr = nctr + 1
      nctrsp = nctrsp + 1

      if (nctr.ge.50) param = 0.01

c     Impose the boundary conditions.
      call bcs

c     Back-substitute and get node potentials.
      call solve2

c     Use the overpotential expression to evaluate current
c     densities corresponding to the surface potentials.
```

```
      tmax = 0.
      do 125 i=1,48
        lment = ldrv(i)
        node1 = ltopo(lment,1)
        node2 = ltopo(lment,2)
        p1 = f(node1)
        p2 = f(node2)
        pave = 0.5 * (p1 + p2)
        eta = delvapp - pave

c       Evaluate current from trapezoidal A/P curve fit.
        hold = - eta * 0.001
        if (eta.le.etaap(1)) go to 12
        hold = 0.0
        do 115 j = 2, noap
          if (eta.le.etaap(j)) go to 1445
          go to 115
1445      hold = - curap(j-1)  -  ( curap(j) - curap(j-1) ) *
     b      ( eta - etaap(j-1) ) / ( etaap(j) - etaap(j-1) )
          go to 121
115       continue
c       If we're well into passivity, set current to zero.
        hold = 0.0

121     t = abs( hold - cur(i) )
        if (t.gt.tmax) tmax=t
        cur(i) = hold
        drvnew = - cur(i) * scale / ( cond * 1000.0 )
        drv(i) =  param * drvnew  +  (1.0-param) * drv(i)
125     continue

c      As an option, print out tmax for each iteration.
c       if (nctr.gt.1) go to 165
c       print 160
c160    format(/2x,*nctr*,3x,*tmax*)
c165    print 166, nctr, tmax
c166    format(2x,i4,4x,f10.8)
c       go to 1673
c1665   print 167, cur(1), cur(16), cur(31), cur(46)
c167    format(f7.3,1x,f7.3,1x,f7.3,1x,f7.3)
c       go to 1675

c      Spot print current values as a convergence aid.
c1673   if (ispotpr.ne.1) go to 168
c       if (nctrsp.ge.intspot) go to 1665
c1675   if (nctrsp.gt.intspot) nctrsp = 1

c168    if(tmax.lt.tol) go to 180
      if (tmax.lt.tol) go to 180
```

```
c       If you've exceeded the max allowed iterations, nctrmax,
c       say so and call the output routine.
        if (nctr.ge.nctrmax) go to 170

c       If you don't meet the convergence criterion,
c       do the cycle again.
        go to 110

170     print 171
171     format(*You've exceeded nctrmax iterations.*)

c       If you have converged, call the output subroutine.
180     call output

        stop
        end



        subroutine input

c       This subroutine reads in all the input data and, if
c       requested, prints it all back out again as a check.
c       It also generates coords, ltopo and ldrv.

        common//nonods,mbandw
        common/set1/nfixp(4),pfix(4),nofixp,noels
        common/set2/coords(154,2),ltopo(225,3),iaxism,rinner
        common/set4/nolcur,cond,scale,lcur(48),cur(48)
        common/set5/nctr,nctrmax,tol,param
        common/set6/delvapp
        common/set7/noldrv,ldrv(48),drv(48)
        common/set9/nctrsp,ispotpr,intspot
        common/set11/noap,curap(20),etaap(20)

c       Read in all the input.

        Read 310
310     format(////)

c       Read iprint (print all data unless iprint equals
c       zero), ispotpr (values are spotprinted every 'intspot'
c       iterations if ispotpr = 1), and intspot.
        read 315, iprint, ispotpr, intspot
315     format(///i1,9x,i1,9x,i3)

c       Read iaxism.  This is a flag; if it reads zero, the
c       problem is not axisymmetric.  A value of one or
```

```
c       anything else indicates axisymmetry.
        read 316, iaxism
316     format(///i1)

c       Read nonods, the number of nodes in the grid.
        read 317, nonods
317     format(///i3)

c       Read noels, the number of elements.
        read 318, noels
318     format(///i3)

c       Read nofixp, the number of nodes at which the
c       potential is to be fixed.
        read 319, nofixp
319     format(///i3)

c       Read noldrv, the number of elements at which
c       derivative is fixed.
        read 3195, noldrv
3195    format(///i3)

c       Read nolcur, number of elements at which current
c       is calculated.
        read 3196, nolcur
3196    format(///i3)

c       Read mbandw, a good guess for an upper limit on
c       the half-bandwidth of the stiffness matrix.
        read 320, mbandw
320     format(///i3)

c       Read the conductivity of the electrolyte in mho/cm2.
        read 321, cond
321     format(///f6.4)

c       Read the value of the applied anode potential with
c       respect to the rest potential, ''delvapp.''
        read 3215, delvapp
3215    format(///f5.3)

c       Read the A/P current/voltage characteristic data
        read 3216, noap
3216    format(///i3)
        read 3217
3217    format(//)
        read 3218, (curap(i),etaap(i),i=1,noap)
3218    format(5x,f7.3,2x,f7.4)
```

```
c     Read in the scale factor for the model representation
c     of the domain.  One centimeter of distance in the
c     electrochemical cell equals (SCALE) * (one unit of
c     distance in the model.)    For the test geometry,
c     scale = 0.3175 cm/model unit.
      read 322, scale
322   format(///f10.4)

c     Read in the value of rinner.  This is the distance
c     (in the distance units of the model) from the innermost
c     point of the nodal structure to the centerline.  If the
c     problem is not axisymmetric, we don't
c     need rinner, but we must read a value anyhow.
      read 323, rinner
323   format(//////////f9.1)

c     Read the value of the convergence tolerance, tol.
      read 324, tol
324   format(///f12.10)

c     Read in the relaxation parameter.
      read 325, param
325   format(///f7.5)

c     Read in values and iteration counts of param adjustments.
      read 3255, nctr2, param2, nctr3, param3, nctr4, param4
3255  format(///10x,i4,10x,f7.5/10x,i4,10x,f7.5/10x
     b,i4,10x,f7.5)

c     Read in the maximum number of iterations allowed.
      read 326, nctrmax
326   format(///i4)

c     Read in the first guess for uniform current density at
c     the electrode in mA/cm2.
      read 327, curinit
327   format(///f7.3)

c     The following lines generate a FE grid for the rotating
c     cylinder test geometry.  There is a regular repeating
c     pattern detailed in the 10/12 writup ''New AP Grid.''

c     Generate ''Coords'', the list of nodal coordinates
c     in z and r.  Let ''rm2'' be the length of a unit grid
c     cell in model units.  For the experimental cell, the
c     scale is 0.3175 cm/model unit.

      do 360 m=1,25
        rm2 = float( 2 * (m-1) )
```

```
      m6 = 6 * (m-1)

      k = 1
      node = m6 + k
      coords(node,1) = rm2
      coords(node,2) = 0.

      k = 2
      node = m6 + k
      coords(node,1) = rm2
      coords(node,2) = 1.0

      k = 3
      node = m6 + k
      coords(node,1) = rm2 + 1.0
      coords(node,2) = 1.0

      k = 4
      node = m6 + k
      coords(node,1) = rm2
      coords(node,2) = 2.25

      k = 5
      node = m6 + k
      coords(node,1) = rm2
      coords(node,2) = 4.0

      k = 6
      node = m6 + k
      coords(node,1) = rm2 + 1.0
      coords(node,2) = 0.0

360   continue

c     The following coordinates don't fit in the automatic scheme:

      coords(151,1) = 50.0
      coords(151,2) =  0.0
      coords(152,1) = 50.0
      coords(152,2) =  1.0
      coords(153,1) = 50.0
      coords(153,2) =  2.25
      coords(154,1) = 50.0
      coords(154,2) =  4.0

c     Generate the element topology matrix, ''ltopo.''

      do 361 m = 1, 25
        m9 = 9 * (m-1)
```

```
m6 = 6 * (m-1)

k = 1
lment = m9 + k
ltopo(lment,1) = m6 + 1
ltopo(lment,2) = m6 + 6
ltopo(lment,3) = m6 + 2

k = 2
lment = m9 + k
ltopo(lment,1) = m6 + 6
ltopo(lment,2) = m6 + 3
ltopo(lment,3) = m6 + 2

k = 3
lment = m9 + k
ltopo(lment,1) = m6 + 6
ltopo(lment,2) = m6 + 7
ltopo(lment,3) = m6 + 3

k = 4
lment = m9 + k
ltopo(lment,1) = m6 + 7
ltopo(lment,2) = m6 + 8
ltopo(lment,3) = m6 + 3

k = 5
lment = m9 + k
ltopo(lment,1) = m6 + 2
ltopo(lment,2) = m6 + 3
ltopo(lment,3) = m6 + 4

k = 6
lment = m9 + k
ltopo(lment,1) = m6 + 3
ltopo(lment,2) = m6 + 10
ltopo(lment,3) = m6 + 4

k = 7
lment = m9 + k
ltopo(lment,1) = m6 + 3
ltopo(lment,2) = m6 + 8
ltopo(lment,3) = m6 + 10

k = 8
lment = m9 + k
ltopo(lment,1) = m6 + 4
ltopo(lment,2) = m6 + 10
ltopo(lment,3) = m6 + 5
```

```
        k = 9
        lment = m9 + k
        ltopo(lment,1) = m6 + 10
        ltopo(lment,2) = m6 + 11
        ltopo(lment,3) = m6 + 5

361     continue
c     Now make the following revisions for the end of the nodal
c     structure.
      ltopo(222,2) = 153
      ltopo(224,2) = 153
      ltopo(223,3) = 153
      ltopo(225,1) = 153
      ltopo(225,2) = 154


c     Generate ''ldrv,'' the list of elements at which the normal
c     deriviative will be specified.

      do 362 m = 2, 25
        m9 = 9 * (m-1)
        m2 = 2 * (m-1)
        ldrv(m2 - 1) = m9 + 1
        ldrv(m2) = m9 + 3
362     continue

c     For the present, lcur is identical to ldrv.
      do 363 i=1,noldrv
        lcur(i) = ldrv(i)
363     continue

c     Read the coordinates of each node in global order.
c      read 364
c364   format(///)
c      read 365,(coords(i,1),coords(i,2),i=1,nonods)
c365   format(7x,f7.1,1x,f7.1)

c     Read in the element topology.  This is a list of the
c     elements in order and, for each one, the global node
c     numbers of local nodes 1,2 and 3 in that order,
c     (numbered counterclockwise.)
c      read 366
c366   format(///)
c      read 367,(ltopo(i,1),ltopo(i,2),ltopo(i,3),i=1,noels)
c367   format(8x,i3,5x,i3,5x,i3)

c     Read in nfixp and pfix side by side.  This will be an
c     ordered list of nodes at which potential is to be fixed,
c     and the corresponding values of potential in volts.
```

```
      read 368
368   format(///)
      read 3685,(nfixp(i),pfix(i),i=1,nofixp)
3685  format(8x,i3,5x,f6.4)

c     Read in lcur, alist of elements, in spacial order, at
c     which current density is to be calculated and printed out.
c      read 3693
c3693  format(///)
c      read 3694,(lcur(i),i=1,nolcur)
c3694  format(8x,i3)

c     Initialize the electrode gradients and currents.
      do 3695 i=1,nolcur
        drv(i) = - curinit * scale / (cond * 1000.0)
        cur(i) = curinit
3695    continue

c     Print out all the input data (if iprint is not
c     equal to zero.)

      if (iprint.eq.0) go to 398

      print 3696, iaxism
3696  format(*iaxism.ne.0 for axisymmetric problem.*/i2)

      print 370, nonods, noels, nofixp, mbandw
370   format(/*nonods*/i3//*noels*/i4//*nofixp*/i4//*mbandw*/i4)

      print 371, cond
371   format(/*conductivity in mho/cm*/f7.4)

      print 372, scale
372   format(/*scale, centimeters per model unit*/f11.4)

      print 373, rinner
373   format(/*rinner, distance off the centerline,
     $   model units*/f10.1)

      print 3731, tol
3731  format(/*tol*/f12.10)

      print 3732, param
3732  format(/*param*/f7.4)

      print 3733, nctrmax
3733  format(/*nctrmax*/i4)

      print 3734, curinit
```

```
3734  format(/*curinit*/f8.3)

      print 3735, delvapp
3735  format(/*delvapp*/f6.4)

      print 375
375   format(/*node coordinates*/*node*,4x,*x or z*,
     $   2x,*y or r*)
      print 376,(i,coords(i,1),coords(i,2),i=1,nonods)
376   format(i3,5x,f8.1,1x,f8.1)

      print 380
380   format(/*element topology*/*element*,2x,*node 1*,
     $    2x,*node 2*,2x,*node 3*)
      print 381,(i,ltopo(i,1),ltopo(i,2),ltopo(i,3),i=1,noels)
381   format(i3,5x,i3,5x,i3,5x,i3)

      print 385
385   format(/*node numbers and values of fixed potentials*)
      print 386
386   format(*count*,5x,*node*,2x,*potential*)
      print 387,(i,nfixp(i),pfix(i),i=1,nofixp)
387   format(2x,i3,6x,i3,2x,f7.4)

      print 390
390   format(/*elements at which current density is to
     $    be calculated*)
      print 391
391   format(*count*,4x,*element*)
      print 392, (i,lcur(i),i=1,nolcur)
392   format(2x,i3,6x,i3)

398   return
      end



      subroutine stiff

c     In this subroutine, we formulate the big ''stiffness''
c     matrix, q, in the element equations.

c     We also set aside some diagonal elements of q in the
c     vector ''diagq.'' We use these later on in subroutine bcs.

      real x(3),y(3),b(3),c(3),d(3,3)
```

```
      common//nonods,mbandw
      common/set1/nfixp(4),dummie(4),nofixp,noels
      common/set2/coords(154,2),ltopo(225,3),iaxism,rinner
      common/set3/big,diagq(154)
      common/matrix/q(154,8)

c     Start by filling half-banded matrix q with zeros.

      do 402 i = 1, nonods
        do 401 j = 1, mbandw
          q(i,j) = 0
401       continue
402     continue

c     Set up the stiffness matrix.  For each element, k, make
c     a contribution from the local element equations to the
c     global set of equations.

      do 480 lment = 1, noels

        do 410 nlocal = 1, 3
          nodeno = ltopo(lment,nlocal)
          x(nlocal) = coords(nodeno,1)
          y(nlocal) = coords(nodeno,2)
410       continue

        b(1) = y(2) - y(3)
        b(2) = y(3) - y(1)
        b(3) = y(1) - y(2)

        c(1) = x(3) - x(2)
        c(2) = x(1) - x(3)
        c(3) = x(2) - x(1)

        d(1,1) = b(1)*b(1) + c(1)*c(1)
        d(2,2) = b(2)*b(2) + c(2)*c(2)
        d(3,3) = b(3)*b(3) + c(3)*c(3)
        d(1,2) = b(1)*b(2) + c(1)*c(2)
        d(2,1) = d(1,2)
        d(1,3) = b(1)*b(3) + c(1)*c(3)
        d(3,1) = d(1,3)
        d(2,3) = b(2)*b(3) + c(2)*c(3)
        d(3,2) = d(2,3)

c       Delx2 is delta (the area of the triangular element)
c       times two.

        delx2 = b(1)*c(2) - b(2)*c(1)
```

```
c       If the problem is not axisymmetric, set the centroid
c       equal to one and finish calculating the local stiffness
c       matrix d(3,3).

        cen = 1.0
        if(iaxism.eq.0) go to 415

c       If the problem is axisymmetric, calculate the
c       centroid of the element.  This can be thought of
c       as an average distance between the element and
c       the axis of symmetry.

        cen = rinner + ( y(1) + y(2) + y(3) )/3

415     do 425 i = 1, 3
          do 420 j= 1, 3
            d(i,j) = d(i,j) * cen / delx2
420         continue
425       continue

c     Now we add the stiffness matrix for this element
c     into the global stiffness matrix.

        do 440 ilocal = 1, 3
          iglobl = ltopo(lment,ilocal)
          do 435 jlocal = 1, 3
            jglobl = ltopo(lment,jlocal)

            if (iglobl.gt.jglobl) go to 435

            iband = iglobl
            jband = jglobl - (iglobl - 1)

            q(iband,jband) = q(iband,jband) + d(ilocal,jlocal)

435         continue
440       continue
480     continue

c     Save, for later use in formulating vector f in
c     successive solutions, those diagonal elements of
c     q which correspond to nodes at which potential
c     is to be fixed.  Stick 'em in vector diagq.

      do 450 i = 1, nofixp
        ifixp = nfixp(i)
        diagq(i) = q(ifixp,1)
450     continue
```

```
      return
      end



      subroutine bcs

c     This subroutine formulates the forcing vector f, the
c     vector which embodies the boundary conditions of the problem.
c     BC's of both types (essential and natural) are treated.

      common//nonods
      common/set1/nfixp(4),pfix(4),nofixp
      common/set2/coords(154,2),ltopo(225,3),iaxisym,rinner
      common/set3/big,diagq(154)
      common/set7/noldrv,ldrv(48),drv(48)
      common/vector/f(154)

c     Fill vector f with zeros.

      do 505 i =1, nonods
        f(i) = 0
505     continue

c     The value of big, an artifice for the solving-
c     bookkeeping scheme, was set in subroutine solve1.
c     (Be sure that you call solve1 before calling bcs.)

c     Assign values to those elements of f corresponding
c     to fixed potentials.

      do 510 i =1, nofixp
        ifixp = nfixp(i)
        f(ifixp) = pfix(i) * diagq(i) * big
510     continue

c     Here we make contributions to the forcing vector,
c     f, which come from the natural boundary condition.
c     (fixed gradients.)

      do 520 i=1, noldrv
        idrv = ldrv(i)
        node1 = ltopo(idrv,1)
        node2 = ltopo(idrv,2)
        node3 = ltopo(idrv,3)
        y1 = coords(node1,2)
        y2 = coords(node2,2)
```

```
        y3 = coords(node3,2)
        x1 = coords(node1,1)
        x2 = coords(node2,1)
        disp = sqrt ( (x1-x2)*(x1-x2) + (y1-y2)*(y1-y2) )
        cen = 1.
        if (iaxism.ne.0) cen = (y1 + y2 + y3)/3.0 + rinner
        f(node1) = f(node1) + drv(i) * disp * cen
        f(node2) = f(node2) + drv(i) * disp * cen
520     continue

      return
      end



      subroutine solve1

c     This subroutine basically does forward reduction on
c     the stiffness matrix.

c     The matrix a comes in as symmetric, banded,
c     positive-definite matrix of length ''nonods''
c     and half-bandwidth ''mbandw.'' This matrix is
c     decomposed into three factors l, d and l-transpose.
c     d and l-transpose are stored (overlapped) in the
c     matrix a when it is returned to the main program.

c     This subroutine is always followed by subroutine solve2.

      common//nonods,mbandw
      common/set1/nfixp(4),dummie(4),nofixp
      common/set3/big
      common/matrix/a(154,8)

c     In keeping with the scheme for imposing the essential
c     boundary conditions, multiply by a large number, ''big,''
c     those diagonal elements of matrix ''a'' that correspond
c     to nodes at which the potential is to be fixed.

      big = 1.e15

      do 605 i = 1, nofixp
        ifixp = nfixp(i)
        a(ifixp,1) = big * a(ifixp,1)
605     continue

c     Forward reduce matrix a.
```

```
      ipjmax = nonods + 1
      kmax = nonods -1
      do 625 k = 1, kmax
        do 620 ip = 2, mbandw
          i = k + ip - 1
          do 480 j = 1, mbandw
            ipj = i + j
            if (ipj.gt.ipjmax) go to 485
            ipjmk = i + j - k
            if (ipjmk.gt.mbandw) go to 485
            a(i,j) = a(i,j) - a(k,i-k+1)*a(k,j+i-k)/a(k,1)
480         continue
485       dummy = 1
620       continue
625     continue

c     Transform the reduced matrix into an overlapping
c     storage of diagonal matrix d and upper-unit-triangular
c     matrix l-transpose.

      do 635 i = 1, nonods
        do 630 j = 2, mbandw
          a(i,j) = a(i,j) / a(i,1)
630       continue
635     continue

      return
      end



      subroutine solve2

c     This subroutine operates on the forcing vector
c     and returns a vector of answer potentials (in the
c     same storage location.) It uses the components of
c     the stiffness matrix (determined in solve1) and
c     performs back substitution.

      common//nonods,mbandw
      common/matrix/a(154,8)
      common/vector/v(154)

c     Throughout this subroutine, matrix a contains the
c     matrix components l-transpose and d. The contents
c     of vector v change as noted.
```

```
c     This subroutine may immediately follow solve1 or may
c     be called on to perform successive solutions involving
c     a given stiffness matrix.

c     Start by determining the intermediate vector fprime
c     form the forcing vector f by solving the equation
c     l * fprime = f.

      do 660 i = 1, nonods
        sum = 0
        do 650 j =2, mbandw
          if (j.gt.i) go to 655
          sum = sum + a(i-j+1,j) * v(i-j+1)
650       continue
655     v(i) = v(i) - sum
660     continue

c     Multiply the intermediate vector, fprime, by the inverse
c     of the diagonal matrix, d.

      do 665 i = 1, nonods
        v(i) = v(i) / a(i,1)
665     continue

c     Back substitute the last time to obtain node potentials.
c     vector v goes from holding intermediate vector fprime to
c     holding the answer potentials.

      jmax = mbandw - 1
      do 680 if = 1, nonods
        i = nonods + 1 - if
        sum = 0
        do 670 j = 1, jmax
          ipj = i + j
          if (ipj.gt.nonods) go to 675
          sum = sum + a(i,j+1) * v(ipj)
670       continue

675     v(i) = v(i) - sum
680     continue

      return
      end
```

subroutine output

```
c     This subroutine prints out the node potentials in
c     order.  It also calculates current density at the
c     outside surface of any element specified in LCUR
c     (a list of elements at which current density is
c     to be calculated, in spacial order.)  It prints
c     out these values of current density (vector CUR)

      common//nonods
      common/set2/coords(154,2),ltopo(225,3)
      common/set4/nolcur,cond,scale,lcur(48),cureta(48)
      common/set5/nctr
      common/set12/tmax
      common/vector/f(154)

      dimension curgrd(48)

      print 710
710   format(/////*here are the node potentials.*/)
      print 711
711   format(*node*,2x,*potential*)
      print 712,(i, f(i), i = 1, nonods)
712   format(i3,3x,f7.4)

c     Print heading for printout of current densities.
      print 715
715   format(//*here are the current densities and
     $   their locations.*)
      print 716
716   format(/*count*,3x,*element*,3x,*i field*,2x,*i
     $   eta*,,6x,*x1*,6x,
     $   *y1*,8x,*x2*,6x,*y2*)

c     Calculate the current densities.

      do 730 i=1,nolcur

        lment = lcur(i)
        node1 = ltopo(lment,1)
        node2 = ltopo(lment,2)
        node3 = ltopo(lment,3)

        x1 = coords(node1,1)
        y1 = coords(node1,2)

        x2 = coords(node2,1)
        y2 = coords(node2,2)

        x3 = coords(node3,1)
        y3 = coords(node3,2)
```

```
      p1 = f(node1)
      p2 = f(node2)
      p3 = f(node3)

c     Find the coordinates of the point (x4,y4), the
c     perpendicular projection on point 3 onto the line
c     containing points 1 and 2.

      d12 = sqrt( (y2-y1)*(y2-y1) + (x2-x1)*(x2-x1) )

      x4 = (y3-y1)*(x2-x1)*(y2-y1)
      x4 = x4 + x3*(x2-x1)*(x2-x1)
      x4 = x4 + x1*(y2-y1)*(y2-y1)
      x4 = x4/(d12*d12)

      y4 = (x3-x1)*(y2-y1)*(x2-x1)
      y4 = y4 + y3*(y2-y1)*(y2-y1)
      y4 = y4 + y1*(x2-x1)*(x2-x1)
      y4 = y4/(d12*d12)

c     Interpolate (or extrapolate) to find the value of
c     potential at the shadow point, (x4,y4) according to
c     the planar interpolation among p1, p2, and p3.

      d24 = sqrt( (x4-x2)*(x4-x2) + (y4-y2)*(y4-y2) )

c     If point 4 lies beyond point 1, we must allow for this.

      if (d24.gt.d12) go to 720

      d14 = sqrt( (x4-x1)*(x4-x1) + (y4-y1)*(y4-y1) )
      p4 = p1 + (p2-p1)*d14/d12
      go to 725

720   p4 = p2 + (p1-p2)*d24/d12
725   d34 = sqrt( (y3-y4)*(y3-y4) + (x3-x4)*(x3-x4) )

c     Calculate the potential gradient.
c     Think of SCALE as centimeters per model unit.

      grad = (p4 - p3) / (d34 * scale)

c     Calculate the current density from ohm's law.

      curgrd(i) = - cond * grad * 1000.

c     Print out the current density for this element, mA/cm2.
      print 729, i,lment,curgrd(i),cureta(i),x1,y1,x2,y2
729   format(i3,6x,i3,8x,f9.3,2x,f9.3,2x,
```

```
     $          f7.1,1x,f7.1,1x,f7.1,1x,f7.1)

730    continue

       print 740, nctr
740    format(/i4,* iterations*)
       print 750, tmax
750    format(/*tmax equals *,f10.7)
       return
       end
```

Input-Data File

```
This file contains input data for a 154-node A/P problem.
The grid is generated within the program (input routine).
The solution scheme involves imposing the natural bc at
the electrode.
comments

iprint   ispotpr   nctrsp
i          i         iii
1          0          20

iaxism
i
1

nonods
iii
154

noels
iii
225

nofixp
iii
  4

noldrv
iii
 48

nolcur
iii
 48

mbandw
iii
  8

cond  (mho/cm)
d.dddd
0.276

delvapp (applied cell potential wrt. rest potential, volts)
d.ddd
1.200

noap (number of i-v pairs used to describe the A/P BC)
```

```
iii
 18

curap and etaap (A/P b.c. pairs, mA/cm2 and Volts)
skip ddd.ddd  dd.dddd
1       0.0     0.0
2       0.197   0.0304
3       0.393   0.0609
4       0.590   0.0913
5       0.787   0.122
6       0.983   0.152
7       1.967   0.183
8       6.883   0.213
9      12.29    0.243
10     19.67    0.274
11     29.50    0.304
12     35.40    0.335
13     38.35    0.365
14     34.41    0.396
15     24.58    0.426
16     16.22    0.456
17      0.98    0.487
18      0.0     0.517

scale   (centimeters per model unit)
ddddd.dddd
    0.3175

vertex coords of smallest box enclosing domain
 xmin     xmax
ddddd.d ddddd.d
    0.0      4.0
 ymin     ymax
ddddd.d ddddd.d
    0.0     25.0

rinner
ddddddd.d
      4.0

tol
d.dddddddddd
0.0001

param (fraction of new value used in relaxed substitution)
d.ddddd
0.15

iteration counts and param values for adjusting the
```

```
          iiii          d.ddddd              convergence parameter
nctr2       25   param2 0.1
nctr3       75   param3 0.1
nctr4      200   param4 0.1

nctrmax
iiii
4000

curinit (mA/cm2)
ddd.ddd
 -5.0

Node Numbers and Values of Fixed Potentials
Count    Node  Potential
skip     iii     d.dddd
1          1     0.
2          2     0.
3          4     0.
4          5     0.
```

This report was done with support from the Department of Energy. Any conclusions or opinions expressed in this report represent solely those of the author(s) and not necessarily those of The Regents of the University of California, the Lawrence Berkeley Laboratory or the Department of Energy.

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.

LAWRENCE BERKELEY LABORATORY
TECHNICAL INFORMATION DEPARTMENT
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720