Lawrence Berkeley National Laboratory
Recent Work

Title
DESIGN OF THIN-GAP CHANNEL FLOW CELLS

Permalink
https://escholarship.org/uc/item/9z33f1bb

Author
Edwards, V.

Publication Date
1986-07-01
DESIGN OF THIN-GAP CHANNEL FLOW CELLS

V. Edwards
(Ph.D. Thesis)

July 1986

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

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.
Design of Thin-Gap Channel Flow Cells

Victoria Edwards

Ph. D. Thesis

Department of Chemical Engineering
University of California
and
Lawrence Berkeley Laboratory
Berkeley, CA 94704

July, 1986
Design of Thin-Gap Channel Flow Cells

Victoria Edwards

Abstract

This dissertation is an investigation of thin-gap channel electrochemical reactors. A thin-gap flow cell consists of parallel plates, each with an embedded electrode. Electrolyte flows between the plates, and electrochemical reactions occur on the electrode surfaces. The term “thin-gap” applies when the spacing between the plates is thin compared to the diffusion-boundary-layer thickness.

The first three chapters of the dissertation contain a theoretical investigation of channel flow cells. A mathematical model is presented that includes the effects of interacting boundary layers, simultaneous electrochemical reactions, and ohmic potential drop. Two techniques are developed for calculating the steady-state current, concentration, and potential distribution in the flow cell.

Experimental verification of the mathematical model is presented in the last two chapters. The experiments were carried out with a $K_3Fe(CN)_6/K_4Fe(CN)_6/NaOH$ electrolyte in a channel flow cell with nickel electrodes. The current-voltage curves showed excellent agreement with the model over a wide range of boundary-layer thicknesses.
Acknowledgment

I would like to thank Professor Newman for everything he has taught me. His insight and methodology for tackling difficult problems are resources I have only begun to tap. In addition to having a vast wealth of knowledge and expertise, Professor Newman is a generous man who is dedicated to his students. Professor Tobias' advice — both technical and nontechnical — is gratefully acknowledged. I also thank him for his friendship and for reviewing the manuscript. I thank Professor DeJonghe for his review of the manuscript.

The machine-shop personnel are to be commended, particularly Fred Wolff and Andy Anderson, for their helpful suggestions, and Clay Taylor, for building the flow cell.

I am happy to have made several high-quality friends at Berkeley, particularly Gina Whitney and Mike Matlosz. My friends have provided me with an enjoyable and stimulating work environment and encouragement when it was needed.

Finally, I would like to thank Dad for his technical advice and training and both parents for their constant support.

This work was supported by the Assistant Secretary for Conservation and Renewable Energy, Office of Energy Systems Research, Energy Storage Division of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.
Table of Contents

Abstract

Acknowledgments

Chapter 1. Analysis of Thin-Gap Channel Flow Cells

1. Introduction
   1.1. Literature Review
   1.2. A Mathematical Model of Thin-Gap Channel Flow Cells
   1.2.1. Assumptions
   1.2.2. Governing Equations
   1.3. Dimensionless Formulation
   1.4. Solution Techniques
   1.4.1. Advantages and Disadvantages of the Two Solution Techniques
   1.4.2. Mesh Spacing
   1.4.3. Integration Methods
   1.5. Summary

Nomenclature for Chapter 1

Chapter 2. The Asymmetric Graetz Problem in Channel Flow

2.1. Introduction
2.2. Problem Statement
2.3. The Lévêque Approach
2.4. The Graetz Approach
2.5. Solution of the Eigenvalue Problem
2.6. Combining the Graetz and Lévêque Solutions
2.7. Asymptotic Forms for Large Eigenvalues
2.8. Empirical Approach
2.9. Summary of Results

Nomenclature for Chapter 2

Chapter 3. Thin-Gap-Model Results

3. Introduction
   3.1. Limiting Cases of Current Distribution
   3.1.1. Results for Thin Diffusion Boundary Layers
   3.1.2. Results for Straight Current Lines
   3.2. Thin-Gap-Model Results for a Plating Reaction
   3.3. Results for a Redox Reaction

Nomenclature for Chapter 3
Chapter 4. Experimental Apparatus and Procedure

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1. Introduction</td>
<td>73</td>
</tr>
<tr>
<td>4.2. Selection of the Electrochemical System</td>
<td>74</td>
</tr>
<tr>
<td>4.3. Electrolyte Properties and Preparation</td>
<td>78</td>
</tr>
<tr>
<td>4.4. Channel-Flow-Cell Apparatus and Procedure</td>
<td>79</td>
</tr>
<tr>
<td>4.4.1. Experimental Apparatus</td>
<td>79</td>
</tr>
<tr>
<td>4.4.2. Experimental Procedure</td>
<td>86</td>
</tr>
<tr>
<td>4.5. Rotating-Disk Experiments</td>
<td>87</td>
</tr>
<tr>
<td>4.5.1. Experimental Procedure</td>
<td>87</td>
</tr>
</tbody>
</table>

Nomenclature for Chapter 4

Chapter 5. Comparison of Experimental and Theoretical Results

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1. Introduction</td>
<td>92</td>
</tr>
<tr>
<td>5.2. Results for Thin Boundary Layers</td>
<td>93</td>
</tr>
<tr>
<td>5.3. Discussion</td>
<td>105</td>
</tr>
<tr>
<td>5.4. Measured and Simulated Polarization Curves</td>
<td>108</td>
</tr>
</tbody>
</table>

Nomenclature for Chapter 5

Appendices

Appendix A. The Léveque Series

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1. Introduction</td>
<td>126</td>
</tr>
<tr>
<td>A.2. Mathematical Formulation</td>
<td>126</td>
</tr>
</tbody>
</table>

Appendix B. Numerical Solution of the Asymmetric Graetz Problem

Appendix C. Program EIGEN

Appendix D. Iterative Methods

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>D.1. Introduction</td>
<td>143</td>
</tr>
<tr>
<td>D.2. Comparison of Iterative Methods</td>
<td>144</td>
</tr>
<tr>
<td>D.3. Application of Some Iterative Methods to the Rotating-Disk Problem</td>
<td>150</td>
</tr>
<tr>
<td>D.3.1. Cutting the Loop</td>
<td>151</td>
</tr>
<tr>
<td>D.3.2. Calculating the Jacobian</td>
<td>152</td>
</tr>
<tr>
<td>D.4. Iterative Methods for the Channel Problem</td>
<td>155</td>
</tr>
<tr>
<td>D.4.1. Successive Substitution</td>
<td>155</td>
</tr>
<tr>
<td>D.4.2. Newton-Raphson with a Global Approximation Method</td>
<td>156</td>
</tr>
<tr>
<td>D.5. Collocation Applied to the Channel Problem</td>
<td>157</td>
</tr>
<tr>
<td>D.5.1. Introduction</td>
<td>157</td>
</tr>
<tr>
<td>D.5.2. Choice of Trial Functions</td>
<td>159</td>
</tr>
<tr>
<td>D.5.3. Basic Iteration Scheme</td>
<td>160</td>
</tr>
<tr>
<td>D.5.4. Limiting-Current Problems</td>
<td>170</td>
</tr>
<tr>
<td>D.5.5. First-Order Continuation</td>
<td>172</td>
</tr>
</tbody>
</table>
D.6. Results at the Limiting Current 173
D.7. Summary and Suggestions for Future Work 179

Appendix E. Programs for Investigating Iterative Methods for the Rotating-Disk Problem 181

E.1. Program CURDB 181
E.2. Program CURDBN 194
E.3. Program CURDE 206
E.4. Program CURDEN 218

Appendix F. Program CHANNEL 227

F.1. Program Description 227
F.2. Major Subroutines 231
F.3. Program Listing 234

Appendix G. Program NEWCHAN 319

G.1. Program Description 319
G.2. Major Subroutines 323
G.3. Program Listing 326

Appendix H. Tables of Experimental Parameters 379

References 393
1. Analysis of Thin-Gap Channel Flow Cells

1. Introduction

A channel flow cell consists of two parallel plates, which serve as the anode and the cathode for electrochemical reactions. The electrolyte flows between the electrodes, and the current flow is roughly perpendicular to the fluid flow. Figure 1-1 shows a schematic of a channel flow cell. When the gap between the two plates is thin compared to the diffusion-boundary-layer thickness, the cell is called a thin-gap channel flow cell. The advantage of a thin-gap cell is its low resistance.

Channel cells are used in many industrial electrochemical processes, such as copper refining, some zinc-halogen energy storage cells, and the Monsanto process for conversion of acrylonitrile to adiponitrile in the preparation of nylon.1-8 Channel flow cells are attractive for such applications because they provide continuous production, are simple to operate and maintain, and do not require a high capital investment.9

Some cells have a membrane separator at the midplane of the channel; others are undivided. Separators are used to prevent “chemical shorting,” where a product from one electrode diffuses to the other electrode and reacts back to the starting material. With a separator, it is possible for the anolyte and catholyte to have different compositions and flow rates,10 allowing greater control over the selectivity of the reaction. The problems with separators, however, are that they are expensive and they increase significantly the cell resistance, which is a critical factor in electroorganic syntheses because the electrolytes typically have low conductivities. In some cases, the separator must be eliminated because an intermediate produced on one electrode is needed for the reaction on the other electrode. Reactions of this type, called “paired” syntheses, are attractive because both electrode reactions contribute directly to useful products. Some
Figure 1-1. Channel flow cell geometry.
examples of paired syntheses\textsuperscript{1} include the epoxidation of propylene to form propylene oxide,\textsuperscript{1,2,11-15} cathodic hydrodimerization of acrylonitrile to adiponitrile with anodic oxidation of 2-propanol to acetone,\textsuperscript{16} and production of sorbitol and gluconic acid from glucose.\textsuperscript{17} Undivided thin-gap cells may be useful for such applications because the gap is thinner than the diffusion boundary layer. If the gap is too thin, however, there may be electrical shorts caused by impurities bridging the gap, or bubbles from gas-evolving side reactions may become trapped in the cell. Also, as the gap spacing is decreased, the heat generation and pressure drop increase. Thus, the design of a thin-gap cell requires an analysis of the economic tradeoff between decreased cell resistance, increased pressure drop, and increased diffusion of products across the gap.

The analysis of thin-gap channel flow cells is complex because several coupled phenomena must be taken into account, such as mass transfer, thermodynamics, and the kinetics of multiple reactions. Although channel-flow-cell models have been developed previously,\textsuperscript{18-27} each model contains restrictive assumptions. In this study, a laminar-flow model is developed that allows for multiple electrochemical reactions in cells with interacting diffusion boundary layers (thin-gap cells). The model predicts current, concentration, and potential distributions along the electrode surfaces. The major limitation of the model is that it does not treat homogeneous reactions or gas-evolving reactions.

Chapter 1 presents, first, a brief review of previous laminar-flow models, and, second, a discussion of the new thin-gap model. Chapter 2 develops a solution that is needed in Duhamel's superposition integral to construct the solution to the channel-flow problem with arbitrary axially-varying boundary conditions. Theoretical results are presented in chapter 3. Chapter 4 describes an experiment made to verify the model, and chapter 5 discusses the comparison between the theoretical and experimental
results.

1.1. Literature Review

Modeling electrochemical systems, such as channel flow cells, requires equations describing mass transfer, current flow, and electroneutrality. Mass transfer is generally described by material balances and flux expressions that contain terms representing diffusion, convection, and migration in the electric field. The current flow is caused by the motion of charged species and is therefore related to the fluxes of the ions. Electroneutrality is observed in all solutions, except in the thin (≈ 1 to 10 nm) double charge layer near electrodes and other boundaries. These equations are discussed in chapter 11 of Newman’s book.28

In this work, we focus on mathematical models that assume laminar flow between the plates. This assumption is valid if the Reynolds number is less than about 2100. The choice between laminar and turbulent flow is dictated by the economic tradeoff between enhanced mass transport and increased pressure drop. Acosta et al.29 have examined this tradeoff in an experimental investigation of thin-gap flow cells. Another incentive for using turbulent flow is to ensure that the flow is fast enough to sweep out bubbles from any gas-evolving side reactions.

Another assumption sometimes made in mathematical models of flow cells is that the diffusion boundary layers are thin, which will be true when

\[ L < 0.005 \, Re \, Sc \, d_e. \]  

This assumption simplifies the analysis of the mass transfer because not only can the electrodes be treated separately, but also the parabolic velocity profile can be taken to be linear throughout the boundary layer, and the Léveque approximation28,30,31 may be used.32 If
then the mass transfer is fully developed, and the electrodes must be treated simultaneously.

A third assumption is that migration of the reacting ions may be neglected in the flux equations. This assumption is valid when there is an excess of nonreacting ("supporting") electrolyte.

The assumptions discussed above — thin boundary layers and excess supporting electrolyte — have been used both by Parrish and Newman and by Lee and Selman. Parrish and Newman obtained the current and concentration distributions in an undivided channel flow cell with a single metal deposition and dissolution reaction. Lee and Selman included the effects of a separator and of finite electrode resistance. Both models account for mass transfer, reaction kinetics, thermodynamics, and ohmic potential drop.

White et al. have developed two models that include migration and multiple reactions, without invoking the thin-diffusion-layer assumption. Their first model assumes that the aspect ratio, $h/L$, is small, and that the current lines are therefore straight. That model, therefore, does not account for the two-dimensional structure of the potential distribution. This approach is similar to that taken by Alkire and Ng for modeling flow-by porous electrodes. Fedkiw and Risch discuss the two-dimensional nature of the potential distribution in flow-by porous electrodes. Recently, Mader et al. published a simplified version of this model that has a single mesh point in the axial direction. This simplification cuts the computer time by a factor of 40, but the results are significantly less accurate. They suggest using the simple model for screening sets of parameters to use in the more expensive model. The model

\[ L > 0.05 \text{Re Sc } d_e, \] (1-2)

leads the potential difference arising from the flow of current through the resistive solution.
of Nguyen et al.\textsuperscript{19} is much more complete than the earliest model; it includes axial migration and diffusion and time dependence. A recently-published extension of Nguyen's model\textsuperscript{39} includes a CSTR at the outlet and partial recycle to the channel. Although their model does not include homogeneous chemical reactions within the channel, it could be easily extended to do so. Homogeneous reactions are particularly important in electro-organic reactions, such as the paired syntheses mentioned earlier.\textsuperscript{1,2,11-17}

Sakellaropoulos and Francis\textsuperscript{23-25} discuss channel flow cells in the context of controlling selectivity in cells with multiple competing reactions (parallel\textsuperscript{24,25} or consecutive\textsuperscript{23} reactions). Their model assumes plug flow in the channel, and it does not include ohmic drop or reactions at the other electrode.

Several limiting cases for channel-flow-cell behavior are analyzed by Pickett.\textsuperscript{26} One limiting case, for "short electrodes," is the Lévêque solution, which applies when mass transfer alone dominates the current distribution. Effects of finite electrode width or short hydrodynamic entry lengths are also discussed. Another limiting case, for "long electrodes," is fully-developed mass transfer. In this case, however, Pickett assumes constant flux along the electrode surfaces and does not account for ohmic potential drop or electrode kinetics. These effects should be included in models of cells below the limiting current.

Bürgi et al.\textsuperscript{27} present a model for mass-transfer controlled metal deposition in radial thin-layer cells, which are in the pilot-plant stage of development by BASF for electro-organic syntheses,\textsuperscript{1,2,15,40-43} such as the production of adiponitrile.\textsuperscript{16,40} A radial cell is similar to a channel cell, but the average linear velocity decreases with radial distance from the central entry port.

Other variations on the channel flow cell have been proposed. The "Swiss roll" cell consists of rolled-up nickel sheet electrodes and separators placed in a cylindrical hous-
ing. This cell is used commercially to carry out one of the reactions in the synthesis of vitamin \( \text{C} \).\(^{44} \) Another variation is the "pump cell" proposed by Fleischmann, Jansson, \( \text{et al.}^{14,15,45,46} \)

1.2. A Mathematical Model of Thin-Gap Channel Flow Cells

1.2.1. Assumptions

Presented here is a multiple-reaction model of channel flow cells that does not invoke the assumption of thin diffusion layers or of straight current lines. The model accounts for mass transfer, electrode kinetics and thermodynamics, and ohmic potential drop. This model, however, does not include the effect of migration; thus it is only valid for systems with excess supporting electrolyte. Because most industrial processes use excess supporting electrolyte to decrease the cell resistance, migration of the reacting species is often negligible. The other assumptions in the model are that there are no homogeneous reactions or gas-evolving reactions, there is no separator, the fluid is in fully-developed laminar flow (the Reynolds number must be less than \( 2100 \)), axial diffusion is negligible (true for Péclet number \( > 10 \)), the fluid properties are constant, and the electrode kinetics are described by the Butler-Volmer equation. Although other kinetic equations could have been used, the Butler-Volmer equation is convenient, and it applies to most electrochemical reactions.

1.2.2. Governing Equations

Figure 1-1 shows the channel geometry. \( L \) is the electrode length, \( h \) is the gap thickness, and \( B=\frac{h}{2} \) is the channel halfwidth.

The governing equations and their underlying assumptions are discussed below. In systems with excess supporting electrolyte, the potential in the solution, as measured by a reference electrode of a given kind, satisfies Laplace's equation\(^{28} \)
with boundary conditions

$$\frac{\partial \Phi}{\partial y} = 0 \text{ at } y = -B \text{ and } y = B \quad x < 0 \text{ and } x > L \quad (1-4a)$$

and

$$\frac{\partial \Phi}{\partial y} = -\frac{i_{an}(x)}{\kappa_\infty} \text{ at } y = -B \quad 0 < x < L \quad (1-4b)$$

$$\frac{\partial \Phi}{\partial y} = \frac{i_{cath}(x)}{\kappa_\infty} \text{ at } y = B \quad 0 < x < L \quad (1-4c)$$

where \(y = -B\) is the location of the anode and \(\kappa_\infty\) is the feed-solution conductivity. The normal current density \(i_{an}\) is taken to be positive, and \(i_{cath}\) is negative (current flows from the solution into the cathode).

The solution to equation 1-3, evaluated near the cathode surface, is

$$\Phi_{cath}(x) = \Phi^* - \frac{1}{2\pi\kappa_\infty} \left\{ \int_0^L \left[ i_{cath}(x') \ln \sinh^2\left(\frac{\pi (x-x')}{2h}\right) \right. \right.$$  

$$+ \left. i_{an}(x') \ln \cosh^2\left(\frac{\pi (x-x')}{2h}\right) \right] \, dx' \right\}, \tag{1-5}$$

where \(\Phi^*\) is an integration constant. This equation can be rewritten

$$\Phi_{cath}(x) = \Phi^* - \frac{1}{2\pi\kappa_\infty} \int_0^L \left[ i_{cath}(x') + i_{an}(x') \right] \ln \cosh^2\left(\frac{\pi (x-x')}{2h}\right) \, dx'$$

$$- \frac{1}{2\pi\kappa_\infty} \int_0^L i_{cath}(x') \ln \tanh^2\left(\frac{\pi (x-x')}{2h}\right) \, dx'. \tag{1-6}$$

This form is convenient for evaluating the potentials numerically and for evaluating the potentials infinitely far upstream and downstream. The expression for the potential near the anode surface is the same as equations 1-5 and 1-6, but with the subscripts reversed.
The integration constant, Φ*, is determined by the requirement of equal total currents on the two electrodes:

\[ \int_0^L [i_{an}(x') + i_{cath}(x')] dx' = 0, \]

where the total current on an electrode is the sum of the partial current densities from each electrode reaction, for example,

\[ i_{an}(x) = \sum_j i_{j,an}(x). \]

The flux of a species \( i \) is given, in the absence of migration, by Fick's law. Because the governing equations are linear, the Fick's-law expression can be rewritten as a superposition integral involving the step-function-boundary-condition solutions to the convective diffusion equation without axial diffusion (see chapter 2):

\[
N_{i,cath}(x) = -\frac{D_i}{B} \int_0^x \frac{dc_{i,cath}}{dz} \left| \frac{\partial \theta}{\partial \xi}(\zeta - \zeta', \xi = -1) \right| dx',
\]

\[
+ \frac{D_i}{B} \int_0^x \frac{dc_{i,an}}{dz} \left| \frac{\partial \theta}{\partial \xi}(\zeta - \zeta', \xi = 1) \right| dx', \tag{1-9a}
\]

and

\[
N_{i,an}(x) = \frac{D_i}{B} \int_0^x \frac{dc_{i,an}}{dz} \left| \frac{\partial \theta}{\partial \xi}(\zeta - \zeta', \xi = -1) \right| dx',
\]

\[
- \frac{D_i}{B} \int_0^x \frac{dc_{i,cath}}{dz} \left| \frac{\partial \theta}{\partial \xi}(\zeta - \zeta', \xi = 1) \right| dx'. \tag{1-9b}
\]

Both fluxes are in the +y direction. The dimensionless variables are defined as

\[ \theta = \frac{c_i - c_f}{c_{i,o} - c_f}, \tag{1-10} \]

\[ \zeta = x \frac{D_i}{\frac{3}{2} B^2 <v>}, \tag{1-11} \]

\[ \xi = y/B, \tag{1-12} \]

where \( c_f \) is the feed concentration of species \( i \) and \( c_{i,o} \) is the surface concentration of
The flux is also given by Faraday's law:

\[ N_i = - \sum_j \frac{s_{ij} i_j}{n_j F} \]  

The sum is over the electrode reactions, which can be written in the form

\[ \sum_i s_{ij} M_i z_i \rightarrow n_j e^- \]

where \( n_j \) denotes the number of electrons transferred, \( M_i \) is a symbol for the chemical formula of species \( i \), and \( z_i \) is the charge number of species \( i \).

The current density, \( i_j \), depends exponentially on the surface overpotential according to the Butler-Volmer kinetic expression:

\[ i_j = i_{o,j} \left[ \exp \left( \frac{\alpha_{o,j} F}{RT} \eta_{s,j} \right) - \exp \left( -\frac{\alpha_{o,j} F}{RT} \eta_{s,j} \right) \right] \]

where \( i_{o,j} \) is the exchange current density for reaction \( j \) and depends on the surface concentrations \( c_{i,o} \). Usually, this dependence is expressed as

\[ i_{o,j} = i_{o,j,ref} \prod_i \left( \frac{c_{i,o}}{c_{i,ref}} \right)^{\gamma_{ij}} \]

where \( i_{o,j,ref} \) is the exchange current density evaluated at reference concentrations, which should be selected to be nonzero. Newman\(^{28} \) gives a set of rules for estimating \( \gamma_{ij} \), based on the stoichiometry and transfer coefficients of the reaction.

The surface overpotential for reaction \( j \), \( \eta_{s,j} \), is the potential of the working electrode relative to a reference electrode of the same kind as the working electrode, placed in the solution adjacent to the surface of the working electrode, but just outside the

\(^{1}\)Reference 51 presents the Nusselt number, which is proportional to \( \partial \theta / \partial x \). Unfortunately the Nusselt number was
diffuse double layer. Figure 1-2 illustrates the placement of reference electrodes for assessing potential variations in a solution. The letter $s$ designates reference electrodes of the same kind as the working electrode, and the letter $g$ refers to reference electrodes of a given kind. Using the notation of figure 1-2, the surface overpotential for a reaction $j$ is written

$$\eta_{s,j} = V - V_{r1s} = V - V_{r1g} - (V_{r1s} - V_{r1g}), \quad (1-17a)$$
or

$$\eta_{s,j} = V - \Phi^o - U_{j,o}, \quad (1-17b)$$

where $\Phi^o = V_{r1g}$ is the potential just outside the double layer, as measured by a reference electrode of a given kind, and

$$U_{j,o} = V_{r1s} - V_{r1g}$$

$$= \left[ U_j^e - \frac{RT}{n_jF} \sum_i s_{ij} \ln \left( \frac{c_{io}}{\rho_o} \right) \right] - \left[ U_{re}^e - \frac{RT}{n_{re}F} \sum_i s_{i, re} \ln \left( \frac{c_{i, re}}{\rho_o} \right) \right]. \quad (1-18)$$

The subscript $re$ denotes the reference-electrode reaction. The electrode potential $V$ in equation 1-17 is specified with respect to an arbitrary zero of potential. For example, if $V_{cath}$ is set equal to zero, then $V_{an}$ is equal to the total cell potential, which is specified in this problem formulation. Note that the average current density on the electrodes is not set here but is calculated from

$$\langle i \rangle = \frac{1}{L} \int_0^L i_{an}(x) \, dx. \quad (1-19)$$

A summary of the equations and unknowns for $i$ species and $j$ reactions is given in table 1-1. The left column shows the number of equations and whether the equation is based on the channel gap ($h$) instead of the equivalent diameter ($2h$).
Figure 1-2. Reference electrodes, which may be imaginary, positioned in the bulk solution and within the diffusion layer. The letter $s$ designates reference electrodes of the same kind as the working electrode, and the letter $g$ refers to reference electrodes of a given kind.
Table 1-1

<table>
<thead>
<tr>
<th>Equations</th>
<th>Unknowns</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2i(x)$</td>
<td>Fluxes</td>
</tr>
<tr>
<td>Fick’s Law (1-9a,b)</td>
<td>$2i(x)$</td>
</tr>
<tr>
<td>$2i(x)$</td>
<td>Surface Concentrations</td>
</tr>
<tr>
<td>Faraday’s Law (1-13)</td>
<td>$2i(x)$</td>
</tr>
<tr>
<td>$2j(x)$</td>
<td>Partial Currents</td>
</tr>
<tr>
<td>Butler-Volmer (1-15,16)</td>
<td>$2j(x)$</td>
</tr>
<tr>
<td>$2j(x)$</td>
<td>Surface Overpotentials</td>
</tr>
<tr>
<td>Surface Overpotential (1-17b)</td>
<td>$2j(x)$</td>
</tr>
<tr>
<td>$2j(x)$</td>
<td>$U_{j,o}$ (1-18)</td>
</tr>
<tr>
<td>$2j(x)$</td>
<td>$U_{j,o}$</td>
</tr>
<tr>
<td>$2(x)$</td>
<td>$\Phi^e(x)$ (1-5)</td>
</tr>
<tr>
<td>$2(x)$</td>
<td>$\Phi^e(x)$</td>
</tr>
<tr>
<td>1 Equal Currents (1-7)</td>
<td>1 $\Phi^e$</td>
</tr>
<tr>
<td>$2(x)$</td>
<td>Total Currents (1-8)</td>
</tr>
<tr>
<td>$2(x)$</td>
<td>Total Currents</td>
</tr>
<tr>
<td>1 Cell Potential</td>
<td>1 $V_{an}$</td>
</tr>
<tr>
<td>1 Zero of Potential</td>
<td>1 $V_{cath}$</td>
</tr>
</tbody>
</table>

evaluated at each axial position, $x$.

1.5. Dimensionless Formulation

It is useful to write the equations in table 1-1 in dimensionless form, because fewer parameters are needed to define each problem. For a system with $i$ species and $j$ reactions, $(2 + 2ij + 4i + 4j)$ dimensionless parameters are needed. These are parameters characterizing the electrode reactions, the geometry, the transport properties, and the operating conditions. Table 1-2 lists the dimensionless parameters. The parameters $J_i$, $N$, and $Gz$ are defined as follows:
\[ J_j = \frac{-n_m}{s_{RM}} \left( \frac{FL}{RT\kappa_{\infty}} \right)^{i_{o,j,ref}} \tag{1-20} \]

\[ N = \frac{n_m^2}{s_{RM}^2} \frac{P^2D_RC_{R\infty}}{RT\kappa_{\infty}} \left( \frac{6<v>L^2}{hD_R} \right)^{1/3} \tag{1-21} \]

\[ Gz = \frac{\pi}{4} Re Sc \frac{d_e}{L} = \frac{\pi d_e^2 <v>}{4D_R L} \tag{1-22} \]

As the Graetz number, \( Gz \), approaches infinity (for example, high flowrate), the thin-diffusion-boundary-layer approximation becomes valid. As \( Gz \to 0 \), the boundary layer becomes thick.

The reader should note that the definition of the Graetz number in equation 1-22 differs from that used in the literature. Based on Graetz's original work for mass transfer in a tube, Drew and others began to use the quantity \( A<v>/DL \) to describe mass transfer in other geometries, where \( A \) is the flow cross-sectional area. By 1942 this quantity was being called a Graetz number. For a tube, this presents no difficulty, because it reduces to

\[ Gz_{tube} = \frac{\pi d_e^2 <v>}{4DL} \tag{1-23} \]

For an annulus, however, this Graetz number becomes

\[ Gz_{annulus} = \frac{\pi (d_o^2 - d_i^2) <v>}{4DL} \]

\[ = \frac{\pi d_o^2 (1 + \kappa) <v>}{4DL (1 - \kappa)} \] \tag{1-24}

where \( \kappa \) is the ratio of the inner to the outer radius. As \( \kappa \to 1 \), the parallel plate geometry is approached, but, according to this definition, the Graetz number approaches infinity; therefore the square of the hydraulic diameter should appear in \( \omega C_p/kL \) in heat-transfer nomenclature.
place of the actual cross-sectional area. To be consistent with the definition of the Graetz number for a tube, we include the factor $\pi/4$ in equation 1-22.

1.4. Solution Techniques

In this section, two approaches and algorithms are presented, along with some details, such as mesh spacing and integration techniques. The first approach is a method of successive substitution applied to the potential distribution, and the second approach is a method of collocation with a Newton-Raphson iteration on the coefficients. The advantages and disadvantages of each method are listed in section 1.4.1.

Clearly, the assumptions incorporated into the governing equations of any mathematical model influence the choice of a numerical technique. For example, White et al.\textsuperscript{18} assume straight current lines; therefore their partial differential equations are parabolic. This allows them to use a straightforward technique, in which they use implicit stepping in the $x$ direction, and solve the boundary-value problem in the $y$ variable by Newman's method.\textsuperscript{28,57}

Although the choice of a numerical technique is influenced by the nature of the equations, the technique is not uniquely determined because there is often a choice of methods. In the present model, for example, Laplace's equation for the potential is not parabolic, and the implicit stepping technique cannot be used. White et al.\textsuperscript{18} approach this problem by using an implicit alternating direction (IAD) technique. Here, we approach the problem with two methods, both of which use the known analytic solution to Laplace's equation. Our methods, however, cannot be used to solve White's non-linear equations with migration and homogeneous reactions.

The first numerical method to be discussed here is similar to that of Parrish and Newman,\textsuperscript{21} and the second (see appendix D) is similar to the orthogonal collocation
method used by Cabán and Chapman. The motivation for both approaches is to take advantage of the linearity of the governing equations by using superposition integrals for the solution of Fick’s second law of diffusion. The power of the superposition technique is that it reduces the original two-dimensional problem to a one-dimensional problem. (The $y$-dependence is eliminated.) Note that although the detailed concentration profiles within the channel are not obtained, one can still obtain the cup-mixing average concentration (or conversion) at the exit, or any other axial position, by performing a material balance.

Parrish and Newman’s scheme is to alternate between revising the surface concentrations and the potential distribution. The details of our procedure are slightly different from those of Parrish and Newman, however, because the electrodes cannot be treated separately, and because there are multiple reactions.

The procedure now contains four nested iterative loops to solve the equations in table 1-1: the innermost loop takes a given overpotential distribution and solves the flux equations for the anode and cathode surface concentrations and currents at each axial mesh point; the second loop chooses the integration constant from equation 1-5 to satisfy the condition that the total currents are equal on the anode and cathode; the next loop finds the average current density; and the outermost loop iterates on the potential distribution until the distribution used to calculate surface concentrations agrees with that calculated from the solution to Laplace’s equation. The loop for average current density is not needed in principle, but it was added to promote stability in the iterations.

Figures 1-3 and 1-4 show the iterative scheme summarized above. Figure 1-3 is a detailed picture of the order for solving the governing equations, and figure 1-4 shows how the potential distributions are damped between iterations.

Before examining figure 1-3 in detail, one must first understand the physical
Figure 1-3. Scheme for calculating a new potential distribution.
Figure 1-4. Damping scheme.
significance of the integration constant, $\Phi^*$, and the average current density, $<i>$. Note that in equation 1-5, if one subtracts $\Phi^*$ from both sides and divides through by $<i>$, the left side is $(\Phi^0 - \Phi^*)/<i>$, which represents the shape of the potential distribution; the right side shows how this shape depends on the shape of the current distributions, $i(x)/<i>$. The integration constant, $\Phi^*$, in equation 1-5 sets the relative amounts of overpotential on the anode and cathode. For example, if $\Phi^*$ is too close to $V_{an}$, the total overpotential $(V - \Phi^*)$ on the anode will be too small, and the integral of $i_{an}(x)$ will be less in magnitude than the integral of $i_{cath}(x)$ (see figure 1-5). Thus, $\Phi^*$ is adjusted to satisfy the condition that the currents be equal on the two electrodes. The average current, $<i>$, sets the magnitude of the overpotentials. For example, if $<i>$ is too high, the overpotentials calculated with the $\Phi^*$ from equation 1-5 will be small, and, hence, the currents calculated from these overpotentials will be too small to agree with the input $<i>$ (see figure 1-6). In summary, $(\Phi^0 - \Phi^*)/<i>$ describes the shape of the potential distribution, $\Phi^*$ determines the placement of $\Phi_{an}$ and $\Phi_{cath}$ relative to the electrode potentials, and $<i>$ sets the magnitude of $(V - \Phi^*)$.

We can now discuss the entire scheme of figure 1-3 in detail. Proceeding from top to bottom on figure 1-3, first we make a guess for the potential distributions and for $\Phi^*$ and $<i>$. Recall that the electrode potentials, $V_{an}$ and $V_{cath}$, are specified. The potential distributions $\Phi_{an}(x)$ and $\Phi_{cath}(x)$ are then substituted into the definition of surface overpotential (1-17b with 1-18). Equation 1-17b is, in turn, substituted into the Butler-Volmer kinetic expression (1-15 with 1-16) to give each of the partial current densities, $i_j$. The partial current densities, $i_j(x)$, are then used in the Faraday's-law flux expression (1-13). The resulting Faraday's-law expression, therefore, contains the guessed potential distribution and unknown surface concentrations (arising from the concentration dependence of the exchange current density). For convenience, the concentrations may be brought outside the exponentials. On the cathode, for example,
Figure 1-5. Effect of $\Phi^*$ on the potential distribution.
Figure 1-6. Effect of \(<i>\) on the potential distribution.
\[
N_{i,\text{cath}}(x) = -\sum_j \frac{s_{ij}}{n_j F} i_{o,j,\text{ref}} \prod_k \left( \frac{c_{k,j,\text{cath}}}{c_{k,\text{ref}}} \right) \rho_{ij} \exp \left[ \frac{\alpha_{ij} F}{RT} (V_{\text{cath}} - \Phi_{\text{cath}} - U_{j,\text{ref}}) \right] \\
+ \sum_j \frac{s_{ij}}{n_j F} i_{o,j,\text{ref}} \prod_k \left( \frac{c_{k,\text{cath}}}{c_{k,\text{ref}}} \right) q_{ij} \exp \left[ \frac{-\alpha_{ij} F}{RT} (V_{\text{cath}} - \Phi_{\text{cath}} - U_{j,\text{ref}}) \right], \quad (1-25)
\]

where \( U_{j,\text{ref}} \) is equation 1-18 with \( c_{i0} \) replaced by \( c_{i,\text{ref}} \),

\[
p_{kj} = \gamma_{kj} + \alpha_{s_k/\gamma_{kj}} s_{kj}, \quad (1-26a)
\]

and

\[
q_{kj} = \gamma_{kj} - \alpha_{s_k/\gamma_{kj}} s_{kj}. \quad (1-26b)
\]

The Fick’s-law flux expression, developed in chapter 2, is discretized by the method of Acrivos and Chambre, and it contains unknown local surface concentrations and the known upstream concentrations. By equating the Fick’s-law and Faraday’s-law expressions for the flux of each species on each electrode, one obtains, at each mesh point, a set of equations to be solved for the unknown surface concentrations, \( c_i(x) \). These equations are solved by a multi-dimensional Newton-Raphson method. The spacing of the mesh points is fine near the leading edge but coarse downstream, where the profiles are generally smooth (see section 1.4.2). To speed the convergence of the Newton-Raphson method, the initial guesses are made for the surface concentrations by extrapolating linearly from the previous two axial positions. Let it be emphasized that the unknowns solved for simultaneously are the concentrations only at the present mesh point. Also, the electrodes do not interact at the present mesh point; rather the interaction propagates downstream from the already-solved mesh points.

After solving for the surface concentrations, \( \Phi^* \) and \( <i> \) are adjusted. The \( <i> \) loop was added to promote stability in the outer potential-distribution loop. \( \Phi^* \) is adjusted by a one-dimensional search to satisfy the condition that the currents are equal.
on the two electrodes. \(<i>\) is likewise adjusted to ensure that the magnitude of the current calculated from the kinetic expression agrees with the current that was substituted into the solution to Laplace's equation (equation 1-5).

The new current distributions from these inner loops are then substituted into Laplace's equation to calculate a new potential distribution. With the new potential distribution, one can check for convergence by comparing with the original guess.

Two possible ways to iterate on the potential distribution are the multi-dimensional Newton-Raphson method and successive substitution with damping. The relative merits of these two methods are discussed in appendix D and references 60 through 62. For this complex problem, successive substitution was chosen. A damping factor averages the projected potential distribution with that from the previous iteration:

\[
\Phi^* = (\text{damp})\Phi_{\text{old}}^* + (1-\text{damp})\Phi_{\text{new}}^*.
\] (1-27)

The damping factor must be chosen carefully, because if the damping factor is too small, the potential distribution oscillates from iteration to iteration, but if it is too large, convergence is too slow or is never achieved at all. Since the program does not converge well when the same damping factor is used for all iterations, the damping factor must be adjusted as the iteration proceeds. The criterion for this adjustment is that the optimum damping factor minimizes the mean square difference between the new potential distribution obtained with the damping factor and the distribution that would result on the subsequent iteration if that damping were then removed. The computer program finds the optimum damping factor by choosing initially three damping factors and fitting a parabola to the resulting curve (sum-of-squares deviation vs. damping factor). When the computer program has found the optimum damping from the fitting function, it proceeds to the next iteration. The iteration continues until the sum-of-squares deviation is as small as desired.

The damping scheme summarized above is shown in figure 1-4. First, the projected
potential distribution is calculated by the procedure shown in figure 1-3. This procedure is denoted by dotted boxes in both figures 1-3 and 1-4. The calculated distribution is stored in \((\Phi^o - \Phi^*)/\langle i \rangle^{\text{save}}\).

Next, after checking for convergence, the optimum damping factor is sought. The lower part of figure 1-4 shows that a damping factor is applied to the saved distribution, and that damped distribution, denoted \((\Phi^o - \Phi^*)/\langle i \rangle^{\text{try}}\), is used as input to calculate \((\Phi^o - \Phi^*)/\langle i \rangle^{\text{new}}\). When the sum-of-squares difference between \((\Phi^o - \Phi^*)/\langle i \rangle^{\text{try}}\) and \((\Phi^o - \Phi^*)/\langle i \rangle^{\text{new}}\) is a minimum, the optimum has been found. When the optimum damping factor has been found, it is applied to the saved potential distribution, and the old potential distribution is updated.

Although a successful computer run can take more than 40 outer iterations and several hundred seconds on a CDC-7600 computer, the run time can be reduced to less than 100 seconds by using previous runs to provide the initial guesses for the potential distributions. This technique is particularly effective when performing a series of runs to study the effects of a given parameter.

Here, we shall briefly mention the second method for solving the equations of section 1.2; more details are given in appendix D (section D.5). The technique, which is similar to the collocation method of Cabán and Chapman,\textsuperscript{22} is to express the partial current densities as superpositions of trial functions and to solve for the coefficients by a Newton-Raphson method.

1.4.1. Advantages and Disadvantages of the Two Solution Techniques

One advantage of the first scheme, particularly as originally conceived, is that the potentials are calculated by successive substitution instead of by the more time-consuming Newton-Raphson iteration. In addition, it is not necessary to invent good approximating functions and choose appropriate collocation points.
The disadvantages of the Parrish and Newman scheme are that it is time-consuming and convergence is hardly guaranteed. Although the damping-factor adjustment scheme is of some help, the program often finds that the optimum damping is complete damping, so that no movement is made toward the answer.

Another problem is that it is difficult to evaluate accurately the integrals for Laplace's equation because one must interpolate to find the currents at the quadrature roots (Parrish and Newman\textsuperscript{21} found that Simpson's rule was less accurate than Gaussian quadrature with Lagrange interpolation). This problem is particularly acute for large aspect ratios \((L/h)\) because the current decays rapidly in the edge region, making the interpolation difficult. In spite of the small size of the edge region, it plays an important role in determining the potential distribution, both locally and globally.

The collocation method does not share the problem of inaccurate interpolation because known current functions are used. The main advantage of the collocation method, however, is that the time-consuming integral equations (Laplace's equation and the superposition integrals) are evaluated only once at the start of the calculations instead of repeatedly inside the iteration loops. This savings is possible because of the linearity of the equations.

The major disadvantage of the collocation method is that the accuracy of the results depends on the choice of trial functions and the placement of collocation points; it is not sufficient to use a simple collection of orthogonal polynomials. Another disadvantage is that limiting-current problems must be treated separately and in a more complex manner (see appendix D).

1.4.2. Mesh Spacing

The superposition method described above is most easily carried out for evenly-spaced mesh points. Evenly-spaced points are unsuitable, however, because the concen-
tration profiles can change rapidly near the leading edge, but are smooth away from the leading edge. Therefore the mesh points should be finely spaced near the leading edge, but coarse downstream to avoid unnecessary calculations. For some problems, this difficulty is overcome by using, for example, logarithmically-spaced mesh points. Since this problem has integral equations, it is easiest to use the method of Acrivos and Chambre\footnote{59} and keep the mesh points evenly spaced. Closely-spaced mesh points are obtained near the leading edge by dividing the interval into "decades" and using evenly-spaced points inside the decades.\footnote{63} For example, if the axial-position variable ranges between 0 and 1, one can solve the problem between 0 and 0.1 with, say, 51 mesh points. One then discards all the mesh points except those corresponding to \( x = 0, 0.02, 0.04, \ldots, 0.1 \), and proceeds to solve the problem from \( x = 0.1 \) to \( x = 1 \) with the remaining 45 mesh points. This idea can also be applied to the subdivisions to achieve even finer spacing near the leading edge. Note that the interval need not be divided into tenths; we found that it is best to divide the interval in half, because at \( x = 0.5 \), the profiles are usually smooth, and there will not be a large discontinuity resulting from a drastic change in mesh size.

1.4.3 Integration Methods

Several integrals must be evaluated numerically. These are the integrals for the potential distribution (equation 1-6), the average current density (1-19), and the discretized superposition integrals (from 1-9a and 1-9b). A different method is chosen for each integral, according to its behavior.

The integrals for the potential distribution are evaluated by eliminating the singularity at \( x = x' \) by a method similar to that of Kantorovich and Krylov\footnote{64} and stretching the coordinates to smooth the integrand at the electrode edges.

The method of Kantorovich and Krylov is to write
\[ \int_0^1 i_{\text{cat}}(x') \ln \tanh^2 \left( \epsilon(x-x') \right) \, dx' = i_{\text{cat}}(x) \int_0^1 \ln \tanh^2 \left( \epsilon(x-x') \right) \, dx' \]

\[ + \int_0^1 \left[ i_{\text{cat}}(x') - i_{\text{cat}}(x) \right] \ln \tanh^2 \left( \epsilon(x-x') \right) \, dx', \quad (1-28) \]

with

\[ \epsilon = \frac{\pi L}{2h}, \quad (1-29) \]

so that the integrand containing the unknown \( i_{\text{cat}} \) vanishes at \( x=x' \) (because the singularity is logarithmic). Since the \( \ln \tanh^2 \) integral is still difficult to evaluate, we add and subtract instead a term that can be integrated analytically, but behaves like \( \ln \tanh^2 \) near the singularity:

\[ \int_0^1 i_{\text{cat}}(x') \ln \tanh^2 \left( \epsilon(x-x') \right) \, dx' = i_{\text{cat}}(x) \int_0^1 \ln \left( \epsilon(x-x') \right)^2 \, dx' \]

\[ + \int_0^1 \left[ i_{\text{cat}}(x') \ln \tanh \left( \epsilon(x-x') \right) - i_{\text{cat}}(x) \ln \left( \epsilon(x-x') \right)^2 \right] \, dx'. \quad (1-30) \]

The first term in this equation is integrated analytically and the second term by Gaussian quadrature with stretching of the coordinates (to be discussed shortly).

The \( \ln \cosh^2 \) term from equation 1-6 is also evaluated by Gaussian quadrature, but a correction term is added to ensure that the currents are exactly equal on the two electrodes:

\[ \int_0^1 \left[ i_{\text{cat}}(x') + i_{\text{an}}(x') \right] \ln \cosh^2 \left( \epsilon(x-x') \right) \, dx' \]

\[ = \int_0^1 \left[ i_{\text{cat}}(x') + i_{\text{an}}(x') \right] - \left< i_{\text{cat}} + i_{\text{an}} \right> \ln \cosh^2 \left( \epsilon(x-x') \right) \, dx' \]

\[ = \int_0^1 \left[ i_{\text{cat}}(x') + i_{\text{an}}(x') \right] \ln \cosh^2 \left( \epsilon(x-x') \right) dx' \]

\[ - \left< i_{\text{cat}} + i_{\text{an}} \right> \int_0^1 \ln \cosh^2 \left( \epsilon(x-x') \right) dx', \quad (1-31) \]

where
\[
< i_{\text{cat}} + i_{\text{an}} > = \int_0^1 [i_{\text{cat}}(x') + i_{\text{an}}(x')] \, dx'.
\]

(1-32)

Note that \(< i_{\text{cat}} + i_{\text{an}} >\) is zero in principle, but in practice it is equal to a convergence tolerance, such as \(10^{-6}\).

The stretching of the coordinates mentioned earlier is to let

\[
Y = (x')^n
\]

(1-33)

so that the integral becomes

\[
\int_{x_{\min}}^{x_{\max}} i(x') f(x - x') \, dx' = \int_{(x_{\min})^n}^{(x_{\max})^n} i(x') Y^{1/n - 1} f(x - x') \, dY.
\]

(1-34)

The choice \(n = 1/2\) eliminates the \(x^{-1/2}\) singularity in \(i(x)\) for the primary distribution. Even for more uniform current distributions, the integrand \(i(x') \sqrt{x'} f(x - x')\) is well-behaved. A similar stretching is done to eliminate possible singularities near \(x=1\). The integral is broken up into regions, and the appropriate stretching is used in each one.

The integrals for the average current density are evaluated, using the finely-spaced mesh points, by Simpson's rule. The integral from the first mesh point to the second mesh point is treated separately to give more accurate results near the limiting current. In principle, the limiting current density is infinity at \(x=0\) and decays as \(x^{-1/3}\) (as long as the boundary layer is still thin). To avoid sensitivity to the large, somewhat arbitrary value of \(i_{\text{cat}}\) at the first mesh point \((jx=1)\), the integral is evaluated as \(1.5h \, i_{\text{cat}}(jx=2)\), where \(h\) is the mesh spacing. This form is accurate at the limiting current but will overestimate the integral, by a maximum of 50\%, for more uniform current distributions. Recall, however, that this formula is applied only between the first and second mesh points, which are finely spaced.

The final integrals to be mentioned are the discretized superposition integrals. These are the most straightforward, since most of them can be evaluated analytically. Those that require numerical integration are integrated with the trapezoidal rule.
because the discretized integrals are between adjacent mesh points.

1.5. Summary

A superposition-integral technique is used to solve for the concentration, current, and potential distribution in a thin-gap channel flow cell. The superposition technique takes advantage of the linearity of the governing equation by reducing the problem from two-dimensional to one-dimensional. The model includes multiple electrochemical reactions and interacting diffusion boundary layers but does not include migration, axial diffusion, homogeneous reactions, or turbulent flow.

The next chapter develops the solutions that are needed for the superposition integral, and chapter 3 presents theoretical results from the model.
### Nomenclature for Chapter 1

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>flow cross-sectional area, cm$^2$</td>
</tr>
<tr>
<td>$B$</td>
<td>channel halfwidth, cm</td>
</tr>
<tr>
<td>$c_{i,an}$</td>
<td>anode surface concentration of species $i$, mol/cm$^3$</td>
</tr>
<tr>
<td>$c_{i,cath}$</td>
<td>cathode surface concentration of species $i$, mol/cm$^3$</td>
</tr>
<tr>
<td>$c_{i,ref}$</td>
<td>reference concentration of species $i$ (see equation 1-12), mol/cm$^3$</td>
</tr>
<tr>
<td>$c_{io}$</td>
<td>surface concentration of species $i$, mol/cm$^3$</td>
</tr>
<tr>
<td>$C_p$</td>
<td>heat capacity, cm$^2$/s$^2$-K</td>
</tr>
<tr>
<td>$d_1$</td>
<td>inner radius of annulus, cm</td>
</tr>
<tr>
<td>$d_2$</td>
<td>outer radius of annulus, cm</td>
</tr>
<tr>
<td>$d_e$</td>
<td>equivalent diameter (2$h$ for a channel), cm</td>
</tr>
<tr>
<td>$D$</td>
<td>diffusion coefficient, cm$^2$/s</td>
</tr>
<tr>
<td>$D_i$</td>
<td>diffusion coefficient of species $i$, cm$^2$/s</td>
</tr>
<tr>
<td>$damp$</td>
<td>damping factor (see equation 1-23)</td>
</tr>
<tr>
<td>$F$</td>
<td>Faraday’s constant, 96,487 C/eq</td>
</tr>
</tbody>
</table>
$G_z$  Graetz number (see equation 1-18 and discussion following)

$h$  interelectrode gap thickness, cm

$i(x)$  local current density, A/cm$^2$

$<i>$  average current density, A/cm$^2$

$i_j$  current density due to reaction $j$, A/cm$^2$

$i_{e,j}$  exchange current density for reaction $j$, evaluated at the local surface concentrations, A/cm$^2$

$i_{o,j,ref}$  exchange current density for reaction $j$, evaluated at the reference concentrations, A/cm$^2$

$J_j$  dimensionless exchange current density defined in equation 1-16

$J_{x}$  mesh point

$k$  thermal conductivity, g-cm/s$^2$-K

$L$  electrode length, cm

$N$  dimensionless parameter defined in equation 1-17 (represents dimensionless limiting current when boundary layers are thin)

$N_i$  flux of species $i$, mol/cm$^2$-s

$n_j$  number of electrons transferred in reaction $j$

$p_{kj}$  anodic reaction order for species $k$ in reaction $j$
\( q_{jk} \) cathodic reaction order for species \( k \) in reaction \( j \)

\( R \) universal gas constant, 8.3143 J/mol-K

\( Re \) Reynolds number \((2\langle v \rangle h/\nu \) for a channel\)

\( s_{ij} \) stoichiometric coefficient for species \( i \) in reaction \( j \)

\( Sc \) Schmidt number \((\nu/D)\)

\( T \) absolute temperature, K

\( U_j^\theta \) standard electrode potential for reaction \( j \), V

\( U_{j,s} \) theoretical open-circuit potential for reaction \( j \) relative to a reference electrode of a given kind, evaluated at the surface concentrations, V

\( U_{j,\text{ref}} \) theoretical open-circuit potential evaluated with reference concentrations, V

\( \langle v \rangle \) average fluid velocity, cm/s

\( V \) electrode potential, V

\( w \) mass flow rate, g/s

\( x \) axial coordinate, cm

\( y \) normal coordinate, cm or variable defined in equation 1-26b

\( z \) dummy variable of integration in equation 1-25
Greek

$\alpha_{a,j}$ anodic transfer coefficient for reaction $j$

$\alpha_{c,j}$ cathodic transfer coefficient for reaction $j$

$\gamma_{ij}$ exponent for concentration dependence of exchange current for reaction $j$ on species $i$

$\epsilon$ $\pi L/(2h)$

$\zeta$ dimensionless axial coordinate defined in equation 1-8

$\eta_{a,j}$ surface overpotential for reaction $j$, V

$\theta$ dimensionless concentration defined in equation 1-7

$\kappa$ ratio of inner to outer radius of annulus

$\kappa_\infty$ conductivity of feed solution, ohm$^{-1}$cm$^{-1}$

$\Phi$ potential in the solution as measured by a reference electrode of a given kind, V

$\Phi^o$ potential in solution just outside diffuse double layer, V

$\Phi^*$ integration constant in equation 1-3, V

$\rho_o$ pure solvent density, kg/cm$^3$

$\xi$ dimensionless normal coordinate defined in equation 1-9
### Subscripts

- **an**  
  anode

- **cath**  
  cathode

- **f**  
  feed

- **g**  
  reference electrode of a given kind

- **i**  
  species in solution

- **j**  
  electrode reaction

- **k**  
  dummy index for species

- **m**  
  main reaction

- **o**  
  electrode surface

- **R**  
  principal reactant

- **r1**  
  reference electrode positioned just outside the double layer

- **re**  
  reference electrode reaction

- **ref**  
  reference concentrations at which exchange current density $i_{oj,ref}$ was measured

- **s**  
  reference electrode of the same kind as the working electrode

- **∞**  
  feed solution
Superscripts

new \textsuperscript{new} \textsuperscript{new} \textsuperscript{new} \textsuperscript{new} variables calculated from damped ("try") variables, see figure 1-4

old \textsuperscript{old} \textsuperscript{old} \textsuperscript{old} \textsuperscript{old} variables from the previous iteration

save \textsuperscript{save} \textsuperscript{save} \textsuperscript{save} \textsuperscript{save} variables calculated from "old" variables

try \textsuperscript{try} \textsuperscript{try} \textsuperscript{try} \textsuperscript{try} weighted average of "old" and "save" variables

o \textsuperscript{o} \textsuperscript{o} \textsuperscript{o} \textsuperscript{o} electrode surface

\ast \textsuperscript{\ast} \textsuperscript{\ast} \textsuperscript{\ast} \textsuperscript{\ast} integration constant

\theta \textsuperscript{\theta} \textsuperscript{\theta} \textsuperscript{\theta} \textsuperscript{\theta} unit molality standard state
2. The Asymmetric Graetz Problem in Channel Flow

2.1. Introduction

The problem of mass transfer to fluids in laminar flow in ducts appears in many engineering applications. This problem has been solved for some special cases, but here we shall consider the case of a flat duct, or channel, where the surface-concentration boundary conditions are arbitrary, and may differ on the two channel walls. This problem is of interest when the channel gap is thinner than the diffusion-boundary-layer thickness. In such cases, the fluxes at the two channel walls are not independent.

Since the detailed concentration profile within the flowing fluid is generally not needed, we shall emphasize the distribution of flux along the channel walls. Given this flux distribution, one can calculate the average concentration at the exit by performing an overall material balance.

To obtain the distribution of flux along the two channel walls, Duhamel's superposition principle can be used to treat the nonlinear concentration boundary conditions.

2.2. Problem Statement

For laminar flow in a channel, with negligible axial diffusion, the dimensionless convective diffusion equation is

\[(1-\xi^2) \frac{\partial \theta}{\partial \xi} = \frac{\partial^2 \theta}{\partial \xi^2}, \tag{2-1}\]

where

\[\theta = \frac{c_i-c_\xi}{c_o-c_\theta} \tag{2-2}\]

\[\xi = \frac{y}{B} \tag{2-3}\]
\[ \zeta = x \frac{D_i}{\frac{3}{2} B^2 \langle v_x \rangle} \]  
(2-4)

If the boundary conditions are arbitrary, Duhamel's superposition theorem can be used to write the flux in terms of the solution to the problem with a step-function concentration boundary condition on one wall. For example, if wall "-1" is located at \( \xi = -1 \), and wall "1" is located at \( \xi = 1 \), then the flux of species \( i \) at wall "-1" is

\[ N_{i,-1}(x) = -\frac{D_i}{B} \int_0^x \left. \frac{dc_{i,-1}}{dx} \right|_{\xi = \zeta} \frac{\partial \theta}{\partial \zeta} \left( \xi = \zeta, \xi = -1 \right) dx \]  
(2-5)

\[ + \frac{D_i}{B} \int_0^x \left. \frac{dc_{i,1}}{dx} \right|_{\xi = \zeta} \frac{\partial \theta}{\partial \zeta} \left( \xi = \zeta, \xi = 1 \right) dx, \]

where \( c_{i,-1} \) is the surface concentration of species \( i \) at \( \xi = -1 \) and \( c_{i,1} \) is the concentration of species \( i \) at \( \xi = 1 \). The flux (in the +\( \xi \) direction) of species \( i \) at wall "1" is obtained by reversing the wall subscripts and the signs. In equation 2-5, \( \theta(\zeta, \xi) \) is the solution to equation 2-1 with boundary conditions

\[ \theta = 1 \text{ at } \xi = -1 \]  
(2-6)

\[ \theta = 0 \text{ at } \xi = 1 \]  
(2-7)

\[ \theta = 0 \text{ at } \zeta = 0 \]  
(2-8)

2.3. The Lévêque Approach

In general, equation 2-1 must be solved numerically. If, however, the diffusion boundary layers are thin, an analytic solution for the flux can be obtained by assuming that the velocity profile is linear throughout the boundary layer. This approximation, known as the Lévêque approximation,\(^{28,30,31,65}\) is not valid throughout a thin-gap channel, but it is useful for treating the entrance region, where the diffusion boundary layers are thin. Norris and Streid have solved this Lévêque problem for channel flow.\(^{68}\)

One can extend the range of applicability of the Lévêque solution by writing a Lévêque series for the Nusselt number based on the equivalent diameter. This has been
done for the Graetz problem in a tube,\textsuperscript{30,65} and a similar procedure can be used for a channel to give

\[ Nu = -4 \frac{\partial \theta}{\partial \xi} \bigg|_{\xi = -1} = 2.7131949\xi^{-1/3} - 0.4 - 0.1214690\xi^{1/3} \]  \hspace{1cm} (2-9)

(see appendix A).

2.4. The Graetz Approach

To treat the downstream region, the Graetz approach (separation of variables) should be used. To calculate the dimensionless concentration,

\[ \theta = \frac{1}{2} - \frac{\xi}{2} + \sum_{k=1}^{\infty} a_k e^{-\lambda_k \xi} Y_k(\xi), \]  \hspace{1cm} (2-10)

the coefficients \(a_k\), the eigenvalues \(\lambda_k\), and the eigenfunctions \(Y_k\) are needed. The coefficients are obtained by using the orthogonality of the eigenfunctions with respect to the weighting function \((1-\xi^2)\). The eigenvalues and eigenfunctions must be obtained numerically.

2.5. Solution of the Eigenvalue Problem

One can rewrite the eigenvalue problem in a form convenient for numerical solution by realizing that the eigenvalues \(\lambda\) are constant. Thus,

\[ Y'' + \lambda^2 (1-\xi^2) Y = 0 \]  \hspace{1cm} (2-11)

\[ \frac{d\lambda^2}{d\xi} = 0. \]  \hspace{1cm} (2-12)

To solve equations 2-11 and 2-12, three boundary conditions are needed. The first two boundary conditions result from equations 2-6 and 2-7. The third boundary condition is a normalization condition

\[ Y' = 1 \text{ at } \xi = 1. \]  \hspace{1cm} (2-13)

The system of two ordinary differential equations with the three boundary conditions can be solved numerically using a finite-difference technique.\textsuperscript{30,57} The computed
eigenfunctions are shown in figure 2-1.

2.6. Combining the Graetz and Lévêque Solutions

It should be noted that a truncated Graetz series (obtained from equation 2-10) is accurate for large $\zeta$, while the Lévêque series (equation 2-9) is valid for small $\zeta$. The value of $\zeta$ that divides the two regions is that value at which the ratio of the two asymptotic solutions is closest to unity. For example, if three terms are used in each series, the maximum error in the Nusselt number is $0.48\%$ at $\zeta=0.11$.

2.7. Asymptotic Forms for Large Eigenvalues

If greater accuracy than $0.48\%$ is desired, then it is most efficient to add terms to the Graetz series. Therefore, it is useful to have simple asymptotic forms for calculating the higher eigenvalues and corresponding coefficients.

For the Graetz problem in a tube, Newman$^{30}$ extended the asymptotic forms of Sellars et al.$^{67}$ to achieve accuracy over a greater range of eigenvalues. Using a similar procedure, we devised an asymptotic form for the asymmetric Graetz problem:

$$\lambda = \lambda_0 + \frac{0.03254}{\lambda_0^{4/3}} - \frac{0.11}{\lambda_0^{14/3}},$$

(2-14)

where

$$\lambda_0 = \frac{6k - 1}{3}, \quad k=1, 2, \ldots$$

(2-15)

is the asymptotic form obtained by modifying the method of Sellars et al. (see reference 71). The function in equation 2-14 can be used for $\lambda_6, \lambda_7, \ldots$, with a maximum error of $10^{-5}\%$.

The method of Sellars et al. predicts that the coefficients behave as $A_k = (-1)^{k+1}K/\lambda_k^{1/3}$ as $\lambda_k$ becomes large, where
Figure 2-1. Eigenfunctions for the asymmetric Graetz problem
This asymptotic form was modified to

\[ A_k = (-1)^k \left( \frac{K}{\lambda_k^{1/3}} \right) \left( 1 + 0.03 \lambda_k^{-4/3} - 0.03 \lambda_k^{-8/3} \right). \]  

Equation 2-17 gives a maximum error of 4\times10^{-4}\% for \( k \geq 6 \).

Table 2-1 shows the comparison between the eigenvalues and coefficients as calculated by solving the eigenvalue problem and those calculated from the asymptotic form in equations 2-14 and 2-17. The accurate results of Brown\textsuperscript{66} are also shown.
Table 2-1
Eigenvalues and Coefficients for the Asymmetric Graetz Problem

<table>
<thead>
<tr>
<th>k</th>
<th>Brown</th>
<th>Calculated</th>
<th>Asymptotic</th>
<th>Brown</th>
<th>Calculated</th>
<th>Asymptotic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.6815953222</td>
<td>1.6815953</td>
<td>1.6729924</td>
<td>0.8580866740</td>
<td>0.85808668</td>
<td>0.85807016</td>
</tr>
<tr>
<td>2</td>
<td>3.6722904</td>
<td>3.6721659</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5.6698573459</td>
<td>5.6698540</td>
<td>5.6698540</td>
<td>0.5694628499</td>
<td>0.56946285</td>
<td>0.56949143</td>
</tr>
<tr>
<td>4</td>
<td>7.6688088</td>
<td>7.6688110</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>9.6682424625</td>
<td>9.6682424</td>
<td>9.6682441</td>
<td>0.4760654555</td>
<td>0.47606547</td>
<td>0.47606966</td>
</tr>
<tr>
<td>6</td>
<td>11.667894</td>
<td>11.667895</td>
<td></td>
<td>-0.44701873</td>
<td>-0.44702059</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>13.6676614426</td>
<td>13.667662</td>
<td>13.667662</td>
<td>0.4239737298</td>
<td>0.42397373</td>
<td>0.42397459</td>
</tr>
<tr>
<td>8</td>
<td>15.667496</td>
<td>15.667496</td>
<td></td>
<td>-0.40504973</td>
<td>-0.40505013</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>17.6673735653</td>
<td>17.667374</td>
<td>17.667374</td>
<td>0.3891087061</td>
<td>0.38910871</td>
<td>0.38910889</td>
</tr>
<tr>
<td>10</td>
<td>19.667279</td>
<td>19.667280</td>
<td></td>
<td>-0.37541429</td>
<td>-0.37541436</td>
<td></td>
</tr>
</tbody>
</table>

By using these asymptotic forms in a Graetz series with many terms, one can achieve a high degree of accuracy in the Nusselt number.

2.8. Empirical Approach

Recall that the Léveque solution is applicable only on the wall with the step change in concentration. To fit the region of small $\zeta$ on the opposite wall, it is more convenient to use an empirical function than it is to use many terms in the Graetz series.

The empirical function used here was derived by considering a simpler problem. If the fluid were in plug flow, rather than laminar flow, the mass-transfer problem would
be analogous to the problem of transient heat conduction in a finite slab, where short time, \( t \), is analogous to small \( \xi \).

Based on the short-time solution for heat conduction in a finite slab, it is assumed that \( Nu \) is proportional to \( e^{-\lambda/\xi} \). To match the behavior at low \( 1/\xi \), a correction term of the form \( ce^{-d/\xi} \) can be added. The constants \( c \) and \( d \) were obtained by performing a least squares fit between \( \ln Nu \) as calculated from a 100-term Graetz series and \( \ln Nu = a - b/\xi - ce^{-d/\xi} \), for various specified values of \( d \). The least squares fit was designed to weight the region of small \( \xi \), where the empirical function is to be used. The resulting fitting function for small \( \xi \) is

\[
Nu = -4 \frac{\partial \theta}{\partial \xi} \bigg|_{\xi=1} = 2 \exp(0.9594 - 0.6069 \frac{1}{\xi} - 0.4512e^{-0.276/\xi}) .
\] (2-18)

At \( \xi = 0.18 \), the error between the 3-term Graetz series and the empirical function is 0.013%.

### 2.9. Summary of Results

In summary, a convenient representation of the Nusselt number for the asymmetric Graetz problem has been obtained. For the wall with the step change in concentration, equation 2-9 is used for \( \xi < 0.11 \) and

\[
Nu = 2 + 4 \sum_{k=1}^{3} |A_k| e^{-\lambda_k \xi} \quad \text{for } \xi \geq 0.11,
\] (2-19)

where the eigenvalues and coefficients are listed in table 2-1. For the wall without the step change, equation 2-18 is used for \( \xi < 0.18 \) and

\[
Nu = 2 - 4 \sum_{k=1}^{3} A_k e^{-\lambda_k \xi} \quad \text{for } \xi \geq 0.18.
\] (2-20)

It has been proposed that this solution can be used in a superposition integral to determine the wall flux in problems where the channel wall concentrations are arbitrary and may differ on the two walls.
Nomenclature for Chapter 2

$A$ coefficients for the asymmetric Graetz problem

$B$ channel halfwidth, cm

$c_b$ bulk (feed) concentration of species $i$, mol/cm$^3$

$c_i$ concentration of species $i$, mol/cm$^3$

$c_{i,-1}$ concentration of species $i$ at the wall located at $\xi = -1$, mol/cm$^3$

$c_s$ surface concentration of species $i$, mol/cm$^3$

$D_i$ diffusion coefficient of species $i$, cm$^2$/s

$K$ constant defined in equation 2-16

$N_{i,-1}$ flux of species $i$ at the wall located at $\xi = -1$ (flux is positive in the $+\xi$ direction), mol/cm$^2$-s

$Nu$ Nusselt number (see equation 2-9)

$<v_x>$ average fluid velocity in the $x$-direction, cm/s

$x$ axial coordinate, cm

$y$ normal coordinate, cm
eigenfunctions for the asymmetric Graetz problem

Greek

Γ  gamma function

ζ  dimensionless axial coordinate defined in equation 2-4

θ  dimensionless concentration defined in equation 2-2

λ  eigenvalues for asymmetric Graetz problem

λ₀  asymptotic form for large eigenvalues

ξ  dimensionless normal coordinate defined in equation 2-3

Subscripts

b  bulk

i  species in solution

k  index for eigenvalues and eigenfunctions

o  surface

x  axial direction

-1  wall located at ξ = -1

0  asymptotic form for large eigenvalues

1  wall located at ξ = 1
Superscripts

* dummy variable of integration

\' differentiation with respect to \( \xi \)
3. Thin-Gap-Model Results

3. Introduction

Mathematical models are useful for predicting the performance of complex systems under a variety of conditions, without having to do repeated experiments. While simplified models and dimensional analysis are useful for quick calculations, they cannot predict accurately the interaction of several phenomena.

The complex model discussed in chapters 1 and 2 predicts the total current and the distribution of current, concentration, and potential adjacent to the electrode surfaces in a channel flow cell. In addition, the conversion of a given reactant may be obtained, by a material balance, from the current distributions. One application for the model is to assess some of the economic tradeoffs associated with thin gaps. For example, thin gaps reduce the cell resistance, but increase the "chemical shorting" in the cell, where a product from one electrode diffuses across the cell and reacts back to the starting material. Another use for the model is to provide information that is difficult to obtain experimentally, such as the distributions of current, concentration, and potential. Current distribution can be measured experimentally by a sectioned-electrode technique or by measuring plated deposits, but it is difficult to measure potentials and concentrations in the solution adjacent to the electrodes.

We introduce this chapter with a discussion of the three limiting cases of current distribution and the results of other models and conclude with a discussion of the results from the thin gap model both for a plating and dissolution reaction (section 3.2) and for a redox reaction (section 3.3). Chapter 5 presents a comparison between theoretical and experimental results for the \( \text{Fe(CN)}_6^{3-}/\text{Fe(CN)}_6^{4-} \) redox couple.
3.1. Limiting Cases of Current Distribution

Before examining previous laminar-flow models, it is useful to understand the three limiting cases of current distribution in any electrochemical system. These current distributions are called primary, secondary, and mass-transfer-limited current distributions. The primary distribution results when there are no kinetic or mass-transfer limitations. Only the ohmic potential drop in the solution governs the primary current distribution. If the current distribution is governed by ohmic potential drop and reaction kinetics, but not mass transfer, the distribution is called the secondary current distribution. If mass transfer alone dominates the current distribution, the electrochemical cell is at the "limiting current." That is, the current is limited by how fast the reactants can diffuse to the electrode. In general, all three effects — ohmic potential drop, electrode kinetics, and mass transfer — must be considered simultaneously.

The shapes of two of the limiting cases of current distribution in a channel flow cell are shown in figure 3-1. The primary current density is infinite at the ends of the electrodes, the secondary current density is similar to the primary, but finite everywhere, and the mass-transfer limited current density is high only at the leading edge of the electrode. The primary distribution is calculated by solving Laplace's equation for the potential with a constant potential adjacent to each electrode, and it is infinite at the edges because the current can flow out into the solution and collect back at the edges. (See figure 3-1.) The secondary distribution is similar to the primary, but because of kinetic limitations, the current cannot be infinite at the electrode edges. The shape of the secondary distribution generally depends on the geometry of the system, the kinetic parameters, and the average current density. Wagner has solved the secondary current distribution problem in a channel for large and small aspect ratios. Section 117 of Newman's book discusses the secondary current distribution for a disk elec-
PRIMARY CURRENT DISTRIBUTION

Fast kinetics, no mass-transfer limitations. (Ohmically dominated)

Constant potential near electrodes.

LIMITING CURRENT DISTRIBUTION

Mass-transfer limited.

Surface concentration is zero.

Figure 3-1. Limiting cases of current distribution.
trode, and the same qualitative conclusions apply to channel electrodes. If mass transfer alone dominates the current distribution, the current density is high only at the leading edge of the electrode, where there is undepleted solution.

3.1.1. Results for Thin Diffusion Boundary Layers

Parrish and Newman\textsuperscript{20,21} present current and concentration profiles for a single metal dissolution and deposition reaction. Their model includes ohmic potential drop, reaction kinetics, and mass transfer; however, in their earlier paper,\textsuperscript{20} they do not consider any interaction between the two electrodes. In the second paper,\textsuperscript{21} they account for interaction through the potential distribution, but not through the mass transfer. (They assume thin boundary layers.) Their results are particularly interesting below the limiting current, because the cathodic current distribution can have a maximum. Intuitively, one would expect that mass transfer is not important on the anode because the anodic dissolution reaction is not affected by the concentration of ions in the solution; therefore the current distribution should resemble the secondary current distribution, which is high at both edges. On the cathode, however, mass transfer begins to affect the current distribution in the downstream region, where the solution is depleted of metal ion; in the upstream region, the cathodic current distribution should behave like a secondary current distribution. If the downstream region is small enough, the cathodic current density can have a maximum arising from a compromise between the secondary current distribution and the mass-transfer-limited current distribution. The thin gap model reproduces these results for large Graetz numbers (\(\geq 100\)), as it should.

3.1.2. Results for Straight Current Lines

White \textit{et al.} present two multiple-reaction models, one that assumes straight current lines\textsuperscript{18} and another that includes axial migration and diffusion and time depen-
Both papers discuss a copper system in which Cu(I) reacts at both electrodes and Cu(II) is a product on the anode and a reactant on the cathode. In the first paper, they examine the copper system with excess supporting electrolyte and discuss the current efficiency, selectivity for copper deposition, and conversion per pass. They note that the important independent variables are $Pe \frac{h}{L}$ (which is proportional to the Graetz number defined in chapter 1) and the cell potential.

One can infer from the paper that the average current efficiency for copper deposition can be improved by designing an electrode whose potential varies with axial position. In the example they discuss, the reactions are

\[
\begin{align*}
\text{Cu(I)} \rightarrow \text{Cu(II)} & \quad \text{anode, reaction 1} \\
\text{Cu(II)} \rightarrow \text{Cu(I)} & \quad \text{cathode, reaction 2} \\
\text{Cu(I)} \rightarrow \text{Cu(0)} & \quad \text{cathode, reaction 3,} \\
\end{align*}
\]

and the concentration of Cu(I) is 0.5 M and the concentration of Cu(II) is 0.1 M. In this case, reactions 1 and 3 are claimed to be near limiting current, and reaction 2 is well below limiting current. The idea of designing the electrode with an axially-varying potential is to increase the driving force near the trailing edge so that reaction 2 can proceed faster there and provide Cu(I) to aid reaction 3.

White et al. also discuss the effects of $Pe \frac{h}{L}$ and cell potential on the selectivity of copper, which they define as the moles of Cu produced / moles of Cu(II) produced, and the conversion per pass of Cu(I). They note the existence of a case for which there is a mild maximum in the Cu selectivity vs. $Pe \frac{h}{L}$ and in the conversion per pass vs. $Pe \frac{h}{L}$.

The second paper discusses the effects of axial diffusion and migration and time dependence. The important results are that axial diffusion and migration may be neglected for $h/L < 0.5$ and that the effect of axial diffusion and migration is to smooth
3.2. Thin-Gap-Model Results for a Plating Reaction

In this section, we examine the effects of interacting boundary layers and of the axial component of current in a system with excess supporting electrolyte (no migration). Since the effects of interacting boundary layers are most pronounced when mass transfer is controlling, only limiting-current results will be discussed. It is interesting that, for thick diffusion boundary layers, there is still a limiting current, even though products from the anode can diffuse to the cathode and react.

For simplicity, we shall consider a single electrode reaction and a single species, for example, a metal dissolution reaction on the anode and the metal plating reaction on the cathode. We shall also simplify the interpretation of the results by considering a case with weak interaction of the electrodes through Laplace's equation \((h/L=1)\). If the electrodes interact through Laplace's equation, the anode current distribution begins to resemble the cathode current distribution, as shown by Parrish and Newman. Thus, the source of interaction discussed here will be the diffusion of reacting species across the cell gap. After discussing the effect of interacting boundary layers, we shall discuss the effects of the axial component of current. Note that our choice of a large \(h/L\) enhances the axial current, as shown by Nguyen et al.\(^1\)

Figure 3-2 shows a dimensionless plot of the anode current distribution for three diffusion-boundary-layer thicknesses. The parameter \(Pe \frac{d_e}{L}\) (which is proportional to the Graetz number\(^1\)) is a measure of the dimensionless gap thickness relative to the boundary-layer thickness. For large \(Pe \frac{d_e}{L} (\geq 100)\), for example high flow rates, the boundary layer is thin compared to the interelectrode gap. The current distributions shown in figure 3-2 resemble secondary

\(^1 Pe \frac{d_e}{L} = \frac{4}{\pi} G_z.\)
Figure 3-2. Effect of boundary-layer thickness on anode current distribution.
current distributions because there are no mass-transfer limitations on the anodic dissolution reaction and because the cathode does not have much influence through the potential distribution. \( h/L = 1 \). Therefore the anodic current distribution is not strongly affected by the boundary-layer thickness.

The cathode, on the other hand, is mass transfer limited, and therefore \( i_{\text{cath}} \) is strongly dependent on \( Pe \, d_e/L \). Figure 3-3 shows an example of the cathode current deviation from the thin-boundary-layer prediction. (If \( Pe \, d_e/L \) were 100 or higher, there would be no deviation, and the thin-boundary-layer approximation could be used.) In figure 3-3, the solid curve is the current density resulting from the hypothetical Graetz-Levêque problem, which has a step change in cathode surface concentration (from the feed concentration to zero) and a constant anode surface concentration \( (c_{an} = c_{\text{feed}}) \). This curve decays as \( x^{-1/3} \) near the leading edge, where the boundary layer is thin and levels off to a constant in the downstream region, where the concentration profile across the cell gap is linear (fully-developed mass transfer). The cathode current distribution predicted by the model is shown by the dotted line and decays as \( x^{-1/3} \) near the leading edge, where the boundary layers are thin, but increases again when the anode boundary layer touches the cathode. That is, the solution near the cathode is no longer depleted because metal ions produced at the anode diffuse across the gap to the cathode. Away from the edges of the cell, the cathode current density is low, roughly following the behavior of the anode. Near the trailing edge, the anode current density increases from ohmic effects, and the cathode current increases because some of the ions produced on the anode have diffused across the cell gap.

We have seen how the interaction of the diffusion boundary layers affects the anodic and cathodic current distributions at the limiting current. Again, it is interesting that there is still a limiting current when the boundary layers interact. To understand why there is a limiting current and to understand the anode surface concentration profiles,
Figure 3-3. Deviation from thin-layer approximation (normalized by average current from step-function problem).
one must examine the axial component of current.

Using the anodic and cathodic current distributions, one can calculate, by a 
material balance, the average current density flowing in the axial (fluid-flow) direction 
across a vertical cross-section at position $x$:

$$ I_{axial}(x) = \frac{1}{L} \int_0^L [i_{an}(x') + i_{cath}(x')] \, dx' $$

(3-2)

Thus, negative axial currents indicate that there has been more cathodic reaction up to 
the axial position $x$. (Cathode currents are negative.) Since the cathodic reaction 
depletes the solution of metal ions, the cup-mixing average concentration at the position $x$ is less than the feed concentration for negative axial currents. Figure 3-4 shows the axial current for $Pe \, d_e/L = 1$. Note that the axial current is zero at $x=0$ and $x=L$ because there is no net flow of current beyond the electrode edges. The positive spike in $I_{axial}$ near the leading edge results because the anode current density rises more rapidly 
toward the leading edge than the mass-transfer-limited cathode current density (approximately $x^{-1/2}$ vs. $x^{-1/3}$). Throughout most of the cell, the extra current produced at the 
downstream edge of the anode flows upstream before reaching the cathode. Note that 
this does not imply that metal ions diffuse upstream, because current flows by the 
migration of supporting-electrolyte ions.

The current and potential lines corresponding to figure 3-4 are shown in figure 3-5. 
The potential lines were calculated using the complete solution to Laplace’s equation:
Axial Current, Pe $d_e/L = 1$

Figure 3-4. Axial current distribution for a thick boundary layer.
Figure 3-5. Current and potential lines for a thick boundary layer.
\[ \Phi(x,y) = \Phi^* - \frac{1}{2\pi \kappa_{\infty}} \left\{ \int_0^L \int_{c_{\text{cath}}(x')} \ln \left[ \sinh^2 \left( \frac{\pi(x-x')}{2h} \right) + \sin^2 \left( \frac{\pi(y-h)}{2h} \right) \right] \, dx' \right\} + i_{an}(x') \ln \left[ \sinh^2 \left( \frac{\pi(x-x')}{2h} \right) + \sin^2 \left( \frac{\pi y}{2h} \right) \right] \right\}. \tag{3-3} \]

The current lines were plotted by starting at one electrode and stepping along a two-dimensional grid, numerically calculating the gradient of \( \Phi \) at each \( x \) or \( y \) mesh point. In principle, it should be possible to obtain the same current lines starting on either electrode. However, since the results were slightly different, an average was taken.

The figures show that the axial current is negative throughout most of the cell. Therefore, one would expect the electrolyte to be depleted of metal ion throughout most of the cell. One can examine the depletion of the electrolyte by following the anode surface concentration as a function of axial position. Since the surface concentration of metal ion is zero at the cathode (limiting current), the cup-mixing concentration is roughly equal to the feed concentration when the anode surface concentration is twice the feed concentration. Figure 3-6 shows the anode surface concentration as a function of axial position. Note that the spike in \( c_{an}/c_{\text{feed}} \) above 2 corresponds to the positive spike in axial current. Farther downstream, however, \( c_{an}/c_{\text{feed}} \) is less than 2, and the axial current is negative; therefore the solution is depleted.

The effect of boundary-layer thickness is also shown in figure 3-6. As the diffusion boundary layer becomes thicker (e.g. lower flow rate), the electrolyte becomes more depleted. In fact, the anode surface concentration can fall below the feed concentration, as shown in figure 3-6. Although the results for \( Pe h/L = 0.04 \) should be viewed with some caution, because the Péclet number is low enough that axial diffusion will smooth the current and concentration profiles, the low surface concentrations are still expected.
Figure 3-6. Effect of boundary-layer thickness on anode surface concentration.
We can now discuss the presence of a limiting current for interacting boundary layers. At first, one might think that, by increasing the cell potential, the anode would produce more ions that would diffuse over to the cathode and react, and that, therefore, there would not be a limiting current. However, this is not the case because if $c_{an}$ were much greater than $2c_{feed}$ everywhere, the axial current would be positive everywhere; therefore, the production of metal ions on the anode is limited by the requirement of no net current flow beyond $z=L$. Another way of thinking about this is that the ions produced on the anode must travel some distance downstream before reaching the cathode. The ratio of the electrode length $L$ to this distance is a rough indication of the amplification effect of ions reaching the cathode from the anode instead of just from the feed stream. This amplification factor can be large, but it is still finite, and there is still a limiting value for the current.

The effects of interacting diffusion boundary layers and of the axial component of current have been discussed for a single plating reaction. When $Pe \frac{d_e}{L} < 100$, the boundary layers interact and may not be considered thin. That is, if the gap is thin, the flow rate is low, the electrodes are long, or the diffusion coefficient is high, the electrodes must be treated simultaneously in the mathematical analysis.

An interesting result of the model is that there is a limiting current for thin gaps. Even though products from the anode can react on the cathode, the production of species on the anode is limited by the requirement of equal currents on the two electrodes.

The effect of axial currents is to deplete the electrolyte of metal ions throughout most of the cell. Thus, the two-dimensional structure of the potential distribution is important, and it should not be assumed that the current lines travel straight across
from one electrode to the other.\

3.3. Results for a Redox Reaction

We now examine results for a redox reaction in which there are two reacting species, such as ferri- and ferrocyanide ions. The cathodic reactant is denoted species "1" so that, on the cathode,

\[ M_1 + e^- \rightarrow M_2. \]  \hfill (3-4)

The two additional parameters that arise in changing from a plating to a redox system are the ratios \((c_1/c_2)_{feed}\) and \(D_1/D_2\). These parameters determine which of the two species is the limiting reactant.

For simplicity, we consider first cases in which \(D_1 = D_2\); then if \(c_2\) is greater than \(c_1\), the limiting reactant is species 1. If \(c_2 \gg c_1\), then \(c_1\) behaves like the single species in a plating/dissolution reaction; if \(c_2 = c_1\), then both species behave the same, but on opposite electrodes. These two limiting cases serve as a check of the computer program for redox systems.

Figure 3-7 shows the cathode current distribution for the two limiting cases. For \(c_2 \gg c_1\), the limiting reactant, \(c_1\), reacts at the cathode, so the cathode current density decays as \(x^{-1/3}\) (for high Graetz numbers). This current density also prevails on the cathode if \(c_1 = c_2\) because species 1 is still the limiting reactant. The anode current distribution, shown in figure 3-8, depends on the ratio \((c_2/c_1)_{feed}\) because the reactant, species 2, may or may not be limited. If it is limited, as in the case \(c_{2,feed} = c_{1,feed}\), the anode current decays as \(x^{-1/3}\); if it is not limited at all \((c_{2/c_1}feed = \infty)\), then the anode current resembles that of a single species. The intermediate values of \((c_2/c_1)_{feed}\) correspond to various fractions of limiting current on the anode. The maximum in the anode current distribution (for \(c_{2/c_1}feed = 1.2\) and 1.1 in figure 3-8) was also observed

---

\(^{1}\) Nguyen et al.\(^{19}\) suggest that straight current lines may be assumed for \(h/L < 0.5\).
Figure 3-7. Cathode current distribution for a redox system with equal diffusion coefficients ($h/L = 1$).
Figure 3-8. Anode current distribution for a redox system with equal diffusion coefficients ($h/L = 1$).
by Parrish for a single reactant,\textsuperscript{21} it occurs because the current begins to rise near the trailing edge from ohmic effects, but decreases again because the reaction becomes mass-transfer limited (species 2 is depleted).

The surface concentration profiles corresponding to figures 3-7 and 3-8 are shown in figures 3-9 and 3-10. Figure 3-9 shows the concentration of species 1 on the anode. For $c_{2,\text{feed}} > c_{1,\text{feed}}$, the concentration $c_{1,\text{an}}$ approaches $c_{an}$ for a single-species dissolution reaction; for $c_{2,\text{feed}} = c_{1,\text{feed}}$, the anode concentration steps up to $2c_{\text{feed}}$, reflecting the limiting-current behavior on the cathode, where $c_1$ steps down to zero. The intermediate curves level off at the trailing edge because of mass-transfer limitations. Figure 3-10 shows $(c_{2,\text{an}} - c_{2,\text{feed}})/c_{1,\text{feed}}$ to illustrate that the two species depart from their feed values by the same amounts, but in the opposite direction, on a given electrode.

We consider next an example with unequal diffusion coefficients (see references 28, 74, and 75). At high Graetz numbers, results from the Lévéque approximation apply. One such result is that if the concentration of species 1 drops to zero, then the concentration of species 2 on that electrode steps up to

$$c_2 = c_{2,\text{feed}} + (D_1/D_2)^{2/3} c_{1,\text{feed}}.$$  \hspace{1cm} (3-5)

(The same result with the subscripts reversed applies to the other electrode.) This high-Graetz-number result is obtained with the model, as shown in figures 3-11 and 3-12. The figures demonstrate the effects of the Graetz number and aspect ratio ($h/L$) for $(c_2/c_1)_{\text{feed}} = 1/4$ and $D_2/D_1 = 8$. At high Graetz numbers, the concentrations step to the values predicted by equation 3-5; at low Graetz numbers, however, the mass transfer becomes fully-developed, and the leading-edge surface concentrations cannot be maintained because of the requirement of equal current on the two electrodes. For fully-developed mass transfer, the surface concentrations are
Figure 3-9. Anode surface product concentration for a redox system with equal diffusion coefficients ($h/L = 1$).
Figure 3-10. Anode surface reactant concentration for a redox system with equal diffusion coefficients ($h/L = 1$).
Figure 3-11. Surface concentration profiles for a redox system with unequal diffusion coefficients. \(((c_2/c_1)_{\text{feed}} = 0.25, D_2/D_1 = 8, h/L = 0.01\).
Figure 3-12. Surface concentration profiles for a redox system with unequal diffusion coefficients. \((c_2/c_1)_{feed} = 0.25, D_2/D_1 = 8, h/L = 0.1\).
and the concentration profiles are linear across the cell gap. Note that these concentrations satisfy the condition of equal currents on the two electrodes. It is the ratio of diffusion coefficients that determines, for a given feed composition, which is the limiting species (species 1, in this example).

An interesting feature of figures 3-11 and 3-12 is that $c_{2,an}$ clearly shows the Lévêque (thin-boundary-layer) region and the fully-developed-mass-transfer region; in the Lévêque region, $c_{2,an} \approx 0$ and in the developed region, $c_{2,an} \approx \frac{1}{8} c_{1,feed}$. Also shown in the figures is the effect of $h/L$. One would expect that for small $h/L (<10)$ the electrodes would interact through the potential distribution\(^{21}\) and that for very small $h/L$, both electrodes would behave like the limiting electrode. Parrish and Newman demonstrated this effect for a plating reaction.\(^{21}\) Figures 3-11 and 3-12 also show this effect; for the small aspect ratio, $h/L = 0.01$, the anode is closer to the limiting current than for $h/L = 0.1$ ($c_{2,an}$ is closer to zero near the leading edge.)

In this section we have discussed the effects of $(c_1/c_2)_{feed}$, $(D_1/D_2)$, $Gz$, and $h/L$ for a redox system. $(c_1/c_2)_{feed}$ and $(D_1/D_2)$ determine which species is limiting, the Graetz number (or $Pe d_e/L$) determines the size of the thin-boundary-layer region, and $h/L$ determines the degree of interaction through the potential distribution.
Nomenclature for Chapter 3

\( c_{an} \) \hspace{1cm} \text{anode surface concentration, mol/cm}^3

\( c_1 \) \hspace{1cm} \text{concentration of species 1, mol/cm}^3

\( d_e \) \hspace{1cm} \text{equivalent diameter (2h for a channel), cm}

\( D_1 \) \hspace{1cm} \text{diffusion coefficient of species 1, cm}^2/s

\( Gz \) \hspace{1cm} \text{Graetz number (} \frac{\pi Pe d_e}{L} \text{ for a channel)}

\( h \) \hspace{1cm} \text{interelectrode gap thickness, cm}

\( I_{axial} \) \hspace{1cm} \text{current density flowing in the axial direction, A/cm}^2

\( i_{an} \) \hspace{1cm} \text{anode current density, A/cm}^2

\( i_{cath} \) \hspace{1cm} \text{cathode current density, A/cm}^2

\( L \) \hspace{1cm} \text{electrode length, cm}

\( Pe \) \hspace{1cm} \text{Péclet number (} 2<v>h/D \text{ for a channel)}

\( x \) \hspace{1cm} \text{axial coordinate, cm}

\( y \) \hspace{1cm} \text{normal coordinate, cm}
Greek

$\kappa_\infty$ conductivity of feed solution, ohm$^{-1}$ cm$^{-1}$

$\Phi$ potential in the solution as measured by a reference electrode of a given kind, V

$\Phi^*$ integration constant in equation 3-3

Subscripts

an anode

cath cathode

1 species 1

2 species 2

$\infty$ feed
4. Experimental Apparatus and Procedure

4.1. Introduction

To provide experimental confirmation of the model results, steady-state polarization curves (current vs. voltage) were measured in a channel flow cell containing the potassium ferri-/ferrocyanide redox couple in aqueous sodium hydroxide. The nickel electrodes, forming a portion of the walls of the flow channel, were separated by a polyethylene spacer. By varying the spacer thickness and the feed flow rate, delivered from a syringe pump, various Graetz-number regimes were studied. Since the polarization curves included a wide range of applied potentials, kinetically-controlled and mass-transfer-controlled currents were measured, but because the electrodes were not sectioned, no experimental information about the distribution of current was obtained.

Separate experiments with a rotating-disk electrode were carried out to measure the diffusion coefficient of the ferricyanide ion. Although many investigators had measured the diffusion coefficient of ferricyanide in basic solutions, their measurements were for equimolar solutions of ferri- and ferrocyanide. Since equimolar solutions are undesirable for limiting current measurements in a channel (see section 4.2), diffusion coefficients were required for solutions with excess ferrocyanide. Both Acosta\textsuperscript{76} and Fischl\textsuperscript{77} used such solutions, but they calculated their diffusion coefficients from the correlation developed by Gordon \textit{et al.}\textsuperscript{78} Although this correlation should be accurate, we felt that we might achieve better agreement between theory and experiment with a measured diffusion coefficient.

In section 4.2, we discuss the rationale for the selection of the electrochemical reaction, electrode materials, and reference electrode. Section 4.3 describes the electrolyte preparation. The flow channel itself is described in detail in section 4.4.1, and section
4.4.2 details the operating procedure. The rotating disk experimental procedure is outlined in section 4.5.

4.2. Selection of the Electrochemical System

The potassium ferri-/ferrocyanide redox couple has been used extensively for mass transfer studies. Acosta used the ferri-/ferrocyanide system to examine the economic tradeoff between increased mass transfer and increased pumping power in thin-gap channel flow-cells. He measured mass-transfer coefficients and friction factors in both the laminar and turbulent regimes and studied the effects of surface roughness. Fischl investigated the effects of turbulence promoters on mass transfer and pressure drop in a channel flow cell.

The decision to use the Fe(CN)$_6^{3-}$/Fe(CN)$_6^{4-}$ redox couple for the present study was based on previous studies, which have shown that redox systems in general are suitable for forced-convection mass transfer studies because, compared to plating systems, they have less free convection (smaller density changes), they reach steady state faster, and the electrode surface stays smooth. The Fe(CN)$_6^{3-}$/Fe(CN)$_6^{4-}$ reaction in particular is fast and reversible, and it gives a long limiting-current plateau. Although we were interested in the entire polarization curve, the long plateau is advantageous for studying interacting boundary layers at the limiting current.

There are some disadvantages of the ferri-/ferrocyanide redox couple. One such disadvantage is that the solutions contain cyanide ions, which can react with hydrogen ions to form poisonous HCN gas. This reaction is suppressed in basic solutions. Another disadvantage is that the solutions are unstable due to the light-induced decomposition of ferrocyanide (see equations 4-1 and 4-2) and the reaction of ferrocyanide with any dissolved oxygen to form ferricyanide and OH$^-$. A third disadvantage is that dissolved oxygen reacts on the cathode at approximately the potential of the ferricyanide
reduction, thus obscuring measurements of the current from the main cathodic reaction. The problems of the oxygen side reaction and the solution decomposition can be avoided by sparging the solution with nitrogen to remove the dissolved oxygen and by not storing the solution for more than a few days. Finally, the problem of electrode poisoning by cyanide ion has been reported, but it can be eliminated by careful washing and pretreatment of the electrodes,\textsuperscript{76,89,90} which are usually nickel or platinum. We chose nickel because nickel electrodes are less susceptible to the poisoning than platinum and they are less expensive.

Having chosen the redox couple and electrode material, the next question was whether to use excess supporting electrolyte. In this study, supporting electrolyte is desirable to eliminate migration, which is neglected in the model. Excess supporting electrolyte is generally used in industrial cells to lower the cell resistance, so migration was neglected in the model and suppressed in the experiments.

Basic supporting electrolyte (NaOH) was chosen to hinder the HCN side reaction, which would occur in acid solution or by the decomposition of ferrocyanide in the presence of light:

\begin{align*}
\text{Fe(CN)}_6^{4-} + \text{H}_2\text{O} \overset{\text{light}}{\rightarrow} [\text{Fe(CN)}_5\text{H}_2\text{O}]^{3-} + \text{CN}^- \\
\text{CN}^- + \text{H}_2\text{O} \Leftrightarrow \text{HCN} + \text{OH}^- .
\end{align*}

By adding excess OH\textsuperscript{-} ions, the equilibrium of the second reaction is driven to the left. Other advantages of basic supporting electrolytes compared to neutral supporting electrolytes are the higher solution conductivities and the longer cathodic limiting-current plateaus.

The feed composition was chosen so that the cathode (working electrode) is at the limiting current and the anode is below limiting current. Since the anode and cathode surface areas are equal in the channel and the diffusion coefficients are roughly equal (\(D_{\text{Fe}}\text{tt} \approx 20\% \text{ less than } D_{\text{Fe}}\text{rro})\),\textsuperscript{76} the only way to ensure that the anode is below
limiting current is to use a solution with excess ferrocyanide. It is important that the
counterelectrode be below the limiting current if the mass-transfer limitations are to be
attributed to the working electrode. In limiting-current studies with a redox system,
either electrode may be the limiting electrode. With the Fe(CN)$_6^{3-}$/Fe(CN)$_6^{4-}$ redox
system, there are advantages to choosing the cathode as the working (limiting) elec-
trode: in basic solutions, the cathode has the longer limiting-current plateau and oxygen
evolution on the anode is minimized.

Clearly, the long limiting-current plateau is desirable for accurate measurements of
the limiting current. The reason the cathodic limiting-current plateau is longer is that
the cathode has the larger difference between the thermodynamic potentials of the main
and side reactions. The main reaction is

cathode: Fe(CN)$_6^{3-} + e^- \rightarrow Fe(CN)$_6^{4-}$ (Ferri $\rightarrow$ Ferro) \hspace{1cm} (4-3)
anode: Fe(CN)$_6^{4-} \rightarrow Fe(CN)$_6^{3-} + e^-$ (Ferro $\rightarrow$ Ferri). \hspace{1cm} (4-4)

The side reactions are

cathode: $2H_2O + 2e^- \rightarrow H_2 + 2OH^-$ \hspace{1cm} (4-5)
anode: $4OH^- \rightarrow O_2 + 2H_2O + 4e^-$. \hspace{1cm} (4-6)

Figure 4-1 is a Pourbaix diagram showing the effect of pH on the length of the anodic
and cathodic plateaus. The dotted horizontal line at 0.36 V represents the standard
electrode potential for the ferri-/ferrocyanide redox couple, the line labeled “oxygen
potential” is the standard potential of the anode side reaction as a function of pH, and
the hydrogen-potential line is for the cathode side reaction. The length of the cathodic
plateau is roughly $V_{Ferri/Ferro}^2 - V_{H_2}^o$. The diagram shows that, in basic solutions, the
cathodic plateau is longer.

The other advantage of using the cathode as the working electrode is that the
Figure 4-1. Electrode vs. pH diagram.
oxygen-evolution side reaction can be controlled by forcing the anode to be below limiting current. Oxygen evolution is undesirable because it may produce an oxide film on the anode and, if the boundary layers are thick, the oxygen produced on the anode can react on the cathode. Also, oxygen reacts homogeneously with ferrocyanide to produce ferricyanide. To guarantee limiting current on the cathode only, the electrolyte was chosen to have a 5:1 excess of ferrocyanide. Excess sodium-hydroxide supporting electrolyte was used to suppress migration and the HCN side reaction.

The final task was to choose a reference electrode, although it is often omitted in studies requiring only the limiting current. Sections 34 and 35 of Newman's book give some general comments on the selection of a reference electrode. For example, a good reference electrode is reproducible, and the reaction is reversible (it has a high exchange current density). Since the ferri-/ferrocyanide reaction itself meets these criteria, it makes a good reference-electrode reaction. The advantage of a ferri-/ferrocyanide reference electrode is that the liquid-junction potential is minimized because the electrode is placed in the feed solution; the disadvantage is that the open-circuit potential provides no information about the composition of the feed solution.

A platinum-screen reference electrode was chosen because platinum is inert and less susceptible to poisoning than nickel and because a screen electrode has a high surface area, which is desirable to minimize the current density flowing through the reference electrode (the smaller the current density, the smaller the overpotential at the reference electrode, which represents a deviation from its equilibrium potential).

4.3. Electrolyte Properties and Preparation

The electrolyte compositions are tabulated below. The compositions were chosen to allow comparisons with other investigations. Solutions 1 and 2 were used in the channel experiments, and solutions 1 through 4 were used in the disk experiments.
<table>
<thead>
<tr>
<th>Solution</th>
<th>NaOH</th>
<th>KOH</th>
<th>K₃Fe(CN)₆</th>
<th>K₄Fe(CN)₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Fischl⁷⁷</td>
<td>0.3 M</td>
<td>0.01 M</td>
<td>0.05 M</td>
<td></td>
</tr>
<tr>
<td>2. Eisenberg⁷⁹</td>
<td>2.0 M</td>
<td>0.05 M</td>
<td>0.05 M</td>
<td></td>
</tr>
<tr>
<td>3. Selman⁸⁷</td>
<td>0.4 M</td>
<td>0.0143 M</td>
<td>0.0143 M</td>
<td></td>
</tr>
<tr>
<td>4. Mohr⁸⁶</td>
<td>0.85 M</td>
<td>0.005 M</td>
<td>0.005 M</td>
<td></td>
</tr>
</tbody>
</table>

All solutions were prepared from deoxygenated, deionized water. The water was deoxygenated in 500 ml batches by sparging with nitrogen for half an hour before the solution preparation and just before the experiment. We found that the solutions must be prepared with sparged water. When the solutions were sparged once before the experiment but not during the solution preparation, the results were scattered and irreproducible.† On the other hand, the carefully-prepared solutions gave reproducible results that fit the theoretical predictions (see chapter 5). A SYBRON/Barnstead ROTpure purification system deionized the water to a resistivity of 16-18 MΩ-cm. The physical properties of the solutions are tabulated in appendix H.

4.4. Channel-Flow-Cell Apparatus and Procedure

4.4.1. Experimental Apparatus

Figures 4-2 and 4-3 show the experimental set-up, and figures 4-4 and 4-5 show the channel flow cell. The major equipment — the channel flow cell, syringe pump, and potentiostat — are described below.

The channel flow cell, shown in figures 4-3 through 4-5, consists of parallel plates of lucite, each containing a section with a 6 mm thick nickel 200 electrode (99.0% pure) embedded in Shell Epon 828 oven-cure epoxy resin. The epoxy resin was chosen based on previous work⁷⁶ to provide a smooth junction between the electrode and the insulator.

† These results were discarded.
Figure 4-2. Schematic of experimental set-up.
Figure 4-3. Channel-flow-cell assembly (shown with the 0.323 cm spacer) and syringe pump.
Figure 4-4. Assembly drawing of the channel flow cell showing nickel cover plates, electrode plates, and polyethylene spacer.
Figure 4-5. Nickel cover plates, electrodes, and thin (0.043 cm) spacer.
on polishing. To ensure laminar flow, an entry length of 3 cm on the epoxy surfaces and
an exit length of 1 cm are provided before and after the metal electrodes. With the thin
spacer, these lengths are 30 $d_e$ and 10 $d_e$, respectively. Electrolyte enters and exits the
cell through circular holes in the top plate. The holes are located over recesses in the
lower plate, which are provided to act as distribution and collection plenums. The edges
of the recesses are rounded (see figures 4-4 and 4-5). Electrical connections are made by
banana plugs, screwed into blind-tapped holes on the back of each nickel electrode.
There was no concern about using dissimilar metals because the cell was isothermal, and
hundreds of millivolts were applied to the cell (except near open circuit, where the error
in the potential measurement was about 1 millivolt).

A polyethylene spacer of either 0.043 cm or 0.323 cm thickness separates the plates.
The spacer is sealed against the plates by Viton O-rings seated in grooves in the plates.
Fourteen screws around the perimeter of the cell assembly act on 0.318 cm thick nickel
cover plates, which are used to distribute the forces on the lucite plates, thus providing
uniform pressure on the O-rings.

Fluid is pumped into the cell with a Sage model 355 syringe pump, which can
deliver a maximum of 100 ml at any flow rate between 0.39 $\mu$l/hr and 126 ml/min.
Flow rates are measured by the bucket-and-stopwatch method. The syringe is a
Bekton-Dickenson 100 ml glass syringe with a corrugated uranyl glass fitting added to
the tip. Three-eighth inch O.D. BEV-A-LINE V tubing is heat-shrunk over the corru-
gated tip and fastened to a teflon Swagelok fitting screwed into the entrance hole in the
top plate of the cell assembly.

The platinum-screen reference electrode is placed upstream of the channel in a
small reservoir containing the feed solution (see figures 4-3 and 4-6). A ground-glass
stopcock prevents solution from flowing into the reference-electrode compartment but
allows electrical connection to the working electrode by a thin film of electrolyte
Figure 4-6. Reference-electrode side arm.
surrounding the stopcock (see figure 4-6).

Potential was applied to the cell with an AIS (Division of Floyd Bell) "Aardvark" model V-2LR-D potentiostat. A Pine Instruments RDE 3 potentiostat, tried in preliminary experiments under the same conditions, proved to be unstable.

4.4.2. Experimental Procedure

Before every four or five runs, the electrodes were polished down to one-micron diamond paste. Since the plates were too large for effective polishing with an 8- or 10-inch polishing wheel, the electrodes were hand-polished with a flat disk covered with a velvet polishing cloth. The plates were then washed with Liquinox soap and rinsed with deionized water and methanol, and the cell was assembled as shown in figure 4-4. The electrodes were not cleaned further by hydrogen evolution. A better experimental design would have provided a dummy counterelectrode to use for the hydrogen-evolution cleaning procedure, so that both channel electrodes would remain cathodically activated.

After the electrodes were polished, the cell was assembled, and the syringe was filled. Because the syringe was too large to draw the solution into the tip, it was filled from the top. Care was required to prevent the ground-glass plunger from jamming in the syringe barrel, so the plunger was greased before each run with Dow-Corning high vacuum (silicone) grease. The syringe was then connected to the cell, which was filled in the vertical position to eliminate air bubbles. The cell assembly was then leveled with anode over cathode to suppress free convection, and the syringe pump was started. After the flow became steady, the stopwatch for measuring flowrates was started.

The choice of flow rate depended on the desired Graetz number. For measuring diffusion coefficients, for example, high Graetz numbers (high flow rates) were used, so that the Lévéque approximation applies:
Polarization curves were obtained by measuring cathodic currents at various applied potentials and recording the current vs. time to check visually for steady state before moving to a new potential. To verify that the electrodes were not poisoned during the experiment, a data point on the limiting-current plateau was taken at the beginning and end of each run. If the limiting currents measured at the beginning and end of the run were significantly different, the data were discarded.

4.5. Rotating Disk Experiments

To measure the diffusion coefficient of Fe(CN)$_6^{3-}$ with the rotating disk system, the limiting current is measured at various rotation speeds, and, since the Levich equation applies,

$$I_{lim} = 0.620 nFAD^{2/3} \Omega^{1/2} \nu^{1/6} c_b,$$

the diffusion coefficient is obtained from the slope of a plot of $I_{lim}$ vs. $\Omega^{1/2}$.

4.5.1. Experimental Procedure

The disk experiments were carried out with a 5-mm platinum disk (Pine Instruments) and a Pine model PIR rotator. Experiments with a smaller nickel disk (1 mm) were unsuccessful, probably because of spherical diffusion effects. The potential of the working electrode was controlled by a Princeton Applied Research model 173 potentiostat with the PAR 175 Universal Programmer generating the potential sweeps. The platinum-screen reference and counterelectrodes were placed in the side-arms of the vessel (see figure 4-7).

The solutions were prepared, as in section 4.4, and sparged for one hour in the 300-ml vessel. Table 4-1 shows the four solution compositions used, corresponding to those of Fischl, Eisenberg, Selman, and Mohr.
Figure 4-7. Sketch of rotating disk electrode apparatus.
Table 4-1. Solution Compositions

<table>
<thead>
<tr>
<th>Solution</th>
<th>NaOH</th>
<th>KOH</th>
<th>K₃Fe(CN)₆</th>
<th>K₄Fe(CN)₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1 Fischl</td>
<td>0.3 M</td>
<td>0.01 M</td>
<td>0.05 M</td>
<td></td>
</tr>
<tr>
<td>#2 Eisenberg</td>
<td>2.0 M</td>
<td>0.05 M</td>
<td>0.05 M</td>
<td></td>
</tr>
<tr>
<td>#3 Selman</td>
<td>0.4 M</td>
<td>0.0143 M</td>
<td>0.0143 M</td>
<td></td>
</tr>
<tr>
<td>#4 Mohr</td>
<td>0.85 M</td>
<td>0.005 M</td>
<td>0.005 M</td>
<td></td>
</tr>
</tbody>
</table>

The electrodes were prepared by polishing down to one-micron diamond paste, washing with Liquinox soap, and rinsing with deionized water and acetone. Before each potential sweep, the working electrode was cleaned further by hydrogen evolution for two minutes. Recall that this extra cleaning procedure was not possible with the channel flow cell as designed. Potential sweeps of 5 mV/sec gave flat limiting-current plateaus for the rotation speeds of 400, 900, 1600, and 2500 rpm. Reproducibility was checked by repeating each experiment with a new 300-ml batch of solution and reversing the order of the rotation speeds (i.e. 2500, 1600, 900, and 400 rpm).

The results of both the disk and the channel experiments were reproducible and agreed well with the theory. One suggestion for improvement of the procedure is to obtain direct measurements of the physical properties of the solution, such as concentration and viscosity. Temperature control would also be helpful, although it may be somewhat difficult to achieve in the channel. A discussion of the experimental results can be found in section 5.3.
Nomenclature for Chapter 4

\[ A \quad \text{surface area of electrode, cm}^2 \]
\[ c_b \quad \text{bulk concentration, mol/cm}^3 \]
\[ D \quad \text{diffusion coefficient, cm}^2/s \]
\[ d_e \quad \text{equivalent diameter (}2h\text{ for a channel), cm} \]
\[ F \quad \text{Faraday's constant, 96,487 C/eq} \]
\[ Gz \quad \text{Graetz number } \left( \frac{\pi Pe d_e}{L} \text{ for the channel} \right) \]
\[ h \quad \text{interelectrode gap thickness, cm} \]
\[ I_{\text{lim}} \quad \text{limiting current, A} \]
\[ L \quad \text{electrode length, cm} \]
\[ n \quad \text{number of electrons transferred, eq/mol} \]
\[ Nu_{\text{avg}} \quad \text{average Nusselt number} \]
\[ Re \quad \text{Reynolds number } (2<\nu>h/\nu \text{ for a channel}) \]
\[ Sc \quad \text{Schmidt number } (\nu/D) \]
\[ V_{\text{Ferri}/\text{Ferro}}^\circ \quad \text{standard electrode potential for the ferri- / ferrocyanide redox reaction, V} \]
\( V_{H_2}^o \) standard electrode potential for hydrogen evolution, V

Greek

\( \nu \) kinematic viscosity, cm\(^2\)/s

\( \Omega \) rotation speed, radians/s
5. Comparison of Experimental and Theoretical Results

5.1. Introduction

This chapter presents a comparison between experimental and theoretical results at various Graetz numbers. At high Graetz numbers, the boundary layers are thin, and simple equations allow diffusion coefficients to be obtained from limiting current measurements either on a rotating-disk electrode or in the channel. For low Graetz numbers, we investigate interacting boundary layers in the channel.

Preliminary comparisons between the theoretical and experimental results had shown some discrepancies, even for thin-boundary-layer conditions, therefore we decided to measure the diffusion coefficient of ferricyanide for the solutions we were using, instead of calculating it from Gordon's correlation.\textsuperscript{78} Diffusion coefficients are often measured on a rotating-disk electrode by the limiting current technique. The basis for this technique is that, since the boundary layer is thin, the limiting current may be calculated from the simple Levich equation (see chapter 4, equation 8). The same idea may be applied to the channel geometry because, if the boundary layers are thin, the limiting current may be calculated from the Lévéque equation. The important difference between the two geometries is that the mass-transfer limited current distribution is highly nonuniform in the channel, but uniform on the disk. Another difference between the two experiments is that the disk electrode is platinum and the channel electrodes are nickel. Thus, the two experiments provide independent measurements of the diffusion coefficient.

Sections 5.2 and 5.3 present comparisons between diffusion coefficients measured on the disk, in the channel, and in the literature. Section 5.4 presents measured and simulated polarization curves for the channel at low Graetz numbers (interacting boundary

\textsuperscript{*} Recall that the Graetz number, defined in chapter 1, is a measure of the boundary-layer interaction.
5.2. Results for Thin Boundary Layers

This section presents results for thin boundary layers (high Graetz numbers) both on the rotating-disk electrode and in the channel. The purpose of these experiments was to measure the composition-dependent diffusion coefficient of the ferricyanide ion for the solution that was used in most of the channel experiments. (No previous measurements for this solution were found.) The disk measurements were carried out with four solution compositions — the first was the composition used in the thin gap channel experiments, the same composition used by Fischl,77 and the other three were solutions for which diffusion coefficients have been measured by other investigators. In this way, we could compare our disk and channel measurements to each other for the first solution composition and compare our measurements to the literature for the last three solutions.

The results for each solution composition are presented together to show the comparison between the various measurements. Tables 5-1 through 5-3 show the four solution compositions. Note that only two solutions were analyzed and that the ferricyanide calculated by weighing agrees with the analysis. This conclusion was also verified in separate experiments that showed that the concentration of ferricyanide calculated by weighing and by iodometric titration with thiosulfate differed by about 1.5%. For this reason, the remaining solutions were not analyzed.

Limiting-current measurements for each of the four electrolytes are summarized in tables 5-4 through 5-6 and in appendix H. We shall now discuss the results for each solution.

Figures 5-1 and 5-2 show the Levich plots (see equation 4-8) from the rotating-disk experiments with the solution composition used by Fischl (solution 1). The diffusion
Figure 5-1. Levich plot for solution 1a.
Figure 5-2. Levich plot for solution 1b.
Table 5-1. Nominal solution compositions (mol/liter).

<table>
<thead>
<tr>
<th>Solution</th>
<th>NaOH</th>
<th>KOH</th>
<th>K₃Fe(CN)₆</th>
<th>K₄Fe(CN)₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Fischl⁷⁷</td>
<td>0.3 M</td>
<td>0.01 M</td>
<td>0.05 M</td>
<td></td>
</tr>
<tr>
<td>2. Eisenberg⁷⁹</td>
<td>2.0 M</td>
<td>0.05 M</td>
<td>0.05 M</td>
<td></td>
</tr>
<tr>
<td>3. Selman⁸⁷</td>
<td>0.4 M</td>
<td>0.0143 M</td>
<td>0.0143 M</td>
<td></td>
</tr>
<tr>
<td>4. Mohr⁸⁸</td>
<td>0.85 M</td>
<td>0.005 M</td>
<td>0.005 M</td>
<td></td>
</tr>
</tbody>
</table>

Table 5-2. Solution concentrations for disk experiments (mol/liter).

<table>
<thead>
<tr>
<th>Solution</th>
<th>NaOH</th>
<th>KOH</th>
<th>K₃Fe(CN)₆</th>
<th>K₄Fe(CN)₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a†</td>
<td>0.2457</td>
<td>0.01000</td>
<td>0.05030</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.2582</td>
<td>0.01003</td>
<td>0.04983</td>
<td></td>
</tr>
<tr>
<td>1b</td>
<td>0.2992</td>
<td>0.01001</td>
<td>0.05003</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2.1091</td>
<td>0.04993</td>
<td>0.05008</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.4015</td>
<td>0.01430</td>
<td>0.01479</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.7230</td>
<td>0.00472</td>
<td>0.05020</td>
<td></td>
</tr>
</tbody>
</table>

Table 5-3. Solution concentrations for channel experiments (mol/liter).

<table>
<thead>
<tr>
<th>Solution</th>
<th>NaOH</th>
<th>K₃Fe(CN)₆</th>
<th>K₄Fe(CN)₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>1c</td>
<td>0.3190</td>
<td>0.01001</td>
<td>0.05009</td>
</tr>
<tr>
<td>1d</td>
<td>0.3415</td>
<td>0.01001</td>
<td>0.05109</td>
</tr>
<tr>
<td>1e†</td>
<td>0.2832</td>
<td>0.01012</td>
<td>0.05758</td>
</tr>
<tr>
<td>2</td>
<td>0.2996</td>
<td>0.01002</td>
<td>0.050093</td>
</tr>
<tr>
<td>1f</td>
<td>0.3011</td>
<td>0.01004</td>
<td>0.05042</td>
</tr>
<tr>
<td>2</td>
<td>2.1091</td>
<td>0.04993</td>
<td>0.05008</td>
</tr>
</tbody>
</table>

† Boldface refers to solutions that were analyzed for ferricyanide by iodometric titration with zinc and for ferrocyanide by potentiometric titration with ceric sulfate. All other concentrations were calculated by weighing.
Table 5-4. Diffusion coefficients of Fe(CN)₆³⁻ measured on the rotating-disk electrode: comparison to literature.

<table>
<thead>
<tr>
<th>Solution</th>
<th>$D$ (cm²/s)</th>
<th>$T$ (°C)</th>
<th>$\left(\frac{\mu D}{T}\right)_\text{est.}$ (dyne/K)</th>
<th>$\left(\frac{\mu D}{T}\right)_\text{lit.}$ (dyne/K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6.19×10⁻⁶ ±3%</td>
<td>19.7</td>
<td>2.38×10⁻¹⁰</td>
<td>(2.35×10⁻¹⁰)⁺</td>
</tr>
<tr>
<td>2</td>
<td>4.30×10⁻⁶ ±2%</td>
<td>—</td>
<td>2.30×10⁻¹⁰</td>
<td>2.50×10⁻¹⁰</td>
</tr>
<tr>
<td>3</td>
<td>5.98×10⁻⁶ ±2%</td>
<td>21.3</td>
<td>2.25×10⁻¹⁰</td>
<td>2.60×10⁻¹⁰</td>
</tr>
<tr>
<td>4</td>
<td>6.34×10⁻⁶ ±2%</td>
<td>20.0</td>
<td>2.38×10⁻¹⁰</td>
<td>1.83×10⁻¹⁰</td>
</tr>
</tbody>
</table>

Table 5-5. Comparison between diffusion coefficients measured on the disk and in the channel.

<table>
<thead>
<tr>
<th>Solution</th>
<th>$D_{\text{disk}}$ (cm²/s)</th>
<th>$D_{\text{channel}}$ (cm²/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6.19×10⁻⁶</td>
<td>5.93×10⁻⁶</td>
</tr>
<tr>
<td>2</td>
<td>4.30×10⁻⁶</td>
<td>4.32×10⁻⁶</td>
</tr>
<tr>
<td>3</td>
<td>5.98×10⁻⁶</td>
<td>—</td>
</tr>
<tr>
<td>4</td>
<td>6.34×10⁻⁶</td>
<td>—</td>
</tr>
</tbody>
</table>

This "disk" diffusion coefficient may be compared to the diffusion coefficient measured in the channel. Figure 5-3 is a log-log plot of dimensionless limiting current ($Nu/(ReScd_c/L)$) vs. dimensionless flow rate ($Re_d/L$). The solid line is a fit of the data to the rearranged Lévêque equation,

$$\frac{Nu}{ReScd_c/L} = 1.8488(ReSc\frac{d_c}{L})^{-2/3}. \quad (5-1)$$

The motivation for this equation is to plot a dimensionless current that does not contain the unknown diffusion coefficient. The channel diffusion coefficient, 5.93×10⁻⁶ cm²/s, is obtained from the intercept of this line.

The results from solution 2 (see table 5-2) are shown in figures 5-4 and 5-5. Tables 5-4 and 5-5 summarize the data from the disk and channel; table 5-6 shows the

* Viscosities were estimated from the correlation of Boeffard et al.⁸⁵
Figure 5-3. Fit of channel data to Lévêque equation for solutions 1c, 1d, and 1e.

\[ D_{\text{channel}} = 5.93 \times 10^{-6} \text{ cm}^2/\text{s} \]

- thin spacer, \( c = 0.01 \text{ M} \)
- thick spacer, \( c = 0.01 \text{ M} \)
Figure 5-4. Levich plot for solution 2.
Figure 5-5. Fit of channel data to Léveque equation for solution 2.
comparison to literature. There is some discrepancy between our data and Eisenberg's, but it is not surprising because the experimental techniques are different. Eisenberg used the capillary method, which is to immerse a capillary filled with the solution into a large reservoir of another solution that has no ferricyanide. The composition of the solution is then determined (by titration) after approximately one time constant later, and the diffusion coefficient is obtained from the solution of the one-dimensional transient diffusion equation. Although the titration measurements are generally accurate to 2 or 3%, the capillary method could yield diffusion coefficients that are too high if either spherical diffusion or convection is important. Also, the titration measurements may be less accurate for small volumes of solution. According to Gordon's correlation, which is based on accurate rotating disk measurements in equimolar solutions, the Stokes-Einstein coefficient should be lower than that predicted by Eisenberg; our measurements are only about 3% lower than Gordon's prediction (see table 5-6). Gordon's data suggest that \((D\mu/T)\times10^{10}\) is never less than 2.34.

The results for solution 3 are shown in figure 5-6. These measurements are about 13% lower than those of Selman.\(^87\) Figure 5-7 and table 5-4 show the results for Mohr's\(^86\) KOH solution (solution 4). Here, our diffusion coefficient is higher than the direct measurement,\(^86\) but close to the value predicted by the correlation of Gordon \textit{et al}.\(^78\)

The channel data for the different flow rates, concentrations, and gap thicknesses are summarized in figure 5-8. Note that all the data fit the Lévéque equation

\[
Nu = 1.8488(ReSc\frac{d_c}{L})^{1/3}
\]

because the Graetz number is high enough for the Lévéque approximation to apply. Table 5-6 summarizes all of the data from our experiments and the literature.
Figure 5-6. Levich plot for solution 3.
Figure 5-7. Levich plot for solution 4.
Figure 5-8. Léveque plot of channel data using the diffusion coefficients obtained from figures 5-3 and 5-5.
5.3. Discussion

The accuracy of the experimental results clearly depends on the accuracy of the measurements of concentration, viscosity, and current. The discarded results of preliminary experiments with the rotating-disk electrode had shown that the solutions must be prepared with deoxygenated water and that it is not sufficient to remove the oxygen just prior to the experiment. The necessity of removing all oxygen stems from the reaction of oxygen with ferrocyanide to produce ferricyanide, which changes the composition of the solution (and renders it unknown). Although we were careful to keep out oxygen and light, some decomposition of the solutions may have occurred. A better procedure would be to work with oxygen-free solutions, use a reference electrode that is sensitive to ferri- and ferrocyanide, and analyze the solutions immediately before and after the experiments. The analysis of the effluent after the experiment would indicate whether oxygen had evolved on the anode, assuming no hydrogen is evolved on the cathode. When no oxygen is evolved, the concentration of the effluent is the same as that of the feed because the currents are equal on the two electrodes. On the other hand, if oxygen is evolved on the anode and is swept out before reacting homogeneously, more ferrocyanide is produced than ferricyanide; therefore the analysis of the effluent would reveal the total current for the oxygen reaction. The evolution of oxygen would only affect the results of the channel experiments with interacting boundary layers because the oxygen could diffuse to the cathode and react back to hydroxide. In thin boundary layer experiments, the limiting current would be the same, regardless of whether oxygen is evolved on the counterelectrode.

The viscosity of the solution is another property that should be determined accurately by direct measurement. Since the viscosity is sensitive to temperature, the experiments should be carried out in a constant-temperature bath. (In our experiments the temperature drifted by as much as 0.2 °C during the course of a run.)
Finally, the current is sensitive to flow patterns and to the electrode surface. As we found in preliminary experiments, free convection can change the current noticeably; therefore we leveled the cell with the anode over the cathode. Naturally, it is also important to remove any bubbles from the system. The electrode surface plays an important role because it can become poisoned by the cyanide ion. As mentioned previously, our experimental cell design should have provided a dummy counterelectrode to use for the cathodic-hydrogen-evolution cleaning procedure.

In addition to the improvements suggested above, we should consider the possibility of measuring the current by, for example, a sectioned-electrode technique. Another interesting way to learn about the current distribution without using a sectioned electrode is to use reference electrodes both upstream and downstream, as done by Matlosz. The difference in potential between these two electrodes would indicate the amount of axial current, which can be significant even with a large aspect ratio \((L/h)\).

Although our experiments could be improved by the techniques discussed above — use of an upstream and downstream reference electrode, analysis of the solutions before and after the run, measurement of the viscosity, constant-temperature control, and cathodic electrode pretreatment — the essential procedures — removal of oxygen, suppression of free convection, and electrode polishing — were carried out. This procedure led to reproducible results and, in general, the independent disk and channel measurements agree with each other and fall in the range of measurements by other investigators (see table 5-6).

Our measurements may be lower than some owing to the slight uncertainty in the viscosity and concentration of our solutions. Another possible cause of the discrepancy is that the nickel channel electrodes may have been partially poisoned by the cyanide ion. This explanation is unlikely, however, because the platinum disk electrode was

*Calculated from correlation of Gordon et al.*
Table 5-6. Values of \((\mu D/T) \times 10^{10}\) (dyne/K)

<table>
<thead>
<tr>
<th>Supp. Elec.</th>
<th>Ferri 0.01 M</th>
<th>Ferri 0.05 M</th>
<th>Ferro 0.01 M</th>
<th>Ferro 0.05 M</th>
<th>Equimolar ferri/ferro</th>
</tr>
</thead>
<tbody>
<tr>
<td>NaOH 0.3 M</td>
<td>2.06</td>
<td>1.92</td>
<td>2.03</td>
<td>2.25</td>
<td></td>
</tr>
<tr>
<td>NaOH 2 M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NaOH 0.4 M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KOH 0.85 M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KOH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NaOH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KCl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KNO₃</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- disk
- channel
- "Gordon"
- Eisenberg
- Selman
- Mohr
- Arvia
- Sih
- Appel
- Yip

*Calculated from correlation.
†From Smyrl's unpublished data, which Appel referenced as a personal communication. The unpublished \(D_{Ferri}\) is lower than the published \(D_{Ferri}\), contrary to what is expected.²⁸

* Taken from Lin by Yip.²⁸

cleaned carefully, and the diffusion coefficient measured with it was the same (for solution #1) or lower (solution #2) than that measured in the channel. Also, the channel diffusion coefficients were no higher with freshly polished electrodes.
In summary, the measured diffusion coefficients are $D=6.06 \times 10^{-6}$ cm$^2$/s for an 0.01 M $K_3Fe(CN)_6/0.05$ M $K_4Fe(CN)_6/0.3$ M NaOH solution and $D=4.31 \times 10^{-6}$ cm$^2$/s for an 0.05 M $K_3Fe(CN)_6/0.05$ M $K_4Fe(CN)_6/2.0$ M NaOH solution.

5.4. Measured and Simulated Polarization Curves

Figures 5-9 through 5-18 show the comparison between the experimental and predicted results. The parameters used in the model are the measurements of the cell dimensions, the diffusion coefficient measured in the high-Graetz-number experiments, the thermodynamic parameters, and the kinetic parameters from Vetter. As the figures show, the agreement is quite good without any adjustment of parameters. In the low-Graetz-number range, both the theoretical and experimental limiting currents are significantly higher than the value predicted from the Lévêque equation. The slight disagreements between the experimental and theoretical limiting currents are probably due to an uncertainty in the solution concentration, or, possibly, inaccurate flow measurements. The low-current region also shows slight discrepancies; the data show a steeper initial rise in the current vs. potential and an earlier appearance of mass-transfer limitations. Polarization curves obtained by other investigators, such as Eisenberg and Mohr, show discrepancies of the same sign and magnitude. The parameters that typically affect this region are thermodynamic and geometrical parameters, and the transfer coefficients, which are all well-established and should not be adjusted. The exchange current density is less certain, but adjusting it will not improve the fit, as we shall see shortly.

The initial slope in the current-potential curve is often governed by kinetics and not mass transfer. In such cases, increasing the exchange current density will increase the initial slope because the surface overpotential is decreased. On the other hand, if the exchange current density is high, the surface overpotential is a negligible contribution to
Figure 5-9. Components of the total cell potential for runs 1c-1 through 1c-4 with the thick (0.323 cm) spacer, $P_e/L = 586.15$. 
Figure 5-10. Components of the total cell potential for runs 1c-5 through 1c-8 with the thick (0.323 cm) spacer, \( Pe_d/L = 809.5 \).
Figure 5-11. Components of the total cell potential for runs 1d-1 through 1d-6 with the thick (0.323 cm) spacer, $P_d/L = 388.14$. 

\[ \eta_s, \eta_s, \text{cath}, \Phi^0_a, \Phi^0_c \]

\[ V_a, \eta_c, \text{an} \]

\[ V_c, \eta_c, \text{cath} \]
Figure 5-12. Components of the total cell potential for run 1f-1 with the thin (0.043 cm) spacer, $P ed_e/L = 3.4784$. 
Figure 5-13. Components of the total cell potential for run 1f-2 with the thin (0.043 cm) spacer, $P_{ed}/L = 809.5$. 
Figure 5-14. Comparison between theory and experiment for run 1c, $P_{ed} / L = 586.15$. 
Figure 5-15. Comparison between theory and experiment for run 1c, $P_{edc}/L = 809.50$. 
Figure 5-16. Comparison between theory and experiment for run 1d.
Figure 5-17. Comparison between theory and experiment for run 1f-1.
Figure 5-18. Comparison between theory and experiment for run 1f-2.
the cell potential, even below the limiting current. Figures 5-9 through 5-13 demonstrate that this is the case for the $\text{K}_3\text{Fe(CN)}_6 / \text{K}_4\text{Fe(CN)}_6$ system under investigation. The figures show theoretical predictions of each of the components of the total cell potential — $\Phi_d^* - \Phi^*$, $\Phi_e^* - \Phi^*$, surface overpotential, and concentration overpotential — evaluated at the center of the electrodes. The potentials in these plots are relative to $\Phi^*$ (see chapter 1), which is expected to be a rough estimate of $V_{\text{ref}}$, the potential measured by the upstream reference electrode.\footnote{$\Phi^*$ is actually the average between the upstream and downstream potentials, as we shall demonstrate shortly.} The cell potential is the horizontal distance between the line on the far left, $V_{\text{an}}$, and the line on the far right, $V_{\text{cath}}$. The distance between the lines labeled $(\Phi_d^* - \Phi^*)$ and $(\Phi_e^* - \Phi^*)$ is a rough estimate of the ohmic drop. In each experiment, the ohmic drop is a negligible contribution to the cell potential. The important feature of the plots is that the ohmic drop and surface overpotential are negligible; therefore the cell potential is entirely due to concentration overpotential. (The lines for concentration overpotential are superimposed on the lines for total cell potential.) Eisenberg\textsuperscript{79} also showed that, with carefully cleaned electrodes, the surface overpotential is negligible. Thus, adjusting the exchange current density cannot increase the initial slope of the theoretical current-potential curve, leading to the question of whether the discrepancy is an experimental artifact. Since some other investigations show the same discrepancy, this question is difficult to answer. One possible source of experimental error, in our experiments and others, is that the current measurements may not be at steady state. In our experiments, the potential was stepped to each new value, causing a spike in current, and the measurement was taken after a steady current was attained. Eisenberg used a similar method.

It might seem that another possible source of the discrepancy is that $V_{\text{ref}}$ is not equal to $\Phi^*$; that is, the experimental and theoretical curves are not plotted with respect
to the same potential. The difference between these potentials is small, however. Figure 5-19 shows a plot of the potential distribution for a thin-spacer experiment with non-interacting boundary layers. The upper curve shows the anode potential distribution, and the lower curve shows the cathode potential distribution. The short horizontal lines on the left and right are the upstream and downstream potentials, respectively. Converting into dimensional quantities shows that \( V_{ref} \) and \( \Phi^* \) differ by only 0.45 mV.

Figure 5-19 has several interesting features. The center dashed line, for example, is a rough estimate of the potential along the centerline of the channel; it is calculated from the first term of equation 1-6:

\[
\Phi(x) = \Phi^* - \frac{1}{2\pi k}\int_0^L [i_{cat}(x') + i_{an}(x')] \ln \cosh \left( \frac{\pi(x-x')}{2h} \right) \, dx'.
\]

This quantity evaluated in the limit as \( x \to -\infty \) is the first moment of \( (i_a + i_c) \), and it represents the upstream potential. Note that the upstream and downstream potentials are related by

\[
\Phi^*(-\infty) - \Phi^* = -(\Phi^*(+\infty) - \Phi^*);
\]

therefore \( \Phi^* \) is the average of the upstream and downstream potentials. Equation 5-2 shows that if \( i_a = i_c \) everywhere, the upstream potential, \( V_{ref} \), equals \( \Phi^* \). The difference between \( V_{ref} \) and \( \Phi^* \) therefore gives an indication of how well \( i_a \) mirrors \( i_c \).

It is interesting that, for the thin-spacer experiment, one would expect to be able to ignore axial currents because \( h/L \) is very small (0.0075353). Figures 5-19 and 5-20 show that this is not the case. The center-line potential distribution (see figure 5-19) shows a large region of significant axial variation. The potential variation in this edge region is caused by what can be regarded as a dipole of \( (i_a + i_c) \) at the leading edge (see figure 5-20); at \( x = 0 \), \( (i_a + i_c) \ll 0 \), and immediately downstream of \( x = 0 \), \( (i_a + i_c) > 0 \). Note that this potential variation corresponds to having most of the anodic overpotential at the leading edge. Putting more anodic current at the front
Figure 5-19. Computed potential distribution for run 1e.
Figure 5-20. \((i_a + i_c)\) for run 1e.
tends to cancel the dipole; therefore the current and potential distributions represent a balance between competing effects. Also note that the region of axial potential variation is quite large. The potential distribution adjacent to the electrodes has superimposed on the center-line variation the effect of the local current density (see the second term of equation 1-6), which results in the spikes at the leading edge. Although the cathode is insensitive to the local potential variations because it is at limiting current, the anode current distribution is strongly dependent on the potential distribution, whether or not the surface overpotential is negligible. If the concentration overpotential dominates, the potential distribution on the anode will affect the current and concentration distribution. If the surface overpotential dominates, the effect of the ≈1-mV potential variations may be quite significant because the current depends exponentially on the potential. For the reasons discussed above, it is important to consider axial current and potential variations, even for small h/L. It would have been interesting to conduct the experiments with upstream and downstream reference electrodes to measure the amount of axial variation in the center-line potential.

In summary, the model has given detailed information about the potential distribution that would be difficult to obtain experimentally. The close agreement between the theoretical and experimental polarization curves, without any adjustment of parameters, suggests that the model is applicable over a wide range of Graetz numbers.
### Nomenclature for Chapter 5

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>diffusion coefficient, cm$^2$/s</td>
</tr>
<tr>
<td>$D_{channel}$</td>
<td>diffusion coefficient measured with the channel, cm$^2$/s</td>
</tr>
<tr>
<td>$D_{disk}$</td>
<td>diffusion coefficient measured with the disk, cm$^2$/s</td>
</tr>
<tr>
<td>$d_e$</td>
<td>equivalent diameter ($2h$ for a channel), cm</td>
</tr>
<tr>
<td>$D_{Ferri}$</td>
<td>diffusion coefficient of the ferricyanide ion, cm$^2$/s</td>
</tr>
<tr>
<td>$D_{Ferro}$</td>
<td>diffusion coefficient of the ferrocyanide ion, cm$^2$/s</td>
</tr>
<tr>
<td>$L$</td>
<td>electrode length, cm</td>
</tr>
<tr>
<td>$Nu$</td>
<td>Nusselt number</td>
</tr>
<tr>
<td>$Re$</td>
<td>Reynolds number ($2&lt;v&gt;h/\nu$ for a channel)</td>
</tr>
<tr>
<td>$Sc$</td>
<td>Schmidt number ($\nu/D$)</td>
</tr>
<tr>
<td>$T$</td>
<td>temperature, K (unless noted as °C)</td>
</tr>
<tr>
<td>$V_{an}$</td>
<td>potential of the anode, V</td>
</tr>
</tbody>
</table>
Greek

\( V_{\text{cath}} \) potential of the cathode, V
\( V_{\text{ref}} \) potential of the reference electrode, V

\( \mu \) viscosity, g/cm-s

\( \Phi_a^o \) potential in the solution adjacent to the anode (just outside the diffuse double layer), V

\( \Phi_c^o \) potential in the solution adjacent to the cathode (just outside the diffuse double layer), V

\( \Phi^* \) integration constant in equation 1-5, V
Appendix A. The Léveque Series

A.1. Introduction

To describe mass transfer to the wall of a channel with a step change in concentration, the Léveque solution\(^1\) can be used as the first term in a series, which is valid in the mass-transfer entrance region. This approach\(^2,3\) has been applied to the problem of mass transfer to the wall of a tube (the Graetz problem), and the procedure is illustrated for an annulus in problem 1 in chapter 17 of reference 28. Here, the channel problem is treated.

A.2. Mathematical Formulation

In terms of dimensionless variables, the convective diffusion equation for a channel flow -cell is

\[
(1-\xi^2) \frac{\partial \theta}{\partial \xi} = \frac{\partial^2 \theta}{\partial \xi^2},
\]

where, \(\theta\) is the dimensionless concentration, defined as \(\frac{c_i-c_o}{c_b-c_o}\), \(\xi\) is the dimensionless axial variable, and \(\eta\) is the dimensionless transverse variable, with the center of the channel at \(\eta=0\). If the step change is at \(\xi=+1\), the boundary conditions are

\[
\theta = 1 \text{ at } \eta = 0 \\
\theta = 0 \text{ at } \eta = 1 \\
\theta \to 1 \text{ as } \eta \to -\infty \text{ (outside the diffusion layer)}. \]

Equation A-1 can be rewritten in terms of the independent variables \(\eta\) and \(\xi\), where \(\eta\) is the similarity-transform variable:

\[
\eta = (1-\xi)(\frac{2}{9\xi})^{1/3}.
\]

In terms of \(\xi\) and \(\eta\), equation A-1 is
with boundary conditions \( \theta = 0 \) at \( \eta = 0 \) and \( \theta \to 1 \) as \( \eta \to \infty \).

An additional variable substitution, \( Z = \left( \frac{9 \xi}{2} \right)^{1/3} \), transforms equation A-6 to
\[
\frac{\partial^2 \theta}{\partial \eta^2} + 3\eta^2 \frac{\partial \theta}{\partial \eta} - 9\xi \eta \frac{\partial \theta}{\partial \xi} = \frac{3}{2} \eta^3 \left( \frac{9 \xi}{2} \right)^{1/3} \frac{\partial \theta}{\partial \eta},
\]
(A-6)

If \( \theta \), expressed as a power series in \( Z \), is substituted into equation A-7, one can equate the coefficients of the powers of \( Z \) to obtain a set of equations. If \( \theta \) is expressed as
\[
\theta = \sum \theta_i Z^i,
\]
then the set of equations is:
\[
\theta_0^* + 3\eta^2 \theta_0' = 0
\]
(A-9)
\[
\theta_1^* + 3\eta^2 \theta_1' - 3\eta \theta_1 = \frac{3}{2} \eta^3 \theta_0'
\]
(A-10)
\[
\theta_2^* + 3\eta^2 \theta_2' - 6\eta \theta_2 = -\frac{3}{2} \eta^2 \theta_1 + \frac{3}{2} \eta^3 \theta_1'.
\]
(A-11)
where the left side of each equation is of the form
\[
L_m(\theta_m) = \theta_m^* + 3\eta^2 \theta_m' - 3m \eta \theta_m.
\]
(A-12)
Note that equation A-9 is the Lévéque equation; therefore,
\[
\theta_0 = \frac{1}{\Gamma\left( \frac{4}{3} \right)} \int_0^{\eta} e^{-x^3} dx,
\]
(A-13)
where \( \Gamma\left( \frac{4}{3} \right) = 0.89298 \). Thus, A-10 becomes
\[
\theta_1^* + 3\eta^2 \theta_1' - 3\eta \theta_1 = \frac{3}{2} \eta^3 \frac{1}{\Gamma\left( \frac{4}{3} \right)} e^{-\eta^3}.
\]
(A-14)
This equation can be solved by reduction of order, because a homogeneous solution, \( \theta_1 = \eta \), is known. After integration by parts and rearrangement, the solution can be obtained:
\[
\theta_1 = \frac{-\eta^2 e^{-\eta^2}}{10 \Gamma\left(\frac{4}{3}\right)} - \frac{1}{10 \Gamma\left(\frac{4}{3}\right)} \eta \int_\eta^\infty e^{-x^2} dx .
\]  
(A-15)

Differentiating this equation and substituting into equation A-11 gives

\[
\theta_2'' + 3\eta^2 \theta_2' - 6\eta \theta_2 = \frac{9}{20} \eta^7 \frac{e^{-\eta^3}}{\Gamma\left(\frac{4}{3}\right)} .
\]  
(A-16)

The particular solutions to this equation are of the form \( \theta_2 = \eta^p e^{-\eta^3} \), since

\[
L_2(\eta^p e^{-\eta^3}) = p(p-1)\eta^{p-2}e^{-\eta^3} - 3(p+4)\eta^{p+1}e^{-\eta^3} .
\]  
(A-17)

Because \( \eta \) is raised to the 7th power in equation A-16, the particular solution is

\[
\theta_{2,p} = a\eta^p e^{-\eta^3} + b\eta^2 e^{-\eta^3} + ce^{-\eta^3} ,
\]  
(A-18)

where

\[
a = \frac{-3}{200 \Gamma\left(\frac{4}{3}\right)} , \quad b = \frac{-3}{140 \Gamma\left(\frac{4}{3}\right)} , \quad c = \frac{-3}{280 \Gamma\left(\frac{4}{3}\right)} .
\]  
(A-19)

A homogeneous solution is

\[
\theta_{2,h} = K \int_0^1 \frac{x^{1/3}}{(1-x)^{2/3}} \exp\left(\frac{-\eta^3}{1-x}\right) dx ,
\]  
(A-20)

where the constant \( K \) is obtained from the boundary condition \( \theta_2(0) = 0 \). Combining the particular and homogeneous solutions yields the complete solution:

\[
\theta_2 = \frac{-3}{200 \Gamma\left(\frac{4}{3}\right)} \eta^3 e^{-\eta^3} - \frac{3}{140 \Gamma\left(\frac{4}{3}\right)} \eta^3 e^{-\eta^3} - \frac{3}{280 \Gamma\left(\frac{4}{3}\right)} e^{-\eta^3}
\]

\[
+ \frac{\Gamma\left(\frac{5}{3}\right)}{280[\Gamma\left(\frac{4}{3}\right)]^3} \int_0^1 \frac{x^{1/3}}{(1-x)^{2/3}} \exp\left(\frac{-\eta^3}{1-x}\right) dx .
\]  
(A-21)

Using the 3-term expansion for \( \theta \), one can produce a 3-term Lévêque series for the Nusselt number, where \( Nu \), based on the hydraulic diameter, is related to the derivative of \( \theta \):
The partial derivative, \( \frac{\partial \theta}{\partial \eta} \), can be evaluated term by term, but it is somewhat difficult to evaluate \( \theta'_2(0) \). The trick is to make the variable substitution,

\[
t = \frac{\eta^3}{1-x}, \quad dt = \frac{\eta^3}{(1-x)^2} \, dx,
\]

so that \( \theta'_2(0) \) becomes

\[
\theta'_2(0) = -3K \int_0^\infty t^{-1/3} e^{-t} \, dt = \frac{-9}{2} K \Gamma\left(\frac{5}{3}\right),
\]

where \( \Gamma\left(\frac{5}{3}\right) = 0.90275 \). The resulting 3-term Lévêque series is

\[
Nu = \frac{4}{\Gamma\left(\frac{4}{3}\right)} \left(\frac{2}{9}\right)^{1/3} \zeta^{-1/3} - \frac{4}{10} - \frac{18\left(\frac{9}{2}\right)^{1/3}}{280} \frac{[\Gamma\left(\frac{5}{3}\right)]^2}{[\Gamma\left(\frac{4}{3}\right)]^3} \zeta^{1/3}
\]

\[
= 2.7131949 \zeta^{-1/3} - 0.4 - 0.12146690 \zeta^{1/3},
\]

or

\[
= 1.8488 \left( \frac{ReScd_x}{L} \right)^{1/3} - 0.4 - 0.2005 \left( \frac{L}{ReScd_x} \right)^{1/3}. \quad (A-25)
\]

The first two terms of this equation agree with the two terms given in problem 17-1 of reference 28.
Appendix B. Numerical Solution of the Asymmetric Graetz Problem

The finite-difference method developed by Newman, for solving coupled, ordinary
differential equations, has been described elsewhere. Here, the method is applied to
the eigenvalue problem discussed in sections 2.4 and 2.5.

Equations 2-11 and 2-12 may be written in the form

\[ \frac{d^2 y_1}{dx^2} + y_2(1-x^2)y_1 = 0 \]  
\[ \frac{dy_2}{dx} = 0 . \]  

Equation B-1 may be linearized by setting

\[ y_1 = y_1\hat{} + \Delta y_1 \]  
\[ y_2 = y_2\hat{} + \Delta y_2 . \]  

Here, \( y_1 \) is the exact solution, and \( y_1\hat{} \) is an approximate solution or guess. Substituting

B-3 and B-4 into B-1 and neglecting terms of order \( (\Delta y)^2 \) gives

\[ \frac{d^2 y_1\hat{}}{dx^2} + \frac{d^2 \Delta y_1}{dx^2} + y_2\hat{}(1-x^2) + y_2\hat{}(1-x^2)\Delta y_1 + y_1\hat{}(1-x^2)\Delta y_2 = 0 , \]  

or

\[ - \left[ \frac{d^2 \Delta y_1}{dx^2} + y_2\hat{}(1-x^2)\Delta y_1 + y_1\hat{}(1-x^2)\Delta y_2 \right] = \frac{d^2 y_1\hat{}}{dx^2} + y_2\hat{}y_1\hat{}(1-x^2) . \]  

Similarly, equation B-2 becomes

\[ \frac{d \Delta y_2}{dx} = -\frac{dy_2\hat{}}{dx} . \]  

Both equations B-6 and B-7 are of the form

\[ \sum_k a_{ik}(x) \frac{d^2 y_k}{dx^2} + b_{ik}(x) \frac{dy_k}{dx} + c_{ik}(x) y_k = g_i(x) , \]

where

\[ a_{11} = -1 \]
\[ a_{12} = 0 \]
The differential equations represented by B-8 can be written in finite-difference form:

\[ \sum_k A_{ik}(j)Y_k(j-1) + B_{ik}(j)Y_k(j) + D_{ik}(j)Y_k(j+1) = G_i(j). \]  

(B-10)

Note that this is a banded-matrix equation for the vector of \( Y_i \)'s. If central differences are used, then

\[
\begin{align*}
A_{ik}(j) &= a_{ik} - \frac{h}{2} b_{ik} \\
B_{ik}(j) &= -2a_{ik} + h^2 c_{ik} \\
D_{ik}(j) &= a_{ik} + \frac{h}{2} b_{ik} \\
G_i(j) &= h^2 g_i.
\end{align*}
\]  

(B-11)

These equations can be used to put equation B-6 in central-difference form. For equation B-7, however, the forward-difference form is accurate to order \( h^2 \) because \( y_2 \) is a constant. This eliminates the need for image points.

Because the eigenfunctions are all even or odd, it is necessary only to solve the problem in half of the domain, that is, from \( x=0 \) to \( x=L \). For the even eigenfunctions, the boundary condition is \( y_1' = 0 \) at \( x=0 \), and for the odd eigenfunctions, \( y_1 = 0 \) at \( x=0 \),
where \( x=0 \) corresponds to the center of the channel. For all eigenfunctions, \( y_1=0 \) at \( x=1 \), and the normalization condition, \( y'_1=1 \) at \( x=1 \), is applied.

To obtain the coefficients \( A_{ij}, B_{ij}, D_{ij}, \) and \( G_i \) at the endpoints \( j=1 \) and \( j=j_m \), the boundary conditions are needed. For simplicity, we treat the even and odd eigenfunctions separately.

For the even eigenfunctions at \( j=1 \), \( y'_1 \) is zero. To avoid the need for an image point, the differential equation B-6 is written at \( x=0 \):

\[
-\frac{d^2\Delta y_1}{dx^2}(1) - y_2^2(1)\Delta y_1(1) - y_1^2(1)\Delta y_2(1) = \frac{d^2y'_1}{dx^2}(1) + y_2^2(1)y'_1(1) .
\]

The second derivative is written in central-difference form to ensure accuracy to order \( h^2 \), but the boundary condition \( y'_1=0 \) can be used to eliminate the image point at \( j=0 \) because \( y_1(2)=y_1(0) \). Therefore the second derivative is

\[
\frac{d^2y_1}{dx^2}(1) = \frac{2}{h^2} [y_1(2)-y_1(1)] .
\]

Equation B-7 remains the same, because there is no boundary condition on the eigenvalue, \( y_2 \).

For the odd eigenfunctions at \( j=1 \), the boundary condition is \( y_1=0 \). Therefore,

\[
-\Delta y_1(1) = y'_1(1) .
\]

Again, equation B-7 remains the same.

For \( j=j_m \), the boundary conditions are the same for both the even and odd eigenfunctions: \( y'_1=1 \) and \( y_1=0 \). From the boundary condition \( y_1=0 \),

\[
-\Delta y_1(j_m) = y'_1(j_m) .
\]

For the boundary condition \( y'_1=1 \), the backward-difference form is accurate to order \( h^2 \), because the second derivative is zero at \( x=1 \):
Therefore, \[ \left[ \frac{d^2y_1}{dx^2} + y_2(1-x^2)y_1 \right]_{x=0} = \frac{d^2y_1}{dx^2} \bigg|_{x=0} = 0. \] (B-16)

Now that all the equations and boundary conditions have been written in the form of equation B-10, the coefficients \(A_{ij}, B_{ij}, D_{ij}\), and \(G_i\) can be calculated. Subroutines BAND and MATINV (see appendix C) can then be used to solve for the \(\Delta y_i\)'s, which are used to obtain new guesses \(y_i^1\) and \(y_i^2\). This iteration over the nonlinearities is continued until the desired degree of convergence is reached.

\[ \frac{1}{h} (\Delta y_1(j_m) - \Delta y_1(j_{m-1})) = -\frac{1}{h} (y_i^1(j_m) - y_i^1(j_{m-1})) + 1. \] (B-17)
Appendix C. Program EIGEN
This program solves the eigenvalue problem for the asymmetric Graetz problem in a channel. (See V. Edwards and J. Newman, Int. J. Heat Mass Trans., Vol. 28 (1985), 503-505.)

This program modifies sin and lambda initial guesses

1 READ 101, L,NJ
print 112, 1, nj
IF(NJ.EQ.O) STOP

N is the number of equations
H is the step size

N=2
H=1.0/(NJ-1)
PRINT 105
MOD=1
IF(L.EQ.2*(L/2)) MOD=0
pi = 3.141592654

Modify the slope of sqrt(lambda) vs. L to match the 7th eigenvalue (to provide a better initial guess for the higher eigenvalues)

DO 2 J=1,NJ
XI(J) = H*(J-1)

c xi2 is the weighting function for calculating the coefficients
c
XI2(J) = 1.0 - XI(J)**2
C(1, J) = SIN(2*FLOAT(NJ-J)*H)/4*

C(1,J) is the eigenfunction
al(l) = C(2, J) is the eigenvalue (lambda)

al(1) = C(2, J)
if((2*(1/2)).ne.1) go to 22
if(2*(1/4).eq.(l/2)) C(l, J) = -C(l, J)
go to 23
22 if(2*(1-l)/4.eq.(l-1)/2) C(l, J) = -C(l, J)
23 continue
2 C(2, J) = AL(1)
JCOUNT = 0
3 JCOUNT = JCOUNT + 1
J = 0

No image points because the boundary conditions are incorporated into
the equations at j-1 and j=nj

DO 4 I=1,N
DO 4 K=1,N
Y(I, K) = 0.0
4 X(I, K) = 0.0
AL0 = AL(1)
COLD(1, NJ) = C(1, NJ)
5 J = J+1

 initialization

DO 6 I=1,N
G(I) = 0.0
DO 6 K=1,N
A(I, K) = 0.0
B(I, K) = 0.0
6 D(I, K) = 0.0
IF(J.EQ.NJ) GO TO 8

equation for eigenvalue

G(2) = C(2, J+1) - C(2, J)
COLD(1, J) = C(1, J)
B(2, 2) = 1.0
D(2, 2) = -1.0
IF(J.GT.1)GO TO 7
IF(MOD.EQ.0)GOTO 11

The following is for j=1, odd L
BC is y'-0 at xi=0
Second equation is written in forward-difference form

G(1)=2.0*(C(1,J+1)-C(1,J))/H/H+C(2,J)*C(1,J)
D(1,1)=-2.0/H/H
B(1,1)=D(1,1)-C(2,J)
B(1,2)=-C(1,J)
GO TO 12

The following is for j=1, even L
BC is y=0 at xi=0

11 G(1)=C(1,J)
B(1,1)=1.0
12 CALL BAND(J)
GO TO 5

The following is for j=2 to (nj-1)

7 G(1)=(C(1,J+1)+C(1,J-1)-2.0*C(1,J))/H/H+XI2(J)*C(2,J)*C(1,J)
A(1,1)=-1.0/H/H
D(1,1)=\( (1,1) \)
B(1,2)=-XI2(J)*C(1,J)
B(1,1)=-2.0*A(1,1)-XI2(J)*C(2,J)
CALL BAND(J)
GO TO 5

The following is for j=nj

8 G(1)=C(1,J)
B(1,1)=1.0
G(2)=2.0*(C(1,J-1)-C(1,J))/H/H+2./H
B(2,1)=2.0/H/H
A(2,1)=-B(2,1)
CALL BAND(J)

Update the guesses

AL(1)=C(2,1)+AL(1)
DO 13 J=1,NJ
C(1,J)=C(1,J)+COLD(1,J)
13 C(2,J)=AL(1)
IF(JCOUNT.GT.10)GO TO 18

137
If not converged, next jcount

IF(ABS(AL(1)-AL0).GT.ABS(AL(1))*1.0E-10) GO TO 3
DO 19 j=1,nj
   IF(abs(c(1,j)-cold(1,j)).GT.abs(c(1,j))*1.0e-5) GO TO 3
19 CONTINUE

SAVE Y(1)

CSA(L,1)=C(1,1)
SUM1=C(1,1)*H/3.
SUM2=SUM1*C(1,1)
DO 10 JJ=3,NJ,2
   J=JJ-1
10 SUM1-SUM1+ADD

Use Simpson's rule integration for coefficients

even j:

CSA(L,J)=C(1,J)
ADD=XI2(J)*C(1,J)*4.0*H/3.0
SUM2=SUM2+ADD*C(1,J)
IF(MOD.EQ.0) ADD=ADD*XI(J)
SUM1=SUM1+ADD
J=J+1

odd j:

ADD=XI2(J)*C(1,J)*2.0*H/3.0
SUM2=SUM2+ADD*C(1,J)
IF(MOD.EQ.0) ADD=ADD*XI(J)
CSA(L,J)=C(1,J)
10 SUM1=SUM1+ADD

The AM are the coefficients

AM(1)=SUM1/(2.0*SUM2)
IF(MOD.EQ.0) AM(1)=-AM(1)
IF(1.ne.lmax) GO TO 1

PRINT 108
ZETA = 0.0
DZ = 0.001
DO 26 IZ=1,31
   IF(IZ.GE.11) DZ=0.01
   IF(IZ.GE.20) DZ=0.1
   IF(IZ.GE.29) DZ=1.0
26 PRINT CONCS AT A FEW XI'S:
PRINT 113
EIGEN.FOR

do 29 i=1,6
conc(i) = 0.0
29 conc(i) = 0.0
istep = (nj-1)/5
do 32 j=1,nj,istep
iy = 1 + (j-1)/20
do 30 i=1,lmax
pml = 1
if(i.eq.2*(i/2)) pml= -1
conc(iy) = conc(iy) + am(i)*exp(-al(i)*zeta)*csa(i,j)
     *pml
30 conc(iy) = conc(iy) + am(i)*exp(-al(i)*zeta)*csa(i,j)
conc(iy) = conc(iy) + 0.5*(1.0 + xi(j))
32 conc(iy) = conc(iy) + 0.5*(1.0 - xi(j))
print 114,zeta,(conc(i),i=1,6),(concb(i),i=1,6)

sum1,sum2 are fluxes:
sum1 = 0.0
c
sum2 = 0.0
c
do 25 i=1,lmax
c
ali = al(i)
c
sum1 = sum1 + am(i)*exp(-ali*zeta)
c
25 sum2 = sum2 - abs(am(i))*exp(-ali*zeta)
c
sum1 = 1.0 - 2.0*sum1
c
sum2 = 1.0 - 2.0*sum2
c
26 zeta = zeta + dz
go to 21

18 print 109,jcount
21 stop
END
SUBROUTINE BAND(J)
COMMON A(2,2),B(2,2),C(2,801),D(2,5),G(2),X(2,2),Y(2,2),N,NJ
DIMENSION E(2,3,801)
c
This program solves coupled ODEs by casting them into finite
difference form and solving the resulting banded matrix
c
101 FORMAT (15H DETERM=0 AT J=,I4)
   IF (J-2) 1,6,8
1 NP1= N + 1
   DO 2 I=1,N
   D(I,2*N+1)= G(I)
   DO 2 L=1,N
   LPN = L + N
2   D(I,LPN)= X(I,L)
   CALL MATINV(N,2*N+1,DETERM)
   IF (DETERM) 4,3,4
3 PRINT 101, J
4 DO 5 K=1,N
EIGEN.FOR

E(K,NP1,1) = D(K,2*N+1)
DO 5 L=1,N
E(K,L,1) = - D(K,L)
LPN = L + N
5 X(K,L) = - D(K,LPN)
RETURN

DO 7 I=1,N
DO 7 K=1,N
DO 7 L=1,N
D(I,K) = D(I,K) + A(I,L)*X(L,K)
8 IF (J-NJ) 11,9,9
9 DO 10 I=1,N
DO 10 L=1,N
G(I) = G(I) - Y(I,L)*E(L,NP1,J-2)
DO 10 M=1,N
A(I,L) = A(I,L) + Y(I,M)*E(M,L,J-2)
10 IF (J-NJ) 12,11,11
11 DO 12 I=1,N
D(I,NP1) = - G(I)
DO 12 L=1,N
D(I,NP1) = D(I,NP1) + A(I,L)*E(L,NP1,J-1)
DO 12 K=1,N
B(I,K) = B(I,K) + A(I,L)*E(L,K,J-1)
CALL MATINV(N,NP1,DETERM)
IF (DETERM) 14,13,14
13 PRINT 101, J
14 DO 15 K=1,N
DO 15 M=1,NP1
15 E(K,M,J) = E(K,NP1,J)
DO 18 JJ=2,NJ
M = NJ - JJ + 1
DO 18 K=1,N
C(K,M) = E(K,NP1,M)
DO 18 L=1,N
18 C(K,M) = C(K,M) + E(K,L,M)*C(L,M+1)
DO 19 L=1,N
DO 19 K=1,N
19 C(K,1) = C(K,1) + X(K,L)*C(L,3)
20 RETURN
END

SUBROUTINE MATINV(N,M,DETERM)
DIMENSION A(2,2),B(2,2),C(2,801),D(2,5),ID(2)
COMMON A,B,C,D

This program inverts matrix B by pivoting. The D matrix is n x (2n+1) for use by subroutine BAND
DETERM = 1.0
DO 1 I = 1,N
   ID(I) = 0
   DO 18 NN = 1,N
      BMAX = 1.1
      DO 6 I = 1,N
         IF (ID(I).NE.0) GOTO 6
         BNEXT = 0.0
         BTRY = 0.0
         DO 5 J = 1,N
            IF (ID(J).NE.0) GOTO 5
            IF (ABS(B(I,J)).LE.BNEXT) GOTO 5
            BNEXT = ABS(B(I,J))
            BTRY = ABS(B(I,J))
            JC = J
         5 CONTINUE
         IF (BNEXT .GE. BMAX*BTRY) GOTO 6
         BMAX = BNEXT/BTRY
      6 IROW = I
      JCOL = JC
      CONTINUE
      IF (ID(JC).EQ.0) GOTO 8
      DETERM = 0.0
      RETURN
   8 ID(JCOL) = 1
   IF (JCOL.EQ.IROW) GOTO 12
   DO 10 J = 1,N
      SAVE = B(IROW,J)
      B(IROW,J) = B(JCOL,J)
   10 B(JCOL,J) = SAVE
   DO 11 K = 1,M
      SAVE = D(IROW,K)
      D(IROW,K) = D(JCOL,K)
   11 D(JCOL,K) = SAVE
   F = 1.0/B(JCOL,JCOL)
   DO 13 J = 1,N
      B(JCOL,J) = B(JCOL,J)*F
   13 DO 14 K = 1,M
   14 D(JCOL,K) = D(JCOL,K)*F
   DO 18 I = 1,N
      IF (I.EQ.JCOL) GO TO 18
      F = B(I,JCOL)
   18 DO 16 J = 1,N
   16 B(I,J) = B(I,J) - F*B(JCOL,J)
   DO 17 K = 1,M
17 \quad D(I,K) \rightarrow D(I,K) - F \cdot D(JCOL,K)
18 \quad \text{CONTINUE}
19 \quad \text{RETURN}
20 \quad \text{END}
Appendix D. Iterative Methods

D.1. Introduction

To solve for the concentration, current, and potential distributions in an electrochemical system, a set of coupled differential equations must be solved. Although the differential equations can be linear, as in the channel problem with negligible migration, the boundary conditions are often nonlinear because the kinetics depend exponentially on the potential driving force, which varies across the electrode surfaces. Thus, a set of nonlinear equations must be solved, and iteration is required.

The iteration can be viewed as a loop because variables are passed through a set of equations until new guesses for the same variables emerge. If the process is repeated with these new variables, the iteration is called successive substitution or Picard’s method. Usually, however, these new variables are operated on by an iterating function, which, in the most general case, uses information from previous iterations to produce new guesses. Thus, one can think of “cutting” the loop to update the variables. In general, it is best to cut the loop where there are the fewest variables, to simplify and speed the updating procedure.

The iterating function for a vector equation \( \mathbf{f}(\mathbf{x}) = 0 \) may be written

\[
\mathbf{x}_{i+1} = \phi(\mathbf{x}, \mathbf{f}), \tag{D-1}
\]

where \( \mathbf{x}_{i+1} \) is the \((i+1)\)st approximant to \( \mathbf{x} \). \( \phi \) may depend on the previous approximants to \( \mathbf{x} \) and the values of \( \mathbf{f} \) and its derivatives. Here, we shall discuss some special cases — the successive substitution method, the Newton-Raphson method, and the secant method (see, for example, references 61, 62, 100, and 101).
D.2. Comparison of Iterative Methods

The successive substitution method applied to $x = g(x)$ is

$$x_{i+1} = g(x_i).$$  \hfill (D-2)

A variation on this method is successive substitution with damping:

$$x_{i+1} = x_i + k(g(x_i) - x_i).$$  \hfill (D-3)

The successive substitution method is easy to program, but its convergence is slow (first order), and sometimes it will not converge at all.\textsuperscript{51} The speed of convergence can be increased by using a second-order method such as the Newton-Raphson method

$$x_{i+1} = x_i - (df(x_i))^{-1}f(x_i),$$  \hfill (D-4)

where $df(x_i)$ is the Jacobian of $f$. Second-order convergence means that the error at a given iteration is proportional to the square of the error at the previous iteration,

$$e_{i+1} = C_2 e_i^2,$$  \hfill (D-5)

and that the number of significant figures, $n_i$, doubles with each iteration. With a first-order method, on the other hand, the number of significant figures increases linearly:\textsuperscript{52}

$$n_i = -i \log C_1 + n_0,$$  \hfill (D-6)

where $C_p$ is the asymptotic error constant defined, for $p$th order convergence, as

$$e_{i+1} = C_p e_i^p.$$  \hfill (D-7)

If $C_p$ is close to unity, the convergence is slow.

Another way to examine the convergence order is to plot the errors so that the curve becomes linear as quadratic convergence is approached. Typically, a plot of this type will show that, in the early iterations, the truncated Taylor-series approximation of the function is inaccurate; then as the solution is approached, the convergence becomes quadratic within the limit of the machine accuracy.

Figure D-1 is a plot for a Newton-Raphson iteration of $\ln | \ln \frac{e_{i+1}}{e_i} |$ vs. iteration.
Figure D-1. Plot of absolute errors for a Newton-Raphson iteration.
Figure D-2. Plot of relative errors for a Newton-Raphson iteration.
tion number, \( i \), and figure D-2 is a plot of \( \ln \left| \frac{x_i - x_{i+1}}{x_{i-1} - x_i} \right| \), where \( x_i - x_{i+1} \) is a "local" estimate of the error \( e_{i+1} \). The motivation for plotting the log of the log of the error comes from the definition of quadratic convergence (equation D-5). By recursion, equation D-5 is rewritten \( e_i = C_2^{(2^i-1)} \varepsilon_0 \ 2^i \), and further manipulation yields

\[
\ln \left| \ln \left| \frac{e_{i+1}}{e_i} \right| \right| = i \ln 2 + \ln \ln C_2 + \ln \varepsilon_0.
\]

For the more general convergence behavior, \( e_{i+1} = C_p e_i^p \),

\[
\ln \left| \ln \left| \frac{e_{i+1}}{e_i} \right| \right| = i \ln p + \ln \ln C_p + \ln \varepsilon_0^{p-1}.
\]

Thus the slope of the plot of \( \ln \left| \frac{e_{i+1}}{e_i} \right| \) indicates the order of convergence \( p \). (The slope is \( \ln p \).) In figures D-1 and D-2, the numbers next to each point are the exponentials of the slope; they approach 2.0 as the solution is approached quadratically and deviate in the last iteration owing to roundoff error. In figures D-3 and D-4, the iteration was by successive substitution, so the convergence is first order. The calculations for figures D-1 through D-4 were carried out in single precision on a CDC-7600 (16 significant figures).

Clearly, the Newton-Raphson method has the advantage of fast convergence. The disadvantage is that the method requires the evaluation and inversion of the Jacobian (the matrix of partial derivatives of \( f \)). An alternative to calculating the derivatives analytically is to calculate them numerically with a suitable step size ("discretized Newton iteration") or to use the same inverted Jacobian for several iterations. Another approach is to estimate the derivatives based on information from the previous two iterations. For example, the secant method is
Figure D-3. Plot of absolute errors for successive substitution.
Figure D-4. Plot of relative errors for successive substitution.
\[ x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} (x_i - x_{i-1}) \] (D-10)

which is easily generalized to multidimensional problems.\textsuperscript{100, 101} The advantage of the secant method is that \( f' \) is not needed, but, on the other hand, the disadvantages are that convergence is slower than quadratic (order 1.62) and more storage is required.

Traub\textsuperscript{62} discusses the concept of "computational efficiency" to account for the cost of evaluating \( f \) and its derivatives and the number of iterations required for convergence to a given number of significant figures. For example, for a one-dimensional problem, if the cost of calculating the first derivative of \( f \) is more than 0.44 times the cost of calculating \( f \), then the secant method is "cheaper" than the Newton-Raphson method.\textsuperscript{101} In the one-dimensional case, the secant method is cheaper than the successive substitution method because both methods require just one evaluation of \( f \) at each iteration and the secant method converges faster. In the \( n \)-dimensional case, the comparison should consider the cost of calculating the entries of the Jacobian matrix and the cost of inverting it. Thus, for a large system of equations, the successive substitution method could be cheaper than a higher-order method.

D.3. Application of Some Iterative Methods to the Rotating-Disk Problem

The rotating-disk problem was used as an example problem to assess the feasibility of applying the Newton-Raphson method to the overpotential iterations in the thin-gap flow cell problem. The rotating-disk problem is described elsewhere,\textsuperscript{102} but is presented here in summary.

Given a potential distribution, one can calculate the concentration and current distribution at the surface of a rotating-disk electrode by equating the Fick's-law and Faraday's-law expressions for the flux. The resulting current distribution is then substituted into Laplace's equation to calculate a new potential distribution. The solution to Laplace's equation evaluated at the disk surface represents the potential in the solution.
just outside the double layer with the prevailing current distribution, but without concentration variations. For a rotating disk, this potential is

\[ \Phi_\circ(r) = \sum_{n=0}^{\infty} B_n P_{2n}(\eta), \]

where \( P_{2n} \) is a Legendre polynomial of order \( 2n \) and \( \eta \) is a rotational elliptic coordinate evaluated at the disk surface (see reference 102). The coefficients \( B_n \) are calculated from the current distribution:

\[ B_n = [P_{2n}(0)]^2 \frac{(4n+1)\pi r_o}{2\kappa_\infty} \int_0^1 \eta P_{2n}(\eta) d\eta. \]

If the total overpotential at the center of the disk,

\[ E(0) = V - \Phi_\circ(0), \]

is fixed, then the total overpotential, \( E = V - \Phi_\circ \), at any other radial position is

\[ E = V - \Phi_\circ - \sum_{n=0}^{\infty} B_n [P_{2n}(\eta) - P_{2n}(1)]. \]

Thus, a loop can be constructed in which a potential distribution produces a current distribution through the flux equations, and the current distribution, in turn, produces a set of \( B \) coefficients and a new potential distribution (equations D-9 through D-12).

**D.3.1. Cutting the Loop**

To apply a method of successive approximation to the rotating-disk problem, one must decide where to "cut" the iterative loop, that is, one must decide which variables are to be updated by the iterating function. For the rotating-disk problem, it is best to cut the loop at the \( B \) coefficients. The channel problem, however, does not have \( B \) coefficients, so the possibility of cutting the loop at the overpotentials, \( E(r) \), was also considered.
D.3.2. Calculating the Jacobian

Because the successive substitution method is straightforward, we shall concentrate
our discussion on the Newton-Raphson method, which is more complex because the
Jacobian matrix must be calculated (either numerically or analytically). Although
numerical derivatives are easier to program than the analytic derivatives, difficulties
may arise because one must choose an appropriate step size to use in the difference cal-
culations. This step size must be large enough to avoid roundoff error, but small enough
that the derivative of the function is represented accurately by the numerical approxi-
mation. In spite of this disadvantage, the numerical derivatives are useful, and they
provide an easy check for programming errors in calculating the analytic derivatives.

To calculate the analytic derivatives in a complex problem, one can envision two
subroutines — one calculates the value of a function \( f_i(x) \), and the other calculates
\( \frac{\partial f_i}{\partial x_j} \). The second subroutine can be regarded as the "derivative" of the first subrou-
tine. To write the derivative subroutine, one would work with new variables that
represent the partial derivatives of the original variables with respect to \( x_j \).

Sample programs to illustrate the concept of "differentiating" a computer program
are given in figures D-5 and D-6, and the results are plotted in figures D-1 through D-4.
Program EXAMPLE (figure D-5) uses successive substitution, and program DEXAMPL
(figure D-6) uses the Newton-Raphson method with analytic derivatives. In this simple
one-dimensional example, there are three lines of code to calculate \( F \), and, in program
DEXAMPL, three lines of code to calculate \( DF \), the \( 1 \times 1 \) matrix of derivatives. Note
that these three lines could be written in a separate subroutine, as was done for the
rotating-disk problem.

Appendix E lists the programs for the rotating-disk problem. The programs are
revisions of a program written by Newman to solve for the current distribution on a
rotating disk below the limiting current using the method of successive substitution with
Sample program to solve $x = (\exp(x^2) - 0.5)/3$.

by Newton's method

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

dimension x(51)

40 format(' no convergence in',i4,' iterations')
50 format(' iteration ', x')
60 format(1x,i3,5x,lpel7.10)

itmax = 50

iter = 1
x(iter) = 0.d0
print 60, iter, x(iter)
do 20 iter = 1, itmax

xold = x(iter)
a = x(iter)**2
b = dexp(a)
f = (b - 0.5)/3.

c-----------------------------
da = 2.*x(iter)
db = b*da
df = db/3.
c-----------------------------
c

x(iter+1) = xold + 1./(1. - df) * (f - xold)

print 60, iter, x(iter+1)
20 if (dabs(x(iter+1)-xold) .le. 1.d-10*dabs(x(iter+1)))
   go to 25

print 40, itmax
25 stop
end

Figure D-5. Sample program for successive substitution.
Sample program to solve \( x = \frac{\exp(x^2) - 0.5}{3} \) by successive substitution.

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

dimension x(51)

\begin{verbatim}
40 format(‘ no convergence in’ ,i4, ’ iterations’)
50 format(’ iteration x’)
60 format(lx,i3,5x,lpel7.10)
\end{verbatim}

d----------------------------------------
damp = 1.d0
d----------------------------------------

c
iter = 1
x(iter) = 0.d0
print 60, iter, x(iter)
do 20 iter = 1, itmax

xold = x(iter)
a = x(iter)**2
b = dexp(a)
f = (b - 0.5)/3.

c----------------------------------------
x(iter+1) = xold + damp*(f-xold)
c----------------------------------------

c
print 60, iter, x(iter+1)
20 if (dabs(x(iter+1)-xold) .le. 1.d-10*dabs(x(iter+1)))
go to 25

print 40, itmax
25 stop
end

Figure D-6. Sample program for Newton-Raphson method.
damping. Program CURDB uses the Newton-Raphson method to iterate on the $B$ coefficients, and program CURDE iterates on the overpotentials, $E(r)$. Programs CURDBN and CURDEN use numerical derivatives; they were written to check for programming errors in computing the analytic derivatives.

Although the converged results of programs CURD, CURDB, and CURDE are the same, the computer time and the number of iterations are different. In general, the Newton-Raphson method takes fewer iterations but more computer time than successive substitution (see table D-1). Note that the computer time for the Newton-Raphson method depends on the size of the Jacobian, not only because it is expensive to calculate the derivatives, but also because the "cost" of inverting an $n \times n$ matrix is proportional to $n^3$. It is because of the large Jacobian that the successive substitution method is more attractive for the rotating-disk problem.

D.4. Iterative Methods for the Channel Problem

D.4.1. Successive Substitution

Based on our comparison of successive substitution to the Newton-Raphson method for the rotating-disk problem, we chose to use successive substitution with damping for the channel problem. Unfortunately, several problems would not converge with this method. Some improvement was made with the scheme discussed in chapter 1 for

<table>
<thead>
<tr>
<th></th>
<th>Successive Substitution</th>
<th>(Newton)$_B$</th>
<th>(Newton)$_E$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterations</td>
<td>41</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>CP Seconds</td>
<td>0.693</td>
<td>1.739</td>
<td>2.943</td>
</tr>
</tbody>
</table>
adjusting the damping factor as the iterations proceed, but the computer time was significant (several hundred seconds on a CDC-7600 or about 20 minutes on a VAX-8600, given a good initial guess).

D.4.2. Newton-Raphson with a Global Approximation Method

The reason the successive substitution method had seemed more attractive for the channel problem than a Newton-Raphson method was that the Newton-Raphson method would be too time-consuming because of the large Jacobian. For example, with 51 mesh points on each electrode, the Jacobian would be $102 \times 102$.

Fortunately, the problem of large Jacobians can be circumvented by using a global approximation method, such as orthogonal collocation, which is a method of weighted residuals. The idea of collocation is to express the solution as a sum of trial functions multiplied by coefficients that are determined by minimizing the error between the approximate solution and the exact solution. The advantage of a global approximation, compared to a local approximation, is that the Jacobian is small; only a small set of coefficients is required, instead of a large set of variables evaluated at mesh points. A good approximation to the solution between the collocation points is available from the global trial functions and the coefficients. Naturally, the accuracy of the solution between collocation points depends on the choice of approximating functions (see section D.5.2).
D.5. Collocation Applied to the Channel Problem

D.5.1. Introduction

Cabán and Chapman\textsuperscript{22} used orthogonal collocation with polynomial trial functions to solve the problem, originally solved by Parrish and Newman\textsuperscript{21}, of current distribution in a channel flow cell with thin boundary layers. With just seven collocation points on each electrode, they could approximate with reasonable accuracy the current distribution along the electrodes. One advantage of their method is that few mesh points are needed, so the Jacobian is small enough to use the Newton-Raphson method, which has second-order convergence. Another important advantage they mention is that the numerical integrations — Laplace’s equation and the superposition integrals — are performed once, instead of repeatedly, as they were in the successive-substitution method used here (see chapter 1) and in Parrish and Newman’s work. The successive substitution method requires repeated integrations for the potentials and fluxes because the integrands contain unknowns. Cabán and Chapman’s method, on the other hand, uses known approximating polynomials for the current, and the iteration is on the unknown coefficients, which can be taken outside the integrals. Because the trial functions are polynomials, however, their method cannot predict accurately the singular behavior of the current distribution near the electrode edges in cases such as the primary or limiting current distribution. Other investigators\textsuperscript{106-108} have addressed this problem by using functions that correspond to known solutions of the governing equations. For example, Rhee\textsuperscript{106} solved the problem of asymmetric suction and heating in a flow channel with one porous wall by superposing functions that have a step-change in temperature or flux on one wall. Miksis\textsuperscript{107,108} solved the problem of primary current distribution in a rotating ring/disk system by adding to a collection of Legendre polynomials some special functions to match the singularities at the electrode edges.
To apply Miksis' method to the channel problem, one must decide which variables to represent as a sum of trial functions, e.g. current or potential. At first, we tried to preserve the method of chapter 1; therefore the only choice was to approximate the potentials. After realizing that this method would still be time-consuming (because the integrations are carried out inside the nested iteration loops), we decided to restructure the iteration procedure to take advantage of the linearity of Laplace's equation and Fick's law, as done by Cabán and Chapman. Thus, the integrals are evaluated once with the trial functions, and the subsequent iteration is over the nonlinearities in the Butler-Volmer equation.

There are many possible ways to structure the iteration scheme for the channel problem. One question is which variables to approximate — current, concentration, potential, or a mixture. A related question is how to write the superposition integrals. For example, Cabán and Chapman use trial functions for the current and write the concentration as a superposition integral over the currents.

Several factors determine which variables are best to approximate. One consideration is the ease of solving for the other variables. For example, if potentials were approximated, it would be difficult to extract the currents from Laplace's equation. Another factor is the accuracy of the approximation. Because concentration and flux are related by an integral equation, instead of by a direct proportionality, a set of current polynomials does not produce a set of concentration polynomials, and vice versa. This means that a finite set of polynomials for one variable may not adequately approximate the other variables; for example a set of current polynomials cannot accurately represent the limiting current, which has a step change in concentration. A third criterion, which makes current functions attractive, is that some special cases are known, such as the primary and limiting current distributions (constant-potential and constant-concentration boundary conditions, respectively). These limiting cases may be
used to supplement the collection of orthogonal polynomials, as was done by Miksis.\textsuperscript{107, 108}

**D.5.2. Choice of Trial Functions**

A good choice of approximating functions can increase the accuracy, particularly for limiting cases such as the primary or limiting current distribution, where a series of orthogonal polynomials is inadequate to represent the singularity at the electrode edge. Accordingly, we adopt a method similar to that of Miksis,\textsuperscript{107, 108} wherein special functions are added to the collection of orthogonal polynomials.

The trial functions chosen were a set of Legendre polynomials, an approximation to the limiting-current distribution, a function that is similar to the limiting current, but finite at the leading edge, and two functions that each approximate the behavior of a secondary current distribution on one edge only. For the limiting-current function, we superposed the Graetz functions from chapter 2 to produce a function that corresponds to a step change in concentration of the limiting species on both electrodes. Stepping to zero on both electrodes has the advantage that the current has the proper singular behavior at the leading edge, but is small away from the region of interest. The second special function is simply

\[ i(x) = (x + \Delta_1)^{-1/3}, \quad (D-15) \]

a modification of the Lévêque solution, and the last two functions are

\[ i(x) = (x + \frac{\Delta_2}{L/h + 1})^{-1/2} \quad (D-16) \]

and

\[ i(x) = (1 - x + \frac{\Delta_3}{L/h + 1})^{-1/2}. \quad (D-17) \]

The parameters \( \Delta \) are used to make the current finite at the electrode edges, and they may be chosen based on an estimate of the expected current at the edge. The aspect
ratio appears in the denominator because the size of the region whose current deviates from the primary distribution depends on the geometry of the system.

**D.5.3. Basic Iteration Scheme**

We can now discuss the basic iteration scheme, which is built around approximating the partial current densities, \( i_j \), as superpositions of trial functions. First the linear calculations are carried out; then the Newton-Raphson method iterates to find the coefficients. We shall later mention some revisions to this basic method that can improve the convergence behavior, particularly for limiting-current problems.

Recall that the advantage of the new scheme over successive substitution is that the linear integral equations are evaluated only once. These equations are the two flux expressions, Fick's and Faraday's laws, and Laplace's equation for the potential. Although it might seem convenient to follow Cabán and Chapman's method and write the Fick's-law flux as a superposition integral with the fluxes (currents) inside the integrals, we chose not to do so because their equation incorporates the thin-boundary-layer assumption. Since we had already calculated the thick-boundary-layer equation with concentrations inside the integrals, we chose to use the integral (from chapter 1):

\[
N_{i, \text{cat}}(x) = -\frac{D_i}{B} \int_0^x \frac{d\zeta_{i, \text{cat}}}{dz} \left|_x \frac{\partial \theta}{\partial \xi}(\zeta - \zeta', \xi = -1) \right| dx'
+ \frac{D_i}{B} \int_0^x \frac{d\zeta_{i, \text{an}}}{dz} \left|_x \frac{\partial \theta}{\partial \xi}(\zeta - \zeta', \xi = 1) \right| dx'.
\]  
(D-18)

The other linear equations are Faraday's law

\[
N_i = -\sum_j \frac{s_{ij}}{n_j} i_j,
\]  
(D-19)

and the solution to Laplace's equation
\[ \Phi\text{cath}(x) = \Phi^* - \frac{1}{2\pi \kappa_\infty} \left\{ \int_0^L \left[ i_{\text{cath}}(x') \ln \sinh^2 \left( \frac{\pi(x-x')}{2h} \right) \right] dx' \right\} + i_{an}(x') \ln \cosh^2 \left( \frac{\pi(x-x')}{2h} \right) \] \tag{D-20}

The method begins by expressing the partial current densities for each reaction as a sum of coefficients multiplied by trial functions:

\[ i_{j,an}(x) = \sum_{k=1}^{np} A_{j,k,a} i_{j,k,a}(x) \tag{D-21a} \]

and

\[ i_{j,cath}(x) = \sum_{k=1}^{np} A_{j,k,c} i_{j,k,c}(x). \tag{D-21b} \]

This representation will be used to set up the preliminary linear calculations.

Given the entire collection of approximating functions and a guessed set of coefficients, the potentials and concentrations can be calculated from Laplace's equation and the flux expressions, respectively.

Since

\[ i_{an} = \sum_j i_{j,an}, \tag{D-22} \]

we have

\[ i_{an} = \sum_{j=1}^{nrzn} \sum_{k=1}^{np} i_{j,k,a} A_{j,k,a}, \tag{D-23} \]

which we shall rewrite as

\[ i_{an} = \sum_{jka=1}^{(np)(nrzn)} i_{jka} A_{jka}, \tag{D-24a} \]

and

\[ i_{\text{cath}} = \sum_{jkc=1}^{(np)(nrzn)} i_{jkc} A_{jkc}, \tag{D-24b} \]
where $jka$ and $jkc$ are compound indices that collapse the two sums into one. For example, $jka$, the anode index, is defined as

$$jka = (j-1)np + k. \tag{D-25}$$

Equations D-24a and D-24b are the starting point for setting up the linear calculations. Note that each of the trial functions in equations D-24 corresponds to a prescribed partial current density on one electrode and zero current on the other. The coefficients and trial functions are triply subscripted because the partial current density for each reaction on each electrode (subscripts $j$ and $a$) is a different superposition of the trial functions (index $k$). In principle, different trial functions could be used for each partial current and for each electrode. Here, we choose to use the same collection of functions for each electrochemical reaction, $j$, but, in some cases, different functions for the two electrodes (see section D.5.5). The sums in equations D-23 run from $k=1$ to $np$, where $np$ is both the number of trial functions and the number of collocation points. The goal of the method is to solve for the set of coefficients that makes equations D-23 accurate at the collocation points, $x_p$. That is, the approximation to the current must agree with the current computed from the kinetic expression at the collocation points.

It is important to understand that the kinetic expression is nonlinear, but that the expressions for the potential and concentration are linear; therefore one can calculate, once, at the start of the program, the potentials and concentrations resulting from the individual current functions. These concentrations and potentials can then be viewed as trial functions because they are superposed with the same coefficients as those used to generate currents.

This can be seen by substituting into equation D-20 the simplified representations of $i_a$ and $i_c$ (equations D-24):
\[
\Phi_{\text{cath}}(x) - \Phi^* = \sum_{jkc}^L \left( \int_{0}^{L} i_{jkc}(x') \ln \sinh^2 \left( \frac{\pi(x-x')}{2h} \right) dx' \right) A_{jkc}
\]
\[
+ \sum_{jka}^L \left( \int_{0}^{L} i_{jka}(x') \ln \cosh^2 \left( \frac{\pi(x-x')}{2h} \right) dx' \right) A_{jka}
\]
and
\[
\Phi_{\text{an}}(x) - \Phi^* = \sum_{jka}^L \left( \int_{0}^{L} i_{jka}(x') \ln \sinh^2 \left( \frac{\pi(x-x')}{2h} \right) dx' \right) A_{jka}
\]
\[
+ \sum_{jkc}^L \left( \int_{0}^{L} i_{jkc}(x') \ln \cosh^2 \left( \frac{\pi(x-x')}{2h} \right) dx' \right) A_{jkc}.
\]

To calculate the concentrations, we use the two flux expressions — Faraday’s law,
\[
N_{i,\text{cath}} = -\sum_{jkc}^L \frac{s_{ij}}{n_j} i_{jkc} A_{jkc}
\]
and Fick’s law,
\[
N_{i,jkc}(x) = -\frac{D_i}{B} \int_{0}^{x} \frac{dc_{i,\text{cath},jkc}}{dz} \bigg|_{\xi'} \frac{\partial}{\partial \xi}(\xi' = -1) \, dz'
\]
\[
+ \frac{D_i}{B} \int_{0}^{x} \frac{dc_{i,\text{an},jkc}}{dz} \bigg|_{\xi'} \frac{\partial}{\partial \xi}(\xi' = 1) \, dz'.
\]

Equation D-34 is solved by the method of Acrivos and Chambre \(^59\) to give \(c_{i,\text{an},jkc}\) and \(c_{i,\text{cath},jkc}\) evaluated at each of the collocation points along the electrodes. \(c_{i,\text{an},jkc}\) can be regarded as the concentration of species \(i\) on the anode resulting from current \(i_{jkc}\) on the cathode and zero current on the anode. Note that the linearity of the equations implies
\[
c_{i,\text{an}}(x_p) - c_{i,\text{feed}} = \sum_{jka} c_{i,\text{an},jka}(x_p) A_{jka} + \sum_{jkc} c_{i,\text{an},jkc}(x_p) A_{jkc}.
\]

This equation may be written in matrix form as
\[
c = PA,
\]
where \(c\) is the vector of concentrations of each species on each electrode at each
collocation point, \(x_p\), \(A\) is a vector containing the \(A_{jka}\) and \(A_{jkc}\) coefficients, and \(P\) is a \((2)(\text{nspec})(np) \times (2)(\text{nrxn})(np)\) matrix. This matrix form will be useful shortly for discussing the method for treating limiting-current problems.

It is useful to know the shapes of the concentration and potential functions because they must be superposed to produce reasonable concentration and potential profiles to be substituted into the Butler-Volmer kinetic expression. Clearly, if the potential is poorly represented, the current calculated from it will also be inaccurate, particularly for Tafel kinetics, where the current depends exponentially on the potential.

Figures D-7 through D-11 show the potential and concentration functions corresponding to functions for uniform current, a ramp current (shifted Legendre polynomial \(P_1(x)\)), and limiting current. Note that the uniform current produces a nonuniform potential (figure D-7). When this nonuniform potential distribution is substituted into the Butler-Volmer equation, a nonuniform current distribution results. This example demonstrates the need for choosing a set of current functions that can strike a balance between competing factors, such as ohmic potential drop and reaction kinetics.

With the potentials and concentrations computed from the potential and concentration functions, the Butler-Volmer partial currents are calculated, and the coefficients are found by a Newton-Raphson method that zeros the errors between the Butler-Volmer currents and the approximate currents at the collocation points, \(x_p\), where

\[
\text{error}_\text{an} = i_{j,B-V,an}(x_p) - \sum_{jka} i_{jka}(x_p) A_{jka}
\]  

(D-32a)

and

\[
\text{error}_\text{cath} = i_{j,B-V,cath}(x_p) - \sum_{jkc} i_{jkc}(x_p) A_{jkc}
\]  

(D-32b)

Also, the integration constant \(\Phi^*\) from equation D-20 must be determined to satisfy the condition of equal currents on the two electrodes. The vector of coefficients \(A\) is determined by solving for
Figure D-7. Potential functions resulting from a uniform current function on the anode ($h/L = 0.1, Pe \ h/L = 100$).
Figure D-8. Concentration functions resulting from a uniform current function on the anode. \((h/L = 0.1, Pe h/L = 100)\).
Figure D-9. Potential functions resulting from a ramp current function on the anode. ($h/L = 0.1, Pe h/L = 100$).
Figure D-10. Concentration functions resulting from a ramp current function on the anode. \((h/L = 0.1, Pe h/L = 100)\).
Figure D-11. Potential functions resulting from a limiting current function on the anode. \((h/L = 0.1, Pe h/L = 100)\).
\[ \Delta A = A^{new} - A^{old} = -[\text{Derror}]^{-1} \text{(error)} \vert_{A^{old}}, \]

where \([\text{Derror}]\) is the Jacobian containing \( \frac{\partial(i_{i,B-V}(z_p) - i_{i,\text{approx}}(z_p))}{\partial A_{jk}} \). The elements of the Jacobian are calculated by the chain rule

\[
\frac{\partial i_{B-V, an}}{\partial A_{jkc}} = \sum_i \left( \frac{\partial c_i}{\partial c_{i,an}} \frac{\partial c_{i,an}}{\partial A_{jkc}} + \frac{\partial i_{B-V, an}}{\partial (\Phi_{an}^* - \Phi^*)} \right),
\]

and the partial derivatives of concentration and potential with respect to the coefficients are simply the concentration and potential functions; for example,

\[
\frac{\partial c_{i,an}}{\partial A_{jkc}} = c_{i,an,jkc},
\]

from equation D-30. Note that the matrix equation D-33 can be augmented with an extra row and column containing the equation for equal currents and the unknown \( \Phi^* \), respectively.

**D.5.4. Limiting-Current Problems**

The method described above is most successful below the limiting current. A disadvantage of the method is that it is possible, during the iterations, to get negative concentrations, even at the collocation points, \( z_p \). Negative concentrations are not only unrealistic, but also they cannot be used in the Butler-Volmer equation to calculate new currents. To avoid this problem we treat limiting-current problems separately and set the surface concentrations to correspond to limiting current.

We shall first outline how to treat limiting-current problems with a single species. On the limiting electrode, for example, the cathode, we replace the equations D-32b with an equation for the error in concentration

\[
c_{\text{cath}}(z_p) = 0 = c_{\text{feed}} + \sum_{jka} c_{an,jka}(z_p)A_{jka} + \sum_{jkc} c_{an,jkc}(z_p)A_{jkc},
\]

(i.e. some of the rows of the matrix equation (D-32) are replaced).

It is also useful to use different trial functions on the two electrodes. In particular,
the limiting-current function is used on the cathode only and the modified limiting-current function (equation D-15) is used on the anode only. This is done by replacing the equations for the errors at the last collocation points with equations that set the appropriate coefficients to zero.

Another manipulation for the limiting-current calculations is to provide an initial guess for \( \Phi^* \) and the coefficients. The guessed coefficients are provided by equation D-31, with \( c_{\text{cath}} = 0 \) and \( c_{\text{an}} = 2 \), and equations that set the appropriate limiting-current-function coefficients to zero.

For problems with more than one species, it is cautioned that not all of the concentrations may be specified. The coefficients will still be obtained from equation D-36, but since the full \( P \) matrix may be singular, not all of the concentrations may be constrained.

The number of concentrations that can be constrained depends on the rank of the \((s_{ij}/n_j)\) matrix because Faraday's law must be inverted to extract the current coefficients from the fluxes. For example, the matrix for the Fe(CN)\(_6\)^3-/Fe(CN)\(_6\)^4- system with the hydrogen-evolution side reaction is

\[
\frac{s_{ij}}{n_j} = \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix},
\]

where the rows are for ferricyanide and ferrocyanide, respectively, and the columns are for the main and side reactions. Clearly, it is impossible to calculate the coefficients for the side reaction currents because the rank of \((s_{ij}/n_j)\) is one. Note that, although \( n_{\text{spec}} = n_{\text{rxn}} \), and \( P \) is a square matrix, \( P \) cannot be inverted to give the coefficients \( A \) from the concentrations \( c \) because the side-reaction current coefficients are multiplied by zeros. Therefore, only the \( A_{mk} \) coefficients can be calculated from the equation

\[
c_R = MA_m,
\]

where \( m \) is the index for the main reaction, \( R \) is the index for the limiting species on the
given electrode, \( elec \), and \( M \) is the nonsingular \((2)(\text{rank})(np)\times(2)(\text{rank})(np)\) matrix containing \( c_{R,elec,\text{mk}}(x_p) \) (see equation D-32). In the ferricyanide example, \( R \) might be ferricyanide on the cathode and ferrocyanide on the anode; therefore by constraining \( c_{\text{ferri},\text{cath}} \) and \( c_{\text{ferro},\text{anode}} \), one can calculate \( A_{1k,an} \) and \( A_{1k,\text{cath}} \) from equation D-38 by inverting \( M \). The method then proceeds with the new \( A_1 \) and the old \( A_2 \) vectors.

For problems that do not converge readily, the continuation method (section D.5.5) is sometimes helpful. The best results are usually obtained by starting with a small cell voltage and applying continuation to reach the desired cell voltage.

**D.5.5. First-Order Continuation**

It is useful in any iterative problem to use a method that provides a good initial guess. One such method is first-order continuation. The idea behind the method is that if the problem can be made to converge by changing one parameter, then the iterations can start from the "easy" problem and proceed to the difficult problem by gradually changing the parameter. The initial guess for each new problem is provided by a first-order extrapolation from the converged solution of the previous problem. For example, to solve \( f(x) = 0 \), one would write

\[
f(x,p) = f(x_o,p_o) + \left. \frac{\partial f}{\partial x} \right|_{x_o,p_o} \Delta x + \left. \frac{\partial f}{\partial p} \right|_{x_o,p_o} \Delta p,
\]

where \( p_o \) is the value of the parameter \( p \) that gives convergence. Since

\[
f(x_o,p_o) = 0,
\]

therefore, for a given \( \Delta p \), one can guess a new \( x \):

\[
x_{\text{new}} = x_o + \Delta x.
\]

Note that \( \frac{\partial f}{\partial x} \) is the Jacobian for solving \( f(x) = 0 \) by the Newton-Raphson method;
therefore some time can be saved by using "LU decomposition" rather than matrix inversion. As expected, \( \Delta p \) must be chosen carefully; it should be large enough to make progress toward the desired \( p \), but small enough that the linear extrapolation provides a good enough initial guess to achieve convergence in each new problem. An automatic parameter step-size adjustment procedure is to count the number of Newton iterations required for each problem and adjust \( \Delta p \) according to the following scheme:

<table>
<thead>
<tr>
<th>Newton Iterations</th>
<th>( \Delta p_{\text{new}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; 3</td>
<td>( 3 \Delta p_{\text{old}} )</td>
</tr>
<tr>
<td>= 3</td>
<td>( \Delta p_{\text{old}} )</td>
</tr>
<tr>
<td>&gt; 3</td>
<td>( 0.56 \Delta p_{\text{old}} )</td>
</tr>
</tbody>
</table>

This procedure is an attractive candidate for the channel problem because convergence can be achieved under certain conditions, such as small cell voltages or small exchange current densities for side reactions. A disadvantage of the method is that convergence can be too slow.

D.6. Results at the Limiting Current

Figures D-12 through D-15 show results from the limiting-current algorithm (see program NEWCHAN, appendix G) for large and small aspect ratios \( h/L = 1 \) and 0.1. Nine collocation points were used on each electrode, and about 20 iterations were required to achieve convergence without a "restart." The computer time for these runs was about 4 minutes on a VAX 8600, roughly 20 times faster than for the old program (program CHANNEL).

Figures D-12 and D-13 show the anode current density as calculated from the trial functions and as calculated from the nonlinear Butler-Volmer expression. The circles on
Figure D-12. Anode current distributions calculated from trial functions and Butler-Volmer expression. \((h/L = 1, Peh/L = 100)\).
Figure D-13. Anode current distributions calculated from trial functions and Butler-Volmer expression. \((h/L = 0.1, Peh/L = 100)\).
the figures are the collocation points at which agreement was required. The close agreement between the currents at points other than these nine points shows that an appropriate set of trial functions was used. With a poor set of functions, the two curves would have to oscillate to agree at the nine points. Note that the choice of the parameters $\Delta$ for the special trial functions is important. To obtain figures D-12 and D-13, we guessed a set of $\Delta$'s and estimated $i_{\text{max}}/i_{\text{min}}$ from the converged (but oscillatory) solution. The $\Delta$'s were then refined so that each trial function had the correct $i_{\text{max}}/i_{\text{min}}$. One additional refinement was sufficient to give the fit shown in figure D-12.

Figures D-14 and D-15 show the potential distributions. Recall that the potential becomes uniform across the channel infinitely far upstream and downstream, provided the total currents are equal on the two electrode. If the total currents were unequal on the electrodes, the potentials would continue to change, corresponding to current flowing to $+\infty$ and $-\infty$. The distance over which the potentials decay to the constant value depends on the amount of fringing of the current lines (see figure 3-1). The fringing generally occurs over a distance of order $h$, as shown in the figures (see also figure 5-19).

Figure D-14 shows the potential distribution for large $h/L$ ($h/L = 1$). Note that the potential adjacent to the anode is nearly flat, corresponding to the nearly-primary current distribution. Although the cathode is at limiting current, the anode potential distribution is unaffected because $h/L$ is large enough to suppress interaction between the two electrodes. The cathode potential distribution is highly nonuniform.

Figure D-15 shows the potential distribution for $h/L = 0.1$. For this smaller $h/L$, there is an interaction through Laplace's equation, and, therefore, the current distributions on the anode and cathode are nearly identical, except at the leading edge. As in section 5.4, the dipole of $(i_a + i_c)$ leads to a significant difference between the upstream and downstream potentials. Note that although the axial current is smaller for $h/L = 0.1$ than for $h/L = 1$, the difference between the upstream and downstream
Figure D-14. Potential distributions for $h/L = 1$, $Peh/L = 100$. 
Figure D-15. Potential distributions for $h/L = 0.1$, $Peh/L = 100$.)
potentials is larger. This example illustrates the importance of accounting for axial current and potential variations, even for small $h/L$.

D.7. Summary and Suggestions for Future Work

We have investigated iterative techniques with the objective of devising an efficient and accurate method for solving the thin-gap-flow-cell problem. The collocation method as revised by Miksis$^{107,108}$ has proven to be faster than the successive substitution method and more accurate than the simple collocation method used by Cabán and Chapman.$^{22}$ The disadvantages are that the current functions must be selected carefully.

Program NEWCHAN (appendix G), in its present form, is not guaranteed to be robust. Suggestions for improving the program are to automate the selection of the parameters contained in the special functions and the automate the choice between the basic algorithm and the limiting-current algorithm.

To automate the selection of the $\Delta$ parameters from equations D-15 through D-17, one should have an $a \text{ priori}$ estimate of the total current from either the limiting, primary, or uniform current distribution. Next, one would estimate the ratio of the current at the edge to the minimum current for linear or Tafel kinetics.$^{111}$

Linear:  
\[ i_{\text{max}} = \sqrt{\frac{J}{2}} \frac{\tanh \epsilon}{\phi_o} i_{\text{min}} \]  

Tafel:  
\[ i_{\text{max}} = \frac{\delta \phi_o}{K(\tanh^2 \epsilon)} i_{\text{min}} \]  

where $\delta$ in equation D-43b is the dimensionless average current. The parameters $\Delta$ would then be chosen to match the ratio $i_{\text{max}}/i_{\text{min}}$.

The choice between the basic algorithm and the limiting-current algorithm might be governed by the surface concentration; if it falls below some threshold, the program would switch to the limiting-current method. The goal would be to produce a program.
that could generate by first-order continuation an entire polarization curve with current, concentration, and potential distributions at each applied potential.
Appendix E. Programs for Investigating Iterative Methods for the Rotating-Disk Problem

E.1. Program CURDB

Description:

Input Variables

LMAX = number of radial mesh points
NMAX = number of B coefficients
IH = half the number of Gaussian quadrature points
IHH : not used
IM = total number of Gaussian quadrature points
X(I) = Gaussian quadrature roots (in the interval 0 to 1)
W(I) = Gaussian quadrature weights
C(1) = \( c/c_\infty \) at center of disk (input C(1) = 0 to stop running cases)
AN = dimensionless limiting current = \( N \)
TPLUS = transference number
ALPHA = anodic transfer coefficient
BETA = cathodic transfer coefficient
GAMMA = exponent for concentration dependence of exchange current density
EXCH = dimensionless exchange current density = J
DAMP = damping factor (not used)

Output Variables

ERR = 1 if too many iterations, 0 otherwise
ANS(N) = converged B(n)
ITFIN = number of iterations required for convergence
ERR(N,IT) = error in B(n) at iteration IT = $e_{IT}$
ORD(N,IT) = $\ln |\ln \frac{e_{IT+1}}{e_{IT}}|$
SLOPE = slope of ORD vs. IT
V = disk potential
AVG = $\delta/N = i_{avg}/i_{lim}$
R(J) = radial position
C(J) = surface concentration $c/c_\infty$
CUR(J) = dimensionless current density, $i/<i>$
E(J) = total overpotential = $V - \Phi_o$
B(I) = B coefficients
TAF = 0 if Tafel approximation is used, 1 if not
RAT2 = $(V - E(1))/B(1)$, proportional to effective resistance
RAT1 = CUR(1)/(AVG*N*\theta'(0)), ratio of current at center to average current
JCOUNT = iterations required
PROGRAM CURDB(INPUT,OUTPUT)
C CURRENT DISTRIBUTION ON A ROTATING DISK WITH AN INTEGRAL EQUATION
C FOR THE DIFFUSION LAYER
C
C This is program curd rewritten for iteration on B coefficients
C by a Newton-Raphson method
C
DIMENSION PM(21), B(21), C(201), CUR(201), E(201), PP(21,201), R(201), AA
1(200), BB(200), X(40), W(40), CUG(40), PG(21,40), de(201), bd(21,21),
1d(21,43), bin(21), xarr(51,51), err(51,51), ord(51,51), ans(51)
C
COMMON/a/ E, CUR, C, LMAX, TPLUS, AN, EXCH, ALPHA, BETA, GAMMA, TAF, AA, BB, C1
1, C2, EX
common/b/nmax, pp, im, x, dz, r, pg, w, pm

101 FORMAT (6H ERROR, I4)
102 FORMAT (3H N=, F10.4, 10H , TPLUS=, F8.4, 6H , V=, F10.5, 8H , AVG=,
1F10.6/41HO R C CUR ETA/(4F11.5))
103 FORMAT (3I4)
104 FORMAT (9E8.4)
105 FORMAT (7H1ALPHA=, F8.4, 9H , BETA=, F8.4, 10H , GAMMA=, F8.4, 9H , E
1XCH=, F8.4)
107 FORMAT (2H0B, F9.5, 5F11.5/(6F11.5))
108 FORMAT (6E12.9)
109 format (* calculated derivatives*)
132 format(1x, 3e22.15)

print 109
READ 103, LMAX, NMAX
nmax = 2*nmax+1
EX = 2.0/3.0

DO 29 L=1, LMAX
A = L
AA(L) = 2.0*A**EX - (A+1.0)**EX - (A-1.0)**EX
29 BB(L) = A**EX - (A-1.0)**EX

DZ = 1.0/(LMAX-1)

DO 1 L=1, LMAX
Z = (L-1)*DZ
R(L) = Z**(1.0/3.0)
ETA = SQRT(1.0-R(L)*R(L))
DO 1 N=1, NMAX
1 PP(N, L) = P(2*N-2, ETA)

DO 2 N=1, NMAX
2 PM(N) = - 0.6366198/P(2*N-2, 0.0)**2

EX = 1.0/3.0
C2 = 1.11984652
READ 103, IH, IHH, IM
IHP1 = IH + 1
READ 108, (X(I), I=IHP1, IM)
READ 108, (W(I), I=IHP1, IM)

DO 34 I=1, IM
   IF (I-IH) 31, 31, 32
31 IR = IM - I + 1
   X(I) = 0.5 - 0.5*X(IR)
   W(I) = W(IR)
   GO TO 33
32 X(I) = 0.5 + 0.5*X(I)
33 XX = SQRT(1.0 - X(I)**2)
   DO 34 N=1, NMAX
34 PG(N,I) = P(2*N-2, XX)

3 READ 104, C(1), AN, TPLUS, ALPHA, BETA, GAMMA, EXCH, DAMP
   CI = C2*(1.0 - C(1))
   IF (C(1)) 4, 4, 5
4 STOP
5 JCOUNT = 0
   TAF = 1.0

Test for Tafel approximation

IF (EXCH-400) 7, 7, 6
6 TAF = 0.0
   EXCH = 1.0
7 ETAC = ALOG(C(1)) + TPLUS*(1.0 - C(1))
   CUR(1) = - (1.0-C(1))*AN*EXCH*1.11984652/C(1)**GAMMA
   ETAS = - ALOG(TAF-CUR(1))/BETA
8 DO 9 J=1, 1,100
   F = TAF*EXP(ALPHA*ETAS) - EXP(-BETA*ETAS)
   FP = TAF*ALPHA*EXP(ALPHA*ETAS) + BETA*EXP(-BETA*ETAS)
   IF (ABS(CUR(1)-F) - 0.0000001*ABS(CUR(1))) 10, 10, 9
9 ETAS = ETAS + (CUR(1)-F)/FP

10 CUR(1) = 1.11984652*AN*(1.0-C(1))
   DO 11 L=1, LMAX
      C(L) = C(1)
11 E(L) = ETAC + ETAS
   B(1) = 0.0
   PRINT 105, ALPHA, BETA, GAMMA, EXCH
   BOLD = B(1)
JCOUNT = JCOUNT + 1
CALL THETA
DO 16 I=1,IM
   LI = X(I)**3/DZ + 1
16   CUG(I) = CUR(LI) + (CUR(LI+1) - CUR(LI)) * (X(I)**2 - R(LI)**2) / (R(LI+1)**2 - R(LI)**2)
   (R(LI+1)**2 - R(LI)**2)

DO 15 N=1,NMAX
   B(N) = 0.0
   DO 14 I=1,IM
   14   B(N) = B(N) + CUG(I) * X(I) * PG(N,I) * W(I)
15   B(N) = 0.5 * B(N) * (4*N-3) / PM(N)

12   b0ld = b(1)
   DO 310 N=1,NMAX
      xarr(n,jcount) = b(n)
   c   PRINT 160,jcount,b(n),b0ld,xarr(n,jcount)
310  CONTINUE

   jcount = jcount+1

   DO 90 I=1,NMAX
      bin(I) = b(I)
   c   calculate Jacobian
   c   DO 100 I=1,NMAX
100   call deriv(i,bd,b,v)
   c   DO 90 I=1,NMAX
      bin(I) = b(I)
   c   call theta
   DO 216 I=1,IM
   c   LI = X(I)**3/DZ + 1
216   CUG(I) = CUR(LI) + (CUR(LI+1) - CUR(LI)) * (X(I)**2 - R(LI)**2) / (R(LI+1)**2 - R(LI)**2)
   c   (R(LI+1)**2 - R(LI)**2)
   DO 215 N=1,NMAX
   c   B(N) = 0.0
   DO 214 I=1,IM
   c   B(N) = B(N) + CUG(I) * X(I) * PG(N,I) * W(I)
215   B(N) = 0.5 * B(N) * (4*N-3) / PM(N)

   PRINT 145
145   FORMAT(* hand-calc new bs, b2,b1*)
      b2 = (b(2) - bd(2,2) * bin(2)) / (1. - bd(2,2))
      b1 = b(1) + bd(1,2) * (b2 - bin(2))
   c   PRINT 132,b2,b1
5   DO 110 I=1,NMAX
   DO 110 J=1,NMAX
110   d(i,j) = 0.
do 111 i=1,nmax
   do 111 j=1,nmax
      xi = 0.
      if(i.eq.j) xi = 1.0
      bd(i,j) = xi - bd(i,j)
   c
   do 112 i=1,nmax
      d(i,1) = b(i) - bin(i)
   c
   call matinv(nmax,mmax,determ,bd,d)
   if(determ.eq.0.) print 116
      format(27h zero determinant in matinv)
   c
   update
   do 117 i=1,nmax
      b(i) = bin(i) + d(i,1)
   c
   do 320 n=1,nmax
      xarr(n,jcount) = b(n)
      if(jcount.gt.l) print 321,n,xarr(n,jcount)-xarr(n,jcount-l)
      format(* n=*,i2,* deltax=*,lpell.4)
   320 continue
   JERR = 1
   IF (JCOUNT-50) 19,19,20
      cr.t = (ABS(B(1)-BOLD) - 0.000001*ABS(B(1)))
      do 330 n=1,nmax
         print 160,jcount,b(n),bold,xarr(n,jcount)
      160 format(* jcount=*,i2,* b(n)=*,lpel1.4,* bold=*,1pel1.4,* xarr=*,
                     1 1pel1.4)
      330 continue
      IF (ABS(B(1)-BOLD) - 1.e-10*ABS(B(1))) 21,21,12
   20 PRINT 101, JERR
      AVG = - B(1)/0.7853982/AN/1.11984652
   c
   Test errors for quadratic convergence
   c
   do 340 n=1,nmax
      ans(n) = b(n)
      continue
   c
   itfin = jcount
   do 350 n=1,nmax
      print 70,n,ans(n),itfin
      format(* n=*,i2,* answer=*,1pe17.10,* itfin=*,i2)
   70 continue
   c
   print 59
      format(* it err(n,it)*)
   c
do 30 n=1,nmax
  print 61,n
61  format(* n=*,i2)
do 30 it=1,itfin
  err(n,it) = xarr(n,it) - ans(n)
  print 60,it,err(n,it)
60  format(1x,i3,5x,1pe15.8)
30  continue

print 91
91  format(* it ord(it)*)

  itfml = itfin - 1
  do 35 n=1,nmax
    print 61,n
  do 35 it=1,itfml
    ord(n,it) = 0.
    if(err(n,it).eq.0. .or. err(n,it+1).eq.0.) go to 36
    ord(n,it) = alog(abs(a1og(abs(err(n,it+1)/err(n,it)))))
  36  print 60,it,ord(n,it)
  35  continue

  itfm2 = itfin - 2
  do 37 n=1,nmax
    print 61,n
  do 37 it=1,ifm2
    ' slope = ord(n,it+1) - ord(n,it)
    if(ord(n,it+1).eq.0.) go to 37
    print 80,it,slope,exp(slope)
  80  format(1x,i3,5x,1pe11.4,5x,e11.4)
  37  continue

DO 22 L-1,LMAX
22  CUR(L) = CUR(L)/AN/1.11984652
  RAT1 = CUR(1)/AVG
  RAT2 = (V-E(1))/B(1)
  PRINT 102, AN,TPLUS,V,AVG,(R(J),C(J),CUR(J),E(J),J=1,LMAX)
  PRINT 107, (B(I),I=1,NMAX),TAF,RAT2,RAT1
  PRINT 103, JCOUNT
  GO TO 3
END

SUBROUTINE THETA
SUBPROGRAM FOR CALCULATING CONCENTRATION by equating Fick's-law
and Faraday's-law fluxes
(X1-X2)/PN + (C1+X3))*EXCH = 0

DIMENSION E(201),CUR(201),A(200),B(200),TH(201)
COMMON/a/ E,CUR,TH,NZT1,T,PN,EXCH,AL,BE,GAM,TAF,A,B,C1,C2,EX
C

101 FORMAT (17HONOT CONVERGED AT,I4)

NZT = NZT1-1
DEVM = 0.0001
DZ = 1.0/NZT
S = TH(1)
DO 60 NZ = 2,NZT1
Z = (NZ - 1)*DZ
SUM = 0.0
IF (NZ .LE. 2) GO TO 42
C CALC. SUM(TH(J)*A(K))
DO 40 J=3,NZ
K = NZ - J + 1
40 SUM = SUM + TH(J-1)*A(K)
42 ETA = E(NZ)
NJ = NZ - 1
C

DO 56 N=1,20
X1 = TAP*S**((GAM - AL)*EXP(AL*ETA)*EXP(AL*T*(S - 1.0))
DX1 = X1*(((GAM - AL)/S + AL*T)
X2 = S**((GAM + BE)*EXP(-BE*ETA)*EXP(BE*T*(1.0 - S))
DX2 = X2*(((GAM + BE)/S - BE*T)
C3 = 1.50*C2*Z**EX/DZ**(1./3.)
X3 = C3*(TH(1)*B(NJ) + SUM - S)
DTH = S - ((X1-X2)/PN + (C1+X3)*EXCH)/((DX1-DX2)/PN - C3*EXCH)
CUR(NZ) = PN*(C1+X3)
IF (ABS(S-DTH) - DEVM*ABS(DTH)) 60,60,56
56 S = DTH
C

PRINT 101, NZ
60 TH(NZ) = DTH
RETURN
END
C

FUNCTION P(N,X)
C CALCULATION OF LEGENDRE POLYNOMIALS
P1= 1.0
P2= X
IF (N-1) 1,2,3
1 P= P1
RETURN
2 P= P2
RETURN
3 NM1= N - 1
DO 4 NU=1,NM1
P= (X*FLOAT (2*NU+1)*P2-FLOAT (NU)*P1)/FLOAT (NU+1)
P1= P2
4 P2= P
RETURN
END
subroutine deriv(i,bd,b,v)

This subroutine calculates derivatives with respect to the B coefficients

DIMENSION PM(21), B(21), C(201), CUR(201), E(201), PP(21, 201), R(201), AA
1(200), BB(200), X(40), W(40), CUG(40), PG(21, 40), de(201), bd(21, 21),
1dcur(201), thd(201), btemp(21), dcug(40)

COMMON/a/ E, CUR, C, LMAX, TPLUS, AN, EXCH, ALPHA, BETA, GAMMA, TAF, AA, BB, C1
1, C2, EX
COMMON/b/ nmax, pp, im, x, dz, r, pg, w, pm

v = e(1)
DO 1 j=1, NMAX
v = v + b(j)*pp(j,1)
1 Btemp(j) = 0.0

btemp(i) = 1.0
de(1) = 0.0
dcur(1) = 0.0

DO 3 L=2, LMAX
phi = v
dPHI = 0.0
DO 2 N=1, NMAX
phi = phi - b(n)*pp(n,1)
2 dPHI = dPHI - Btemp(N)*(PP(N,L) - pp(n,1))
e(1) = phi
3 dE(L) = dphi

71 format(1x,3e22.15)
CALL THETAp(de, i, dcur, thd)

DO 4 Ii=1, IM
LI = X(Ii)**3/DZ + 1
CUG(Ii) = CUR(LI) + (CUR(LI+1)-CUR(LI))*X(Ii)**2-R(LI)**2)/
1 (R(LI+1)**2 - R(LI)**2)
4 dCUG(Ii) = dCUR(LI)+(dCUR(LI+1)-dCUR(LI))*X(Ii)**2-R(LI)**2)/
1(R(LI+1)**2 - R(LI)**2)

V = E(1)
DO 6 N=1, NMAX
if(i.eq.nmax) B(N) = 0.0
Bd(N,i) = 0.0
DO 5 Ii=1, IM
if(i.eq.nmax) B(N) = B(N) + CUG(Ii)*X(Ii)*PG(N,Ii)*W(Ii)
5 Bd(N,i) = Bd(N,i) + dCUG(Ii)*X(Ii)*PG(N,Ii)*W(Ii)
Bd(N,i) = 0.5*Bd(N,i)*(4*N-3)/PM(N)
if(i.eq.nmax) B(N) = 0.5*B(N)*(4*N-3)/PM(N)
6 V = V + Bd(N,i)*PP(N,1)
c
print 76,i
76 format(* bd, column*,i2)
print 71,(bd(n,i),n=1,nmax)
return
derend

c
SUBROUTINE THETAp(de,je,dcur,thd)
C SUBPROGRAM FOR CALCULATING CONCENTRATION derivatives with
c respect to the B coefficients
C
DIMENSION E(201),CUR(201),A(200),B(200),TH(201),thd(201),de(201),
idcur(201)
C
COMMON/a/ E,CUR,TH,NZT1,T,PN,EXCH,AL,BE,GAM,TAF,A,B,C1,C2,EX
C
101 FORMAT (17HONOT CONVERGED AT,I4)
C
NZT = NZT1-1
DEVM = 0.0001
DZ = 1.0/NZT
S = TH(1)

  ds = 0.
    thd(1) = 0.
  DO 60 NZ = 2,NZT1
Z = (NZ - 1)*DZ
SUM = 0.0
  dsum = 0.0
  IF (NZ .LE. 2) GO TO 42
C
  CALC. SUM(TH(J)*A(K))
C
  DO 40 J-3,NZ
K = NZ - J + 1
    dsum = dsum + thd(j-1)*a(k)
  40 SUM = SUM + TH(J-1)*A(K)
C
42 ETA = E(NZ)
  deta = de(nz)
NJ = NZ - 1
C
  DO 56 N=1,20
X1 = TAF*S**(GAM - AL)*EXP(AL*ETA)*EXP(AL*T*(S - 1.0))
DX1 = X1*((GAM - AL)/S + AL*T)
  xld = a1*xld*deta + dxl*ds
  dx1d = xld*(((gam-al)/s + al*t) - x1*(gam-al)/s**2*ds
X2 = S**(GAM + BE)*EXP(-BE*ETA)*EXP(BE*T*(1.0 - S))
DX2 = X2*((GAM + BE)/S - BE*T)
  x2d = -be*x2*deta + dx2*ds
dx2d = x2d*((gam+be)/s - be*t) - x2*(gam+be)/s**2*ds
C3 = 1.50*C2*Z**EX/DZ**(l./3.)
X3 = C3*(TH( 1)*B(NJ) + SUM - S)
dx3 = c3*(dsum-ds)
DTH= S - ((X1-X2)/PN + (C1+X3)*EXCH)/(DX1-DX2)/PN - C3*EXCH)
dthd= ds - (((xld-x2d)/pn + dx3*exch)*)((dx1-dx2)/pn - c3*exch)
1 - c3*exch)**2
dCUR(NZ) = PN*dX3
CUR(NZ) = PN*(C1+X3)
IF (ABS(S-DTH) - DEVM*ABS(DTH)) 61,61,57
57 S = DTH
56 ds = dthd

PRINT 101, NZ
61 thd(nz) = dthd
60 TH(NZ) = DTH
RETURN
END

SUBROUTINE MATINV(N,M,DETERM,b,d)

This subroutine solves a matrix equation of the form BX=D.
The answer X is put into the first column of D. Note that
D should be dimensioned nxm, where m=2n+1 and that all
columns of D (except the first) should be initialized with
zeros

Input variables

n = dimension of square b matrix
m = number of columns in d matrix
b = matrix in equation bx=d
d : first column contains vector d in bx=d

Output variables

d : first column contains vector x in bx=d
dterm = 0. if matrix has zero determinant

DIMENSION ID(21),b(21,21),d(21,43)

DETERM=1.0
DO 1 I=1,N
1   ID(I)=0
DO 18 NN=1,N
BMAX=1.1
DO 6 I=1,N
IF(ID(I).NE.0) GOTO 6
BNEXT=0.0
BTRY=0.0
DO 5 J=1,N
IF(ID(J).NE.0) GOTO 5

END
IF(ABS(B(I,J)).LE.BNEXT) GOTO 5
BNEXT=ABS(B(I,J))
IF(BNEXT.LE.BTRY) GOTO 5
BNEXT=BTRY
BTRY=ABS(B(I,J))
JC=J
5 CONTINUE
IF(BNEXT.GE.BMAX*BTRY) GOTO 6
BMAX=BNEXT/BTRY
IROW=I
JCOL=JC
6 CONTINUE
IF(ID(JC).EQ.0) GOTO 8
DETERM=0.0
RETURN
8 ID(JCOL)=1
IF(JCOL.EQ.IROW) GOTO 12
DO 10 J=1,N
SAVE=B(IROW,J)
B(IROW,J)=B(JCOL,J)
10 B(JCOL,J)=SAVE
DO 11 K=1,M
SAVE=D(IROW,K)
D(IROW,K)=D(JCOL,K)
11 D(JCOL,K)=SAVE
12 F=1.0/B(JCOL,JCOL)
DO 13 J=1,N
13 B(JCOL,J)=B(JCOL,J)*F
DO 14 K=1,M
14 D(JCOL,K)=D(JCOL,K)*F
DO 18 I=1,N
IF(I.EQ.JCOL) GO TO 18
F=B(I,JCOL)
DO 16 J=1,N
16 B(I,J)=B(I,J)-F*B(JCOL,J)
DO 17 K=1,M
17 D(I,K)=D(I,K)-F*D(JCOL,K)
18 CONTINUE
RETURN
END
~789
41 3
20 10 40
3877241751-21160840707-11926975807-12681521850-13419940908-14137792044-1
4830758017-15494671251-16125538897-16719566846-17273182552-17783056514-1
8246122308-186599595032-19020988070-19328128083-19579168192-19772599500-1
9907262387-19982377097-1
7750594798-27703981816-27611036190-27472316906-27288658240-27061164739-2
1791204582-26480401346-26130624249-25743976910-25322784698-24869580764-2
4387090819-23878216797-23346019528-22793706098-22224584919-21642105838-2
1049828453-24521277099-3
70000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 30000-1
10000-0  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  10000-0
50000-2  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  10000-0
20000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  80000-1
30000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  60000-1
40000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  40000-1
50000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  30000-1
60000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  30000-1
70000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  30000-1
80000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  30000-1
90000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  30000-1
95000-1  78800+1  36300-1  15000-0  50000-1  42000-1  89600-1  80000-1
80000-1  20000+1  50000-1  50000-1  50000-1  50000-1  50000+2  70000-1
80000-1  50000+1  50000-1  50000-1  50000-1  50000-1  50000+2  50000-1
20000-1  50000-0  50000-1  50000-1  50000-1  50000-1  50000+2  80000-1
50000-1  10000+2  50000-1  50000-1  50000-1  50000-1  50000+2  30000-1
50000-1  10000+1  50000-1  50000-1  50000-1  50000-1  50000+2  70000-1
50000-1  20000+1  50000-1  50000-1  50000-1  50000-1  50000+2  60000-1
E.2. Program CURDBN

Description:


Input Variables

LMAX = number of radial mesh points
NMAX = number of B coefficients
IH = half the number of Gaussian quadrature points
IHH : not used
IM = total number of Gaussian quadrature points
X(I) = Gaussian quadrature roots (in the interval 0 to 1)
W(I) = Gaussian quadrature weights
C(1) = \( c/c_\infty \) at center of disk (input C(1) = 0 to stop running cases)
AN = dimensionless limiting current = \( N \)
TPLUS = transference number
ALPHA = anodic transfer coefficient
BETA = cathodic transfer coefficient
GAMMA = exponent for concentration dependence of exchange current density
EXCH = dimensionless exchange current density = $J$

DAMP = damping factor (not used)

**Output Variables**

ERR = 1 if too many iterations, 0 otherwise

ANS(N) = converged $B(n)$

ITFIN = number of iterations required for convergence

ERR(N,IT) = error in $B(n)$ at iteration IT = $e_{IT}$

ORD(N,IT) = $\ln \left(\ln \left|\frac{e_{IT+1}}{e_{IT}}\right|\right)$

SLOPE = slope of ORD vs. IT

$V = \phi_{sk}$ potential

AVG = $\delta/N = \frac{i_{av}}{i_{im}}$

R(J) = radial position

C(J) = surface concentration $c/c_\infty$

CUR(J) = dimensionless current density, $i/<i>$

$E(J) = \text{total overpotential} = V - \Phi_a$

$B(I) = B$ coefficients

TAF = 0 if Tafel approximation is used, 1 if not

RAT2 = $(V - E(1))/B(1)$, proportional to effective resistance

RAT1 = $\text{CUR}(1)/(\text{AVG}^*N^*\theta_\epsilon(0))$, ratio of current at center to average current

JCOUNT = iterations required
PROGRAM CURDBN(INPUT,OUTPUT)
C CURRENT DISTRIBUTION ON A ROTATING DISK WITH AN INTEGRAL EQUATION
C FOR THE DIFFUSION LAYER
C
This is program curd rewritten for iteration on B coefficients
by a Newton-Raphson method with numerical derivatives
C
DIMENSION PM(2l),B(2l),C(201),CUR(201),E(201),PP(21,201),R(201),AA
1(200),BB(200),X(40),W(40),CUG(40),PG(21,40),db(21),bo(21),de(201),
leo(201),thd(201),tho(201),d(21,43),bin(21),bd(21,21),
lxarr(51,51),err(51,51),ord(51,51),ans(51)
C
COMMON/a/
E,CUR,C,LMAX,TPLUS,AN,EXCH,ALPHA,BETA,GAMMA,TAF,AA,BB,
1C1,C2,EX
common/b/nmax,pp,im,x,dz,r,pg,w,pm,v
C
101 FORMAT (6H ERROR,I4)
102 FORMAT (3H N-,F10.4,10H , TPLUS-,F8.4,6H , V-,F10.5,8H , AVG=,
1F10.6/41H0 R C CUR ETA/(4F11.5))
103 FORMAT (3I4)
104 FORMAT (9E8.4)
105 FORMAT (7H1ALPHA=,F8.4,9H , BETA=,F8.4,10H , GAMMA=,F8.4,9H , E
1XCH=,F8.4)
107 FORMAT (2HOB,F9.5,5F11.5/(6F11.5))
108 FORMAT (6E12.9)
109 format(* numerical derivatives*)
C
print 109
READ 103, LMAX,NMAX
nmax = 2*nmax + 1
EX = 2.0/3.0
C
DO 29 L=1,LMAX
   A = L
   AA(L) = 2.0*A**EX - (A+1.0)**EX - (A-1.0)**EX
29 BB(L) = A**EX - (A-1.0)**EX
C
DZ = 1.0/(LMAX-1)
C
DO 1 L=1,LMAX
   Z = (L-1)*DZ
   R(L) = Z**(1.0/3.0)
   ETA = SQRT(1.0-R(L)*R(L))
1 DO 1 N=1,NMAX
1   PP(N,L) = P(2*N-2,ETA)
C
DO 2 N=1,NMAX
   PM(N) = - 0.6366198/P(2*N-2,0.0)**2
   EX = 1.0/3.0
   C2 = 1.11984652
READ 103, IH,IIH,IM
IHP1 = IH + 1
READ 108, (X(I),I=IHP1,IM)
READ 108, (W(I),I=IHP1,IM)

DO 34 I=1,IM
   IF (I-IH) 31,31,32
31   IR = IM - I + 1
   X(I) = 0.5 - 0.5*X(IR)
   W(I) = W(IR)
   GO TO 33
32   X(I) = 0.5 + 0.5*X(I)
33   XX = SQRT(1.0-X(I)**2)
   DO 34 N=1,NMAX
34   PG(N,I) = P(2*N-2,XX)

3 READ 104, C(1),AN,TPLUS,ALPHA,BETA,GAMMA,EXCH,DAMP
   C1 = C2*(1.0-C(1))
   IF (C(1)) 4,4,5
4 STOP
5 JCOUNT = 0
   TAF = 1.0

   Test for Tafel approximation

   IF (EXCH-400) 7,7,6
6   TAF = 0.0
   EXCH = 1.0
7   ETAC = ALOG(C(1)) + TPLUS*(1.0-C(1))
   CUR(1) = (1.0-C(1))*AN*EXCH*1.11984652/C(1)**GAMMA
   ETAS = -ALOG(TAF-CUR(1))/BETA
   IF (CUR(1)) 8,10,8
8   DO 9 J=1,100
      F = TAF*EXP(ALPHA*ETAS) - EXP(-BETA*ETAS)
      FP = TAF*ALPHA*EXP(ALPHA*ETAS) + BETA*EXP(-BETA*ETAS)
      IF (ABS(CUR(1)-F) < 0.0000001*ABS(CUR(1))) 10,10,9
9   ETAS = ETAS + (CUR(1)-F)/FP
10  CUR(1) = 1.11984652*AN*(1.0-C(1))
   DO 11 L=1,LMAX
      C(L) = C(1)
11  E(L) = ETAC + ETAS
   B(1) = 0.0
   PRINT 105, ALPHA,BETA,GAMMA,EXCH
   BOLD = B(1)
   JCOUNT = JCOUNT + 1
   j=1
   CALL THETA(x1,dx1,x2,dx2,x3,x1o,dx1o,x2o,dx2o,x3o,denom,j)

   DO 16 I=1,IM
LI = X(I)**3/DZ + 1
16 CUG(I)= CUR(LI)+(CUR(LI+1)-CUR(LI))*X(I)**2-R(LI)**2)/(R(LI+1)**2
1-R(LI)**2)

V = E(I)
   DO 15 N=1,NMAX
       B(N) = 0.0
   DO 14 I=1,IM
14     B(N) = B(N) + CUG(I)*X(I)*PG(N,I)*W(I)
       B(N) = 0.5*B(N)*(4*N-3)/PM(N)
15     V= V + B(N)*PP(N,1)

bold = b(1)

do 310 n=1,nmax
   xarr(n,jcount) = b(n)
   print 160,jcount,b(n),bold,xarr(n,jcount)
310 continue

jcount = jcount + 1

Calculate numerical derivatives.

do 110 i=1,nmax
   bo(i) = b(i)
   jdum = 0
   call bees(bo,b,jdum,x1,dx1,x2,dx2,x3,xlo,dxlo,x2o,dx2o,x3o,
   1 denom)
x1o = x1
dx1o = dx1
x2o = x2
dx2o = dx2
x3o = x3

do 111 l=1,lmax
   tho(l) = c(l)
111   eo(l) = e(l)

c = fact = 1.e-10
   do 120 j=1,nmax
      denom = fact*bo(j)
      bo(j) = (1. + fact)*bo(j)
      call bees(bo,db,j,x1,dx1,x2,dx2,x3,xlo,dxlo,x2o,dx2o,x3o,
      1 denom)
x1d = x1
dx1d = dx1
x2d = x2
dx2d = dx2
x3d = x3

do 133 l=1,lmax
   thd(l) = c(l)
CURDBN.FOR

133 de(1) = e(1)
bo(j) = bo(j)/(1.+fact)
print 131,j
131 format(* dbs column*,i2)
den = fact*bo(j)
do 130 n=1,nmax
   db(n) = (db(n) - b(n))/den
   bd(n,j) = db(n)
130 continue
do 132,(db(n),n=1,nmax)
132 format(1x,5e12.4)
do 134 l=1,lmax
   thd(1) = (thd(1) - tho(1))/den
134 de(1) = (de(1) - eo(1))/den
denom = den
xld = (xld-xlo)/den
dxld = (dxld-dxlo)/den
x2d = (x2d-x2o)/den
dx2d = (dx2d-dx2o)/den
x3d = (x3d-x3o)/den
120 continue
do 90 i=1,nmax
90 bin(i) = b(i)
   j = 0
   CALL THETA(x1,dx1,x2,dx2,x3,x1o,dx1o,x2o,dx2o,x3o,denom,j)
DO 216 I=1,IM
LI = X(I)**3/DZ + 1
216 CUG(I)= CUR(LI)+(CUR(LI+1)-CUR(LI))*X(I)**2-R(LI)**2)/(R(LI+1)**2-R(LI)**2)
V = E(1)
DO 215 N=1,NMAX
   B(N) = 0.0
   DO 214 I=1,IM
214 B(N) = B(N) + CUG(I)*X(I)*PG(N,I)*W(I)
215 B(N) = 0.5*B(N)*(4*N-3)/PM(N)
do 210 i=1,nmax
210 do 210 j=1,mmax
   d(i,j) = 0.
do 211 i=1,nmax
211 do 211 j=1,mmax
   xi = 0.
   if(i.eq.j) xi = 1.
211 bd(i,j) = xi - bd(i,j)
do 112 i=1,nmax
112  d(i,1) = b(i) - bin(i)
c                        call matinv(nmax,mmax,determ,bd,d)
   if(determ.eq.0.) print 116
116 format(27h zero determinant in matinv)
c                        do 117 i=1,nmax
117  b(i) = b(i) + d(i,1)
c                        do 320 n=1,nmax
320  xarr(n,jcount) = b(n)
c                        JERR= 1
c                        IF (JCOUNT-50) 19,19,20
19  crit=(ABS(B(1)-BOLD) - 0.000001*ABS(B(1)))
c                        do 330 n=1,nmax
330  print 160,jcount,b(n),bold,xarr(n,jcount)
160  format(* jcount=*,i2,* b(n)=*,1pe11.4,* bold=*,1pe11.4,* xarr=*,
1     1 1pe11.4)
330  continue

c                        IF (ABS(B(1)-BOLD) - 1.e-12*ABS(B(1))) 21,21,12
21  print 101, JERR
21   AVG = - B(1)/0.7853982/AN/1.11984652

c                        Test errors for quadratic convergence
c                        do 340 n=1,nmax
340  ans(n) = b(n)
340  continue

c                        itfin = jcount
d                        do 350 n=1,nmax
350  print 70,n,ans(n),itfin
70  format(* n=*,i2,* answer=*,1pe17.10,* itfin=*,i2)
350  continue

c                        print 59
59  format(* it err(n,it)*)
d                        do 30 n=1,nmax
30  print 61,n
61  format(* n=*,i2)
30  do 30 it=1,itfin
60  err(n,it) = xarr(n,it) - ans(n)
60  print 60,it, err(n,it)
60  format(1x,i3,5x,1pe15.8)
30  continue

c
print 91
91 format(* it ord(it)*)
c
itfm1 = itfin - 1
do 35 n=1,nmax
   print 61,n
   do 35 it=1,itfm1
      ord(n,it) = 0.
      if(err(n,it).eq.0 .. or. err(n,it+1).eq.0.) go to 36
      ord(n,it) = alog(abs(aalog(abs(err(n,it+1)/err(n,it)))))
36     print 60,it,ord(n,it)
35 continue
c
itfm2 = itfin - 2
c
do 37 n=1,nmax
   print 61,n
   do 37 it=1,itfm2
      slope = ord(n,it+1) - ord(n,it)
      if(ord(n,it+1).eq.0.) go to 37
      print 80,it,slope,exp(slope)
80 format(1x,i3,5x,lpe11.4,5x,ell.4)
37 continue
c
dO 22 L-1,LMAX
22 CUR(L) = CUR(L)/AN/1.11984652
   RAT1 = CUR(L)/AVG
   RAT2 = (V-E(1))/B(1)
   PRINT 102, AN,TPLUS,V,AVG,(R(J),C(J),CUR(J),E(J),J=1,LMAX)
   PRINT 107, (B(I),I=1,NMAX),TAF,RAT2,RAT1
   PRINT 103, JCOUNT
   GO TO 3
END
c
cSUBROUTINE THETA(x1,dx1,x2,dx2,x3,x1o,dx1o,x2o,dx2o,x3o,demom,
   1 jdum)
cSUBROUTINE FOR CALCULATING CONCENTRATION by equating Fick's-law
   and Faraday's-law fluxes
   (X1-X2)/PN + (C1+C3))EXCH = 0
c
DIMENSION E(201),CUR(201),A(200),B(200),TH(201)
c
COMMON/a/ E,CUR,TH,NZT1,T,PN,EXCH,AL,BE,GAM,TAF,A,B,CL,C2,EX
c
101 FORMAT (17HONOT CONVERGED AT,I4)
c
NZT= NZT1-1
DEVM = 1.e-10
c DEVM = 0.0001
DZ = 1.0/NZT
S = TH(1)

DO 60 NZ = 2, NZT1
Z = (NZ - 1)*DZ
SUM = 0.0
IF (NZ .LE. 2) GO TO 42

CALC. SUM(TH(J)*A(K))

DO 40 J = 3, NZ
K = NZ - J + 1
SUM = SUM + TH(J-1)*A(K)
40 ETA = E(NZ)
NJ = NZ - 1

DO 56 N = 1, 20
X1 = TAF*S**(GAM - AL)*EXP(AL*ETA)*EXP(AL*T*(S - 1.0))
DX1 = X1*((GAM - AL)/S + AL*T)
X2 = S**((GAM + BE)*EXP(-BE*ETA)*EXP(BE*T*(1.0 - S))
DX2 = X2*((GAM + BE)/S - BE*T)
C3 = 1.50*C2*Z**EX/DZ**(1./3.)
X3 = C3*(TH(1)*B(NJ) + SUM - S)
DTH = S - ((X1-X2)/PN + (C1+X3)*EXCH)/((DX1-DX2)/PN - C3*EXCH)
CUR(NZ) = PN*(C1+X3)

IF (JDUM.NE.2 .OR. NZ.NE.2) GO TO 44
XLS = X1
DXLS = DX1
X2S = X2
DX2S = DX2
X3S = X3
XLD = (X1-XLO)/DENOM
DXLD = (DX1-DXLO)/DENOM
X2D = (X2-X2O)/DENOM
DX2D = (DX2-DX2O)/DENOM
X3D = (X3-X3O)/DENOM
44 CONTINUE
IF (ABS(S-DTH) - DEVM*ABS(DTH)) 60, 60, 56

S = DTH

PRINT 101, NZ
60 TH(NZ) = DTH

RETURN
END

FUNCTION P(N,X)

CALCULATION OF LEGENDRE POLYNOMIALS
P1 = 1.0
P2 = X
IF (N-1) 1, 2, 3
1 P = P1
RETURN
2 P = P2
RETURN
3 NM1 = N - 1
   DO 4 NU = 1, NM1
      P = (X * FLOAT (2*NU+1) * P2 - FLOAT (NU) * P1) / FLOAT (NU+1)
      P1 = P2
   4 P2 = P
   RETURN
END

subroutine bees(bo, b, j, x1s, dx1s, x2s, dx2s, x3s, x1o, dx1o, x2o, dx2o, 
   x3o, denom)
DIMENSION PM(21), B(21), C(201), CUR(201), E(201), PP(21, 201), R(201), AA 
   L(200), BB(200), X(40), W(40), CUG(40), PG(21, 40), bo(21)
COMMON/a/E, CUR, C, LMAX, TPLUS, AN, EXCH, ALPHA, BETA, GAMMA, TAF, AA, BB, 
   1C1, C2, EX
COMMON/b/nmax, pp, im, x, dz, r, pg, w, pm, v
v = e(1)
do 19 n = 1, nmax
19 v = v + bo(n) * pp(n, 1)
do 18 l = 2, lmax
   phi = v
   do 17 n = 1, nmax
17 phi = phi - bo(n) * pp(n, 1)
e(1) = phi
CALL THETA(x1s, dx1s; x2s, dx2s, x3s, x1o, dx1o, x2o, dx2o, x3o, denom, j)
DO 16 I = 1, IM
   LI = X(I)**3/DZ + 1
16 CUG(I) = CUR(LI) + (CUR(LI+1) - CUR(LI)) * (X(I)**2 - R(LI)**2) / (R(LI+1)**2 
   - R(LI)**2)
DO 15 N = 1, NMAX
   B(N) = 0.0
   DO 14 I = 1, IM
14 B(N) = B(N) + CUG(I) * X(I) * PG(N, I) * W(I)
15 B(N) = 0.5 * B(N) * (4*N - 3)/PM(N)
return
end
SUBROUTINE MATINV(N, M, DETERM, b, d)
This subroutine solves a matrix equation of the form $BX = D$. The answer $X$ is put into the first column of $D$. Note that $D$ should be dimensioned $nxm$, where $m=2n+1$ and that all columns of $D$ (except the first) should be initialized with zeros.

**Input variables**
- $n$ = dimension of square $b$ matrix
- $m$ = number of columns in $d$ matrix
- $b$ = matrix in equation $bx = d$
- $d$ : first column contains vector $d$ in $bx = d$

**Output variables**
- $d$ : first column contains vector $x$ in $bx = d$
- $\text{determ} = 0.$ if matrix has zero determinant

```
DIMENSION ID(2l), b(2l, 2l), d(2l, 43)
```

```
DETERM=1.0
DO 1 I=1,N
  ID(I)=0
DO 18 NN=1,N
  BMAX=1.1
  DO 6 I=1,N
    IF(ID(I).NE.0) GOTO 6
    BNEXT=0.0
    BTRY=0.0
    DO 5 J=1,N
      IF(ID(J).NE.0) GOTO 5
      IF(ABS(B(I,J)).LE.BNEXT) GOTO 5
      BNEXT=ABS(B(I,J))
      BTRY=ABS(B(I,J))
      JC=J
    5 CONTINUE
    IF(BNEXT.GE.BMAX*BTRY) GOTO 6
    BMAX=BNEXT/BTRY
    IROW=I
    JCOL=JC
  6 CONTINUE
  IF(ID(JC).EQ.0) GOTO 8
  DETERM=0.0
  RETURN
  8 ID(JCOL)=1
  IF(JCOL.EQ.IROW) GOTO 12
  DO 10 J=1,N
    SAVE=B(IROW,J)
    B(IROW,J)=B(JCOL,J)
  10 B(JCOL,J)=SAVE
  DO 11 K=1,M
    SAVE=D(IROW,K)
  11 D(IROW,K)=SAVE
```
D(IRGW,K)=D(JCOL,K)
11 D(JCOL,K)=SAVE
12 F=1.0/B(JCOL,JCOL)
   DO 13 J=1,N
13 B(JCOL,J)=B(JCOL,J)*F
   DO 14 K=1,M
14 D(JCOL,K)=D(JCOL,K)*F
   DO 18 I=1,N
15 IF(I.EQ.JCOL) GO TO 18
   F=B(I,JCOL)
   DO 16 J=1,N
16 B(I,J)=B(I,J)-F*B(JCOL,J)
   DO 17 K=1,M
17 D(I,K)=D(I,K)-F*D(JCOL,K)
18 CONTINUE
RETURN
END

-789

41 21
20 10 40
3877241751-21160840707-11926975807-12681521850-13419940908-14137792044-1
4830758017-15494671251-16125538897-16719566846-17273182552-17783056514-1
8246122308-18659595032-19020988070-19328128083-19579168192-19772599500-1
9907262387-19982377097-1
7750594798-27703981816-27611036190-27472316906-27288658240-27061164739-2
1791204582-26480401346-26130624249-25743976910-25322784698-24869580764-2
4387090819-23878216797-23346019528-22793700698-22224584919-21642105838-2
1049828453-24521277099-3
50000-2 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 10000-0
20000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 80000-1
30000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 60000-1
40000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 40000-1
50000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 30000-1
60000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 30000-1
70000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 30000-1
80000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 30000-1
90000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 30000-1
95000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 80000-1
80000-1 20000+1 50000-1 50000-1 50000-1 50000-1 50000+2 70000-1
80000-1 50000+1 50000-1 50000-1 50000-1 50000-1 50000+2 50000-1
20000-1 50000-0 50000-1 50000-1 50000-1 50000-1 50000+2 80000-1
50000-1 10000+2 50000-1 50000-1 50000-1 50000-1 50000+2 30000-1
50000-1 10000+1 50000-1 50000-1 50000-1 50000-1 50000+2 70000-1
50000-1 20000+1 50000-1 50000-1 50000-1 50000-1 50000+2 60000-1
E.3. Program CURDE

Description:


Input Variables

LMAX = number of radial mesh points
NMAX = number of B coefficients
IH = half the number of Gaussian quadrature points
IHH : not used
IM = total number of Gaussian quadrature points
X(I) = Gaussian quadrature roots (in the interval 0 to 1)
W(I) = Gaussian quadrature weights
C(I) = \( c/c_\infty \) at center of disk (input C(1) = 0 to stop running cases)
AN = dimensionless limiting current = \( N \)
TPLUS = transference number
ALPHA = anodic transfer coefficient
BETA = cathodic transfer coefficient
GAMMA = exponent for concentration dependence of exchange current density
EXCH = dimensionless exchange current density = \( J \)
DAMP = damping factor (not used)

Output Variables

JCOUNT = iteration number
BOLD = B(1) at previous iteration
CRIT = local error = |B(1) - BOLD|
ERR = 1 if too many iterations, 0 otherwise
ANS(N) = converged B(n)
ITFIN = number of iterations required for convergence
ERR(N,IT) = error in B(n) at iteration IT = \( e_{IT} \)
ORD(N,IT) = \( \ln \left| \ln \left| \frac{e_{IT+1}}{e_{IT}} \right| \right| \)
SLOPE = slope of ORD vs. IT
V = disk potential
AVG = \( \delta/N = \frac{i_{avg}}{i_{lim}} \)
R(J) = radial position
C(J) = surface concentration \( c/c_\infty \)
CUR(J) = dimensionless current density, \( i/i_0 \)
E(J) = total overpotential = \( V - \Phi_0 \)
B(I) = B coefficients
TAF = 0 if Tafel approximation is used, 1 if not
RAT2 = (V - E(1))/B(1), proportional to effective resistance
RAT1 = CUR(1)/(AVG*N*\( \theta' \)(0)), ratio of current at center to average current
JCOUNT = iterations required
PROGRAM CURDE(INPUT,OUTPUT)
C CURRENT DISTRIBUTION ON A ROTATING DISK WITH AN INTEGRAL EQUATION
C FOR THE DIFFUSION LAYER
C
This is program curd rewritten for iteration on overpotentials
by a Newton-Raphson method
C
DIMENSION PM(21),B(21),C(201),CUR(201),E(201),PP(21,201),R(201),AA
1(200),BB(200),X(40),W(40),CUG(40),PG(21,40),ed(41,41),d(41,83),
lein(41)
C
COMMON/a/ E,CUR,C,LMAX,TPLUS,AN,EXCH,ALPHA,BETA,GAMMA,TAF,AA,BB,C1
1,C2,EX
common/b/nmax,pp,im,x,dz,r,pg,w,pm

101 FORMAT (6H ERROR,I4)
102 FORMAT (3H N-,F10.4,10H , TPLUS-,F8.4,6H , V-,F10.5,8H , AVG-,
1F10.6/41H0 R C CUR ETA/(4F11.5))
103 FORMAT (3I4)
104 FORMAT (9E8.4)
105 FORMAT (7H1ALPHA-,F8.4,9H , BETA-,F8.4,1OH , GAMMA-,F8.4,9H , E
1XCH-,F8.4)
107 FORMAT (2HOB,F9.5,5F11.5/(6F11.5))
108 FORMAT (6E12.9)
C
READ 103, LMAX,NMAX
mmax = 2*lmax+1
EX = 2.0/3.0
C
DO 29 L=1,LMAX
  A= L
  AA(L) = 2.0*A**EX - (A+1.0)**EX - (A-1.0)**EX
29 BB(L) = A**EX - (A-1.0)**EX
C
DZ= 1.0/(LMAX-1)
C
DO 1 L=1,LMAX
  Z = (L-1)*DZ
  R(L) = Z**(1.0/3.0)
  ETA = SQRT(1.0-R(L)*R(L))
DO 1 N=1,NMAX
1 PP(N,L)= P(2*N-2,ETA)
C
DO 2 N=1,NMAX
2 PM(N)= - 0.6366198/P(2*N-2,0.0)**2
C
EX = 1.0/3.0
C2 = 1.11984652
READ 103, IH, IHH, IM
IHP1 = IH + 1
READ 108, (X(I), I=IHP1, IM)
READ 108, (W(I), I=IHP1, IM)

DO 34 I=1, IM
  IF (I-IH) 31, 31, 32
  IR = IM - I + 1
  X(I) = 0.5 - 0.5*X(IR)
  W(I) = W(IR)
  GO TO 33
32 X(I) = 0.5 + 0.5*X(I)
33 XX = SQRT(1.0-X(I)**2)
DO 34 N=1, NMAX
34 PG(N, I) = P(2*N-2, XX)

3 READ 104, C(1), AN, TPLUS, ALPHA, BETA, GAMMA, EXCH, DAMP
C1 = C2*(1.0-C(1))
IF (C(1)) 4, 4, 5
STOP
5 JCOUNT = 0
TAF = 1.0

Test for Tafel approximation
IF (EXCH-400) 7, 7, 6

6 TAF = 0.0
EXCH = 1.0
7 ETAC = ALOG(C(1)) + TPLUS*(1.0-C(1))
CUR(1) = -(1.0-C(1))*AN*EXCH*1.11984652/C(1)**GAMMA
ETAS = -ALOG(TAF-CUR(1))/BETA

IF (CUR(1)) 8, 10, 8

8 DO 9 J=1, 100
  F = TAF*EXP(ALPHA*ETAS) - EXP(-BETA*ETAS)
  FP = TAF*ALPHA*EXP(ALPHA*ETAS) + BETA*EXP(-BETA*ETAS)
  IF (ABS(CUR(1)-F) - 0.0000001*ABS(CUR(1))) 10, 10, 9
9 ETAS = ETAS + (CUR(1)-F)/FP
10 CUR(1) = 1.11984652*AN*(1.0-C(1))

DO 11 L=1, LMAX
C(L) = C(1)
11 E(L) = ETAC + ETAS

B(1) = 0.0
PRINT 105, ALPHA,BETA,GAMMA,EXCH
12 BOLD = B(1)
   JCOUNT = JCOUNT + 1

   do 90 1=1,lmax
90 ein(1) = e(1)
   call theta

   DO 16 ii=1,IM
      LI = X(ii)**3/DZ + 1
   16 CUG(ii) = CUR(LI)+(CUR(LI+1)-CUR(LI))*(X(ii)**2-R(LI)**2)/
                1(R(LI+1)**2-R(LI)**2)

   V = E(1)
   DO 15 N=1,NMAX
      B(N) = 0.0
   DO 14 ii=1,IM
14      B(N) = B(N) + CUG(ii)*X(ii)*PG(N,ii)*W(ii)
      B(N) = 0.5*B(N)*(4*N-3)/PM(N)
   15 V = V + B(N)*PP(N,L)

   calculate Jacobian

   do 100 1=1,lmax
100   call deriv(1,ed,b,v)

   do 110 i=1,lmax
   do 110 j=1,mmax
110      d(i,j) = 0.
   do 111 i=1, lmax
   do 111 j=1, lmax
111     xi = 0.
   if(i.eq.j) xi = 1.0
     ed(i,j) = xi - ed(i,j)

   do 112 i=1,lmax
112     d(i,l) = e(i) - ein(i)
   call matinv(lmax,mmax,determ,ed,d)
   if(determ.eq.0.) print 116
116   format(27h zero determinant in matinv)

   update

   do 117 i=1,lmax
117     e(i) = ein(i) + d(i,l)
JERR = 1
IF (JCOUNT-50) 19,19,20
19 crit=(ABS(B(1)-BOLD))
   print 160,jcount,b(1),bold,crit
160 format(* jcount=*,i2,* b(1)=*,1pe11.4,* bold=*,1pe11.4,* crit=*,
   1 lpe11.4)
   IF (ABS(B(1)-BOLD) - 1.e-09*ABS(B(1))) 21,21,12
20 PRINT 101, JERR
21 AVG= - B(1)/0.7853982/AN/1.11984652
   DO 22 L=1,LMAX
22 CUR(L)= CUR(L)/AN/1.11984652
   RAT1= CUR(1)/AVG
   RAT2= (V-E(1))/B(1)
   PRINT 102, AN,TPLUS,V,AVG,(R(J),C(J),CUR(J),E(J),J=1,LMAX)
   PRINT 107, (B(I),I=1,NMAX),TAF,RAT2,RAT1
   PRINT 103, JCOUNT
   GO TO 3
END

SUBROUTINE THETAp(de,je,dcur,thd)
C SUBPROGRAM FOR CALCULATING CONCENTRATION derivatives
C This subroutine is the derivative of subroutine theta
C with respect to E(je)
C
DIMENSION E(201),CUR(201),A(200),B(200),TH(201),thd(201),de(201)
   ldcurl(201)
C
COMMON/a/ E,CUR,TH,NZT1,T,PN,EXCH,AL,BE,GAM,TAF,A,B,C1,C2,EX
C
101 FORMAT (17HONOT CONVERGED AT,I4)
C
NZT = NZT1-1
DEVM = 0.0001
DZ = 1.0/NZT
S = TH(1)
ds=0.
   thd(1)=0.
C
   DO 60 NZ = 2,NZT1
   ds=0.
   Z = (NZ - 1)*DZ
   SUM = 0.0
   dsum = 0.0
   IF (NZ .LE. 2) GO TO 42
C
C CALC. SUM(TH(J)*A(K))
DO 40 J=3,NZ
   K = NZ - J + 1
   dsum = dsum + thd(j-1)*a(k)
40 SUM = SUM + TH(J-1)*A(K)
42 ETA = E(NZ)

derivative of e(nz) with respect to e(je)

deta = de(nz)

S = th(nz)

S is not updated because the thetas were calculated in subroutine theta

NJ = NZ - 1
X1 = TAF*S**(GAM - AL)*EXP(AL*ETA)*EXP(AL*T*(S - 1.0))
DX1 = X1*((GAM - AL)/S + AL*T)
    xl = al*xl*deta
X2 = S**(GAM + BE)*EXP(-BE*ETA)*EXP(BE*T*(1.0 - S))
DX2 = X2*((GAM + BE)/S - BE*T)
    x2 = -be*x2*deta
C3 = 1.50*C2*Z**EX/DZ**(1.3.)
X3 = C3*(TH(1)*B(NJ) + SUM - S)
    dx3 = c3*dsum
dthd = - ((xl-x2)/pn + dx3*exch)/((xl-dx2)/pn - c3*exch)
dCUR(NZ) = PN*c3*(dsum-dthd)
xldn = xld + dx1*dthd
x2dn = x2d + dx2*dthd
dx3n = dx3 - c3*dthd
thd(nz) = dthd
60 continue

RETURN
END

FUNCTION P(N,X)
C
CALCULATION OF LEGENDRE POLYNOMIALS
P1= 1.0
P2= X
IF (N-1) 1,2,3
1 P= P1
RETURN
2 P= P2
RETURN
3 NM1= N - 1
DO 4 NU=1,NM1
P=(X*FLOAT (2*NU+1)*P2-FLOAT (NU)*P1)/FLOAT (NU+1)
P1= P2
4 P2= P
RETURN
END

SUBROUTINE MATINV(N,M,DETERM,b,d)
This subroutine solves a matrix equation of the form BX=D. Note that
D should be dimensioned nxm, where m=2n+1 and that all
columns of D (except the first) should be initialized with
zeros

Input variables
n = dimension of square b matrix
m = number of columns in d matrix
b = matrix in equation bx=d
d : first column contains vector d in bx=d

Output variables
d : first column contains vector x in bx=d
determ = 0. if matrix has zero determinant

DIMENSION ID(41),b(41,41),d(41,83)

DETERM=1.0
DO 1 I=1,N
1 ID(I)=0
DO 18 NN=1,N
BMAX=1.1
DO 6 I=1,N
IF(ID(I).NE.0) GOTO 6
BNEXT=0.0
BTRY=0.0
DO 5 J=1,N
IF(ID(J).NE.0) GOTO 5
IF(ABS(B(I,J)).LE.BNEXT) GOTO 5
BNEXT=ABS(B(I,J))
IF(BNEXT.LE.BTRY) GOTO 5
BNEXT=BTRY
BTRY=ABS(B(I,J))
JC=J
5 CONTINUE
IF(BNEXT.GE.BMAX*BTRY) GOTO 6
BMAX=BNEXT/BTRY
CURRE.FOR

IROW=I
JCOL=JC
6 CONTINUE
IF(ID(JC).EQ.0) GOTO 8
DETERM=0.0
RETURN
8 ID(JCOL)=1
IF(JCOL.EQ.IROW) GOTO 12
DO 10 J=1,N
SAVE=B(IROW,J)
B(IROW,J)=B(JCOL,J)
10 B(JCOL,J)=SAVE
DO 11 K=1,M
SAVE=D(IROW,K)
D(IROW,K)=D(JCOL,K)
11 D(JCOL,K)=SAVE
12 F=1.0/B(JCOL,JCOL)
DO 13 J=1,N
13 B(JCOL,J)=B(JCOL,J)*F
DO 14 K=1,M
14 D(JCOL,K)=D(JCOL,K)*F
DO 18 I=1,N
IF(I.EQ.JCOL) GO TO 18
F=B(I,JCOL)
DO 16 J=1,N
16 B(I,J)=B(I,J)-F*B(JCOL,J)
DO 17 K=1,M
17 D(I,K)=D(I,K)-F*D(JCOL,K)
18 CONTINUE
RETURN
END

subroutine deriv(i,ed,b,v)

This subroutine calculates the derivatives of the calculated
overpotentials with respect to the input overpotentials, and
it updates the overpotentials the last time it is called.

DIMENSION PM(21),B(21),C(201),CUR(201),E(201),PP(21,201),R(201),AA
1(200),BB(200),X(40),W(40),CUG(40),PG(21,40),ed(41,41),d(41,83),
lein(41),bd(41),dcur(201),de(201),thd(201),dcug(40)

COMMON/a/E,CUR,C,LMAX,TPLUS,AN,EXCH,ALPHA,BETA,GAMMA,TAF,AA,BB,Cl
1,C2,EX
common/b/nmax,pp,im,x,dz,r,pg,w,pm

de is derivative of overpotentials with respect to e(i)
SUBROUTINE THETA
C SUBPROGRAM FOR CALCULATING CONCENTRATION by equating Fick's-law
and Faraday's-law fluxes
C (X1-X2)/PN + (C1+X3)*EXCH = 0
C
DIMENSION E(201),CUR(201),A(200),B(200),TH(201)
C
COMMON/a/ E,CUR,TH,NZT1,T,PN,EXCH,AL,BE,GAM,TAF,A,B,Cl,C2,EX
C
101 FORMAT (17HONOT CONVERGED AT,I4)
NZT = NZT1 - 1
DEVM = 0.0001
DZ = 1.0/NZT
S = TH(1)
DO 60 NZ = 2,NZT1
Z = (NZ - 1)*DZ
SUM = 0.0
IF (NZ .LE. 2) GO TO 42

CALC. SUM(TH(J)*A(K))

DO 40 J=3,NZ
K = NZ - J + 1
40 SUM = SUM + TH(J-1)*A(K)
42 ETA = E(NZ)
NJ = NZ - 1

DO 56 N=1,20
X1 = TAF*S**(GAM - AL)*EXP(AL*ETA)*EXP(AL*T*(S - 1.0))
DX1 = X1*(((GAM - AL)/S + AL*T)
X2 = S**((GAM + BE)*EXP(-BE*ETA)*EXP(BE*T*(1.0 - S))
DX2 = X2*(((GAM + BE)/S - BE*T)
C3 = 1.50*C2*Z**EX/DZ**(1./3.)
X3 = C3*(TH(1)*B(NJ) + SUM - S)
DTH = S - ((X1-X2)/PN + (C1+X3)*EXCH)/((DX1-DX2)/PN - C3*EXCH)
CUR(NZ) = PN*(C1+X3)
IF (ABS(S-DTH) - DEVM*ABS(DTH)) 60,60,56
56 S = DTH

PRINT 101, NZ
60 TH(NZ) = DTH
RETURN
END

-789
41 21
20 10 40
3877241751-21160840707-11926975807-12681521850-13419940908-14137792044-1
4830758017-15494671251-16125538897-16719566846-17273182552-17783056514-1
8246122308-18659595032-19020988070-19328128083-19579168192-19772599500-1
9907262387-19982377097-1
7750594798-27703981816-27611036190-27472316906-27288658240-27061164739-2
179120582-26480401346-26130624249-25743976910-25322784698-24869580764-2
4387090819-23878216797-23346019528-22793700698-2224584919-21642105838-2
1049828453-24521277099-3
70000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 30000-1
50000-2 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 10000-0
20000-1 78800+1 36300-1 15000-0 50000-1 42000-1 89600-1 80000-1
<table>
<thead>
<tr>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>30000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>60000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>40000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>30000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>60000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>30000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>70000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>30000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>30000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>90000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>30000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>95000-1</td>
<td>78800+1</td>
<td>36300-1</td>
<td>15000-0</td>
<td>50000-1</td>
<td>42000-1</td>
<td>89600-1</td>
<td>80000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80000-1</td>
<td>20000+1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000+2</td>
<td>70000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80000-1</td>
<td>50000+1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000+2</td>
<td>50000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000+2</td>
<td>80000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50000-1</td>
<td>10000+2</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000+2</td>
<td>30000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50000-1</td>
<td>10000+1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000+2</td>
<td>70000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50000-1</td>
<td>20000+1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000-1</td>
<td>50000+2</td>
<td>60000-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
E.4. Program CURDEN

Description:


Input Variables

LMAX = number of radial mesh points
NMAX = number of B coefficients
IH = half the number of Gaussian quadrature points
IHH : not used
IM = total number of Gaussian quadrature points
X(I) = Gaussian quadrature roots (in the interval 0 to 1)
W(I) = Gaussian quadrature weights
C(1) = c/c∞ at center of disk (input C(1) = 0 to stop running cases)
AN = dimensionless limiting current = N
TPLUS = transference number
ALPHA = anodic transfer coefficient
BETA = cathodic transfer coefficient
GAMMA = exponent for concentration dependence of exchange current density
EXCH = dimensionless exchange current density = J
DAMP = damping factor (not used)

Output Variables

ERR = 1 if too many iterations, 0 otherwise
ANS(N) = converged B(n)
ITFIN = number of iterations required for convergence
ERR(N,IT) = error in B(n) at iteration IT = $e_{IT}$
ORD(N,IT) = ln | $\frac{e_{IT+1}}{e_{IT}}$ |
SLOPE = slope of ORD vs. IT
V = disk potential
AVG = $\delta/N = \frac{i_{avg}}{i_{lim}}$
R(J) = radial position
C(J) = surface concentration $c/c_\infty$
CUR(J) = dimensionless current density, $i/\langle i \rangle$
E(J) = total overpotential = $V - \Phi_\theta$
B(I) = B coefficients
TAF = 0 if Tafel approximation is used, 1 if not
RAT2 = $(V - E(1))/B(1)$, proportional to effective resistance
RAT1 = CUR(1)/(AVG*N*[$\theta'(0)$]), ratio of current at center to average current
JCOUNT = iterations required
PROGRAM CURDEN(INPUT,OUTPUT)

CURRENT DISTRIBUTION ON A ROTATING DISK WITH AN INTEGRAL EQUATION
FOR THE DIFFUSION LAYER

This is program curd rewritten for iteration on overpotentials
by a Newton-Raphson method with numerical derivatives

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

DIMENSION PM(2l),B(2l),C(201),CUR(201),E(201),PP(21,201),R(201),AA
1(200),BB(200),X(40),W(40),CUG(40),PG(21,40),de(201),eo(201),
1thd(201),tho(201),d(21,83),ein(41),ed(41,41),dde(41),bo(21),bd(21)
1,curo(41),cursd(41),co(41),cd(41)

COMMON/a/ CUR,C,LMAX,TPLUS,AN,EXCH,ALPHA,BETA,GAMMA,TAF,AA,BB,
1C1,C2,EX

common/b/rmax,pp,im,x,dz,r,pg,w,pm,v

101 FORMAT (6H ERROR,I4)
102 FORMAT (3H N-,F10.4,10H , TPLUS-,F8.4,6H , V-,F10.5,8H , AVG-,
1F10.6/41H0 R C CUR ETA/(4F11.5))
103 FORMAT (3I4)
104 FORMAT (9E8.4)
105 FORMAT (7H1ALPHA-,F8.4,9H , BETA-,F8.4,10H , GAMMA-,F8.4,9H , E
1XCH-,F8.4)
107 FORMAT (2HOB,F9.5,5F11.5/(6F11.5))
108 FORMAT (6E12.9)
109 format(' numerical derivatives')

print 109
READ 103, LMAX,NMAX
mmax = 2*lmax+1
EX = 2.0d0/3.0d0

DO 29 L=1,LMAX
A = dfloat(L)
AA(L) = 2.0*A**EX - (A+1.0)**EX - (A-1.0)**EX
29 BB(L) = A**EX - (A-1.0)**EX

DZ = 1.0d0/(LMAX-1)

DO 1 L=1,LMAX
Z = (L-1)*DZ
R(L) = Z**((1.0/3.0)
ETA = dSQRT(1.0-R(L)*R(L))
DO 1 N=1,NMAX
1 \( PP(N,L) = P(2*N-2,ETA) \)

c
2 \( PM(N) = -0.6366198/P(2*N-2,0.0)**2 \)

\[ EX = 1.0d0/3.0d0 \]

\[ C2 = 1.11984652d0 \]

READ 103, IH,IHH,IM

IH1 = IH + 1

READ 108, \((X(I),I=IH1,IM)\)

READ 108, \((W(I),I=IH1,IM)\)

c
3 \( DO \ I=1,IM \)

IF (I-IH) 31,31,32

31 \( IR = IM - I + 1 \)

\[ X(I) = 0.5d0 - 0.5d0*X(IR) \]

\[ W(I) = W(IR) \]

GO TO 33

32 \( X(I) = 0.5d0 + 0.5d0*X(I) \)

33 \( XX = dSQR(1.0-X(I)**2) \)

\( DO \ N=1,NMAX \)

34 \( PG(N,I) = P(2*N-2,XX) \)

c
3 \( READ 104, C(1),AN,TPLUS,ALPHA,BETA,GAMMA,EXCH,DAMP \)

\[ C1 = C2*(1.0-C(1)) \]

IF (C(1)) 4,4,5

4 STOP

5 \( JCOUNT = 0 \)

\( TAF = 1.0d0 \)

c
c
Test for Tafel approximation

c
IF (EXCH-400) 7,7,6

c
6 \( TAF = 0.0d0 \)

\( EXCH = 1.0d0 \)

7 \( ETAS = dLOG(C(1)) + TPLUS*(1.0-C(1)) \)

\[ CUR(1) = -\( (1.0-C(1))*AN*EXCH*1.11984652/C(1)**GAMMA \) \]

\[ ETAS = -dLOG(TAF-CUR(1))/BETA \]

c
IF (CUR(1)) 8,10,8

c
8 \( DO \ J=1,100 \)

\[ F = TAF*dEXP(ALPHA*ETAS) - dEXP(-BETA*ETAS) \]

\[ FP = TAF*ALPHA*dEXP(ALPHA*ETAS) + BETA*dEXP(-BETA*ETAS) \]

IF (dABS(CUR(1)-F) - 0.0000001*dABS(CUR(1))) 10,10,9

9 \( ETAS = ETAS + (CUR(1)-F)/FP \)

c
10 \( CUR(1) = 1.11984652*AN*(1.0-C(1)) \)
DO 11 L=1,LMAX
   C(L) = C(1)
   E(L) = ETAC + ETAS
11 de(1) = e(1)
   B(1) = 0.0d0
PRINT 105, ALPHA,BETA,GAMMA,EXCH
12 BOLD = B(1)
c
   JCOUNT = JCOUNT + 1
c
do 110 1=1,LMAX
110 eo(1) = e(1)
   jdum = 0
   call ees(eo,e,b,jdum,denom,x1,dx1,x2,dx2,x3,s,sums)
   print 345
345 format(' b(n)')
   print 346,(b(n),n=1,nmax)
   346 format(1x,1pe2.4,5el1.4)
do 111 n=1,nmax
111 bo(n) = b(n)
do 113 l=1,LMAX
   co(l) = c(l)
113 curo(l) = cur(l)
   sumo = sums
   so = s
   vo = v
   xlo = x1
   dxlo = dx1
   x2o = x2
   dx2o = dx2
   x3o = x3
c
do 120 j=1,LMAX
   denom = 0.000001d0*eo(j)
   eo(j) = 1.000001d0*eo(j)
   call ees(eo,de,b,j,denom,x1,dx1,x2,dx2,x3,s,sums)
   if(j.ne.2) go to 335
do 112 n=1,nmax
112 bd(n) = b(n)
c
do 114 l=1,LMAX
   cd(l) = c(l)
114 cursd(l) = cur(l)
c
   sumd = sums
   ds = s
   vd = v
   xld = x1
CURREN.FOR

\[ dxld = dxl \]
\[ x2d = x2 \]
\[ dx2d = dx2 \]
\[ x3d = x3 \]

335 continue
\[ eo(j) = eo(j)/1.000001d0 \]

\[ do \quad 130 \quad n=1,lmax \]
\[ dde(n) = (de(n) - e(n))/(0.000001*eo(j)) \]
\[ ed(n,j) = dde(n) \]
\[ if(j.eq.2) \text{cursd}(n) = (\text{cursd}(n)-\text{curo}(n))/(0.000001*eo(j)) \]
\[ if(j.eq.2) \text{cd}(n) = (\text{cd}(n)-\text{co}(n))/(0.000001*eo(j)) \]

130 continue

if(j.ne.2) go to 120
\[ \text{denom} = 0.000001*eo(j) \]
\[ \text{sumd} = (\text{sumd}-\text{sumo})/\text{denom} \]
\[ \text{ds} = (\text{ds}-\text{so})/\text{denom} \]
\[ \text{vd} = (\text{vd}-\text{vo})/\text{denom} \]
\[ x1d = (x1d-x1o)/\text{denom} \]
\[ dx1d = (dx1d-dx1o)/\text{denom} \]
\[ x2d = (x2d-x2o)/\text{denom} \]
\[ dx2d = (dx2d-dx2o)/\text{denom} \]
\[ x3d = (x3d-x3o)/\text{denom} \]

120 continue

c

stop
\[ \text{JERR} = 1 \]
c
IFI(JCOUNT-100) 19,19,20
c

19 \[ \text{crit} = (dABS(B(l)-BOLD) - 0.000001*dABS(B(l))) \]
\[ \text{print} 160, jcount, b(l), bold, crit \]
\[ \text{format('jcount='},i2,' b(l)='},lpebl.4,' bold='},lpebl.4,' crit='},lpebl.4) \]
\[ IF (dABS(B(l)-BOLD) - 0.000001*dABS(B(l))) 21,21,12 \]

20 PRINT 101, JERR
21 AVG = -B(1)/0.7853982/AN/1.11984652
DO 22 L=1,LMAX
22 CUR(L) = CUR(L)/AN/1.11984652
\[ \text{RAT1} = \text{CUR}(1)/\text{AVG} \]
\[ \text{RAT2} = (V-E(1))/B(1) \]
\[ \text{PRINT} \quad 102, \text{AN}, \text{TPPLUS}, \text{V}, \text{AVG}, (R(J), C(J), CUR(J), E(J), J-1, LMAX) \]
\[ \text{PRINT} \quad 107, (B(I), I-1, NMAX), TAF, RAT2, RAT1 \]
\[ \text{PRINT} \quad 103, JCOUNT \]
GO TO 3
END
SUBROUTINE THETA(e,xls,dxls,x2s,dx2s,x3s,
1   jdum,ss,sums)

SUBPROGRAM FOR CALCULATING CONCENTRATION by equating Fick's-law
and Faraday's-law fluxes

(X1-X2)/PN + (C1+X3)*EXCH = 0

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

DIMENSION E(201),CUR(201),A(200),B(200),TH(201)

COMMON/a/ CUR,TH,NZTl,T,PN,EXCH,AL,BE,GAM,TAF,A,B,C1,C2,EX

101 FORMAT (17HONOT CONVERGED AT,I4)

NZT = NZTl-1
DEVM = 0.000001d0
DZ = 1.0d0/NZT
S = TH(1)

DO 60 NZ = 2,NZT1
Z = (NZ - 1)*DZ
SUM = 0.0d0
IF (NZ .LE. 2) GO TO 42

C CALC. SUM(TH(J)*A(K))

DO 40 J-3,NZ
K = NZ - J + 1
40 SUM = SUM + TH(J-1)*A(K)
42 ETA = E(NZ)
NJ = NZ - 1

DO 56 N=1,20
X1 = TAF*S**(GAM - AL)*dEXP(AL*ETA)*dEXP(AL*T*(S - 1.0))
DX1 = X1*((GAM - AL)/S + AL*T)
X2 = S**(GAM + BE)*dEXP(-BE*ETA)*dEXP(BE*T*(1.0 - S))
DX2 = X2*((GAM + BE)/S - BE*T)
C3 = 1.50d0*C2*Z**EX/DZ**((1./3.)
DTH = S - ((X1-X2)/PN + (C1+X3)*EXCH)/((DX1-DX2)/PN - C3*EXCH)
X3 = C3*(TH( 1)*B(NJ) + SUM - dth)
CUR(NZ) = PN*(C1+X3)
if(.not.(jdum.eq.2 .or. jdum.eq.0) .and. nz.eq.4)) go to 44

sums = sum
ss = s
xls = xl
dxls = dxl
x2s = x2  
dx2s = dx2  
x3s = x3  
44  continue  
IF (dABS(S-DTH) - DEVM*dABS(DTH))  60,60,56  
56  S = DTH  

c  PRINT 101, NZ  
60  TH(NZ) = DTH  

c  RETURN  
END  

c  FUNCTION P(N,X)  
    implicit double precision (a-h,o-z)  
C  CALCULATION OF LEGENDRE POLYNOMIALS  
P1= 1.0d0  
P2= X  
IF (N-1)  1,2,3  
1  P= P1  
RETURN  
2  P= P2  
RETURN  
3  NM1= N - 1  
DO 4 NU=1,NM1  
P=(X*dFLOAT (2*NU+1)*P2-dFLOAT (NU)*P1)/dFLOAT (NU+1)  
P1= P2  
4  P2= P  
RETURN  
END  

subroutine ees(eo,e,b,j,denom,x1,dxl,x2,dx2,x3,ss,sums)  
    implicit double precision (a-h,o-z)  
    DIMENSION PM(21),B(21),C(201),CUR(201),E(201),PP(21,201),R(201),AA  
    1(200),BB(200),X(40),W(40),CUG(40),PG(21,40),eo(201)  
    COMMON/a/ CUR,C,LMAX,TPLUS,AN,EXCH,ALPHA,BETA,GAMMA,TAF,AA,BB,  
    1C1,C2,EX  
    common/b/rmax,pp,im,x,dz,r,pg,w,pm,v  
    CALL THETA(eo,x1,dxl,x2,dx2,x3,ss,sums)  
    DO 16 I=1,IM  
    LI= X(I)**3/DZ + 1
16  CUG(I) = CUR(LI) + (CUR(LI+1) - CUR(LI)) * (X(I)**2 - R(LI)**2) / (R(LI+1)**2 - R(LI)**2)

c
\[ v = e(1) \]

c
DO 15 N=1,NMAX
   B(N) = 0.0d0
   DO 14 I=1,IM
14   B(N) = B(N) + CUG(I) * X(I) * PG(N,I) * W(I)
    B(N) = 0.5 * B(N) * (4*N-3) / PM(N)
15  v = v + b(n) * pp(n,1)

c
   DO 18 I=2,1MAX
      phi = v
   DO 17 N=1,NMAX
17     phi = phi - b(n) * pp(n,1)
18    e(1) = phi

c
   return
end
Appendix F. Program CHANNEL

F.1. Program Description

Program CHANNEL solves for the current, concentration, and potential distribution in a channel flow cell that may have multiple reactions and interacting boundary layers. Superposition integrals are used to express the flux in terms of the surface concentrations, and the Butler-Volmer equation and Faraday's law give a second expression for the flux. With a guessed potential distribution, the surface concentrations are obtained from the flux equations by a multidimensional Newton-Raphson method. The iteration on the potentials is by successive substitution with damping. The damping factor is adjusted as the iterations proceed to attempt to speed the convergence without producing instabilities. This program does not converge well, but the restart feature is often helpful.

Input Variables:

NG = half the number of Gaussian quadrature points to be used in the integrals for the potential distribution (maximum allowable = 48)

ROOT(JXP) = roots for Gaussian quadrature (input the roots from 0 to 1)

GW(JXP) = Gaussian quadrature weights

XLAMDA(I) = eigenvalues from the asymmetric Graetz problem

AI(I) = coefficients from the asymmetric Graetz problem

NJ = number of evenly-spaced mesh points

NDEC = number of "decades" or regions with finely-spaced mesh points. Decades start from x=0 and are nested inside each other

ITSTRT = 0 if the run is not a restart, nonzero otherwise

IFINE : not used
NC: not used

IWHITE = 0 to read dimensionless parameters, nonzero to read dimensional parameters

for comparison with White's results\(^{18}\)

H = gap thickness (cm)

XL = electrode length (cm)

VCELL = applied cell voltage (V)

RHOO = solvent density (kg/cm\(^3\))

DR = diffusion coefficient of main reactant (cm\(^2\)/s)

CR = feed concentration of main reactant (mol/cm\(^3\))

XNM = \(n_m\) = number of electrons in main reaction

SRM = \(s_{Rm}\) = stoichiometric coefficient of main reactant in main reaction

IUNIFC(I) = nonzero if uniform current is guessed for electrode I

ILIMC(I) = nonzero if limiting current is guessed for electrode I

IPRIMC(I) = nonzero if primary current is guessed for electrode I

IZEROP = nonzero for no ohmic drop

HOL = aspect ratio, \(h/L\)

ICUT = specifies size of each "decade" of finely-spaced points. A decade starts from

\[ x = 0 \] and ends at \( x = 1/\text{ICUT} \)

\[ XN = N = \frac{n_m^2 F^2 D_R c_{R,ref}}{s_{RM} R \kappa_{feed} RT} \left( \frac{5vL^2}{hD_R} \right)^{1/3} = \text{dimensionless limiting current density}^\dagger \]

PEHOL = \( Pe h/L = \frac{2 <v> h^2}{DL} = \text{dimensionless parameter characterizing boundary} \)

layer thickness (directly proportional to Graetz number)

RHOO = \( \rho_o/c_{R,ref} = \text{solvent density / reference concentration of main reactant} \)

VCELL = \( V_{\text{cell}} \frac{-n_m F}{s_{RM} R T} = \text{dimensionless cell voltage} \)
SON = $\sigma_{R_m}/n_m$

ITMAXO = maximum number of iterations in outer loop

ISEC = nonzero if running secondary current distribution

NSPEC = number of species

NRXN = number of reactions

NRE = index for reference-electrode reaction

XIEND = arbitrary large value of the current at the endpoints for the primary current distribution

ONE(IELEC,J) = 1.d0 if reaction J occurs on electrode IELEC (zero otherwise)

AAJ = $\alpha_{a,j}$ = anodic transfer coefficient for reaction J

ACJ = $\alpha_{c,j}$ = cathodic transfer coefficient for reaction J

XJ(J) = exchange current density for reaction J (A/cm$^2$)

UTHJ = standard electrode potential for reaction J (V)

NSE = number of species in solution that do not participate in electrode reactions (used to calculate conductivity)

DSE(I) = diffusion coefficient of nonparticipating species I

CSE(I) = feed concentration of nonparticipating species I

ZSE(I) = charge number of nonparticipating species I

DI = diffusion coefficient of species I (a participating species)

CREFI = reference concentration of species I (mol/cm$^3$)

CREI = concentration of species I in reference-electrode compartment (mol/cm$^3$)

CI = feed concentration of species I (mol/cm$^3$)

ZI = charge number of species I

GAM(I,J) = exponent for concentration dependence of reaction-J exchange current den-

\[ \text{This interpretation holds only for thin boundary layers.} \]
RATSON(I,J) = \frac{s_{ij}}{n_j} = \text{ratio of stoichiometric coefficient of species I in reaction J to number of electrons in reaction J}

ASON(1,J) = \alpha_{s,J} s_{Rm} / n_m = \text{dimensionless anodic transfer coefficient for reaction J}

ASON(2,J) = \alpha_{c,J} s_{Rm} / n_m = \text{dimensionless cathodic transfer coefficient for reaction J}

XJ(J) = \mathcal{J}_j = -n_m FLi_0 /(s_{Rm}RT\kappa_\infty) = \text{dimensionless exchange current density for reaction J}

UJTH(J) = U_{Jf}(-n_mF)/(s_{Rm}RT) = \text{dimensionless standard electrode potential for reaction J}

UJTH(NRE) = \text{dimensionless standard electrode potential of the reference-electrode reaction}

DIODR(I) = D_i / D_R = \text{dimensionless diffusion coefficient of species I}

CREF(I) = \text{dimensionless reference concentration of species I}

CRE(I) = \text{dimensionless concentration of species I in the reference-electrode compartment}

CIOCR(I) = c_{i,feed} / c_{R,ref} = \text{dimensionless feed concentration of species I}

GAM(I,J) = \text{exponent for concentration dependence of reaction-J exchange current density on species I}

RATSON(I,J) = \frac{s_{ij}n_m}{s_{Rm}n_j} = \text{dimensionless ratio of stoichiometric coefficient of species I in reaction J to number of electrons in reaction J}

\text{Output Variables:}

\text{COEFF} = -N(16Z)^{-1/3}, \text{ where } Z = \frac{3h^2 \langle v \rangle}{8D_R L}, \text{ dimensionless parameter that multiplies superposition integrals}
XIAV = \frac{-n_m FL}{s_{Rm}RT_{\kappa_\infty}} = \text{dimensionless average current density}

PHISTR = \text{dimensionless integration constant in solution to Laplace's equation for the potential}

UREF(J) = \text{dimensionless theoretical open-circuit potential for reaction J, evaluated at the reference concentrations}

PORQ(IP, I, J) = p_{ij} = \text{anodic reaction order of species I in reaction J}

PORQ(IQ, I, J) = p_{ij} = \text{cathodic reaction order of species I in reaction J}

CSURF(IAN, I, JX) = \text{dimensionless anode surface concentration of species I at mesh point JX}

XIJ(IAN, J, JX) = \text{dimensionless anodic partial current density from reaction J at mesh point JX}

XI(IAN, JX) = i_{an}(-n_m/s_{Rm})FL/(RT_{\kappa_\infty}) = \text{dimensionless anodic current density}

FINE(IAN, IDEC, JX) = \text{anode current density at mesh point JX in decade IDEC (currents at the finely-spaced mesh points)}

CAN(IDEC, I, K) = \text{dimensionless anode surface concentration at mesh point K in decade IDEC}

CCATH(IDEC, I, K) = \text{dimensionless cathode surface concentration}

P0MPS(IAN, JX) = (\Phi^{\circ}_{an} - \Phi^{\circ})(-n_m F)/(s_{Rm}RT) = \text{potential calculated from Laplace's equation}

F.2. Major Subroutines

F.2.1. Subroutine INTCXI

This subroutine calculates XINT0 (I, N-K+1, IDEC), which is
\[
\frac{1}{\Delta z} \int_{(k-1)\Delta z}^{k\Delta z} C_i(n\Delta z - x') \, dx',
\]
where \( C \) is the flux calculated from the asymmetric Graetz problem.

**F.2.2. Subroutine EQFIVE**

Calculates the dimensionless form of the \((\Phi^0 - \Phi^*)\) given by equation 5 in Parrish and Newman\textsuperscript{21} (equation 1-6 in this work). The singularity in the equation was eliminated by a modified method of Kantorovich and Krylov.\textsuperscript{64} (See section 1.4.3.)

**F.2.3. Subroutine PHI**

Calculates the new potential distribution that is damped in the main program.

**F.2.4. Subroutine IAVLP**

Contains the iteration loop for \(<i>\). The average current used to calculate the potential distribution from Laplace's equation must agree with the average current resulting from the kinetics and mass transfer. The iteration is by a regula-falsi method with bisection or extrapolation, as needed.

**F.2.5. Subroutine PHILOOP**

Contains the iteration loop for \(\Phi^*\) (similar to subroutine IAVLP).

**F.2.6. Subroutine NEWT**

Calculates concentrations by solving the superposition-integral equation with the method of Acrivos and Chambre. Negative concentrations are reset to \(10^{-6}\) times the previous (positive) value. Subroutine FCNS calculates the functions needed for the matrix equation.

**F.2.7. Function DFDC**

Calculates Jacobian for Newton-Raphson on surface concentrations.
F.2.8. Subroutine NEWTLC

Special version of subroutine NEWT for limiting current. Calculates cathode current from the mass transfer instead of the kinetics. (Calls FCNSLC.)
implicit double precision (a-h,o-z)
logical goback,idump

dimension csurf(2,10,101),iunifc(2),ilimc(2),iprimc(2),
  xij(2,5,101),gam(10,5),svdist(2,101,5),old(2,101,5),
  fine(2,5,101),
  xi(2,101),ujth(5),ciocr(10),x(3),y(3),fit(3),ydum(3),
  p0mpsi(2,101,5),root(96),p0mpso(2,101,5),eta(2,101,5),
  tox(2,101),can(5,10,101),ccath(5,10,101),xitotj(2,5),
  toxj(2,101,5),etas(2,101)

common/evprob/xlamda(3),ai(3),z,diodr(10)
common/co/coeff,xn
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
  nspec,nrnx,one(2,5),rho0
common/matcom/b(20,20),d(20,41)
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/cdata/cnctol,ftol,xint0(10,101,5),
  xintl(10,101,5),n2,nd,iseC
common/elecs/v(2),ian,icath
common/geom/pi,hol,nj,ndec,icut
common/gauss/xp(96),gw(96),ngt,xielec(201),
  xiothr(201),njmore,xx(201)
common/nrdata/phicrt,totcrt,crit,vce11

data pi/3.14159265358979d+00/, ian/1/, icath/2/, ip/1/, iq/2/,
  ifor/1/, iback/2/

trace subprogram time

call readin(iunifc,ilimc,iprimc,iunifp,izerop,iprimp,
  xn,ujth,cioocr,gam,son,ng,root,itmapo,d2,d3,nre,
  irstrt,ifine,nc,xiend)

n2=2*nspec
nd=2*n2+1
coeff=xn*(16.*z)**(-1./3.)
njml=nj-1

Rearrange the Gaussian roots so they are in the interval 0 to 1
instead of -1 to 1.

ngt = 2*ng

do 111 jxp=1,ngt
111 xp(jxp)=(root(jxp)+1.)/2.
do 170 i=1,nspec
  do 171 jx=1,nj
    do 171 ielec=ian,icath
      csurf(ielec,i,jx)=ciocr(i)
      if(ciocr(i).eq.0.) csurf(ielec,i,jx)=1.d-15
    continue
  do 170 j=1,nrxn
    porq(ip,i,j)=gam(i,j)+ason(ian,j)*ratson(i,j)*son
    porq(iq,i,j)=gam(i,j)-ason(icath,j)*ratson(i,j)*son
    print 177,i,j,porq(ip,i,j),porq(iq,i,j)
    format(1x,' p(',i2,' ,',i2,')=',lpell.4,' q=',lpell.4)
  continue
  do 172 j=1,nrxn
    sum=0.d+00
    do 173 i=1,nspec
      if(cref(i) .lt. 1.d-15) go to 91
      sum=sum+ratson(i,j)*dlog(cref(i)/rho0)
    continue
    if(cre(i) .lt. 1.d-15) go to 173
    sum=sum-ratson(i,nre)*dlog(cre(i)/rho0)
  continue
  uref(j)=ujth(j)-ujth(nre)+sum
  continue

  calculate integrals of ln sinh**2 and ln cosh**2 that appear
  in the equation for the potential distribution after application
  of the method of Kantorovich and Krylov (L. V. Kantorovich and
  V. J. Krylov, "Approximate Methods of Higher Analysis,"

  call intcxi(nspec,xint0,xint1)

  temp.

  v(icath)=-6.167d+00
  v(ian)=v(icath)+vcell
  v(icath)=0.d+00
  v(ian)=vcell

  phistr=vcell*ason(ian,1)/(ason(ian,1)+ason(icath,1))+v(icath)

  xia=0.74058*xn
if(irstrt.ne.0) print 280
format(' this run is a restart')

if(irstrt.ne.0) go to 206
   call iguess(iunifc,ilimc,iprimc,iunifp,izerop,iprimp,
      xi,p0mpsi,xiend)
go to 207
206 call rsttrt(ifine,nc,xiav,phistr,xi,p0mpsi)
print 208,xiav,phistr
208 format(' after call rsttrt xiav=',lpell.4,' phistr=',lpell.4)
207 continue

c temp
c phistr=120.d+00
c
n=3
m=2*n+1
c

main iterative loop
c
c
198 x(3)=d3
   x(2)=d2
   x(1)=1.0d+00
   yold=0.d+00
   irep=3
print 198,damp
198 format(' damp=',lpell.4)
do 190 it=1,itmaxo
do 209 idec = l,ndec
do 209 ielec=ian,icath
do 209 jx=1,nj
209 old(ielec,jx,idec) = p0mpsi(ielec,jx,idec)
inrit=0
300 inrit = inrit + 1
print 191,it,inrit,phistr,xiav
CHANNEL.FOR

191  format(1,' outer iteration',i3,' sub iteration',i3,11', phistr=',lpell.4,1 xiav=',lpell.4)

phiin = phistr
xiavin = xiav

i3=0

do 238 idec = 1,ndec
do 238 ielec=i an,icath
do 238 jx=1,nj
238  p0mpso(ielec,jx,idec)=p0mpsi(ielec,jx,idec)
call phi(phiin,xiavin,xitotc,xitota,
1  p0mpsi,eta,csurf,xij,xi,phistr,xiav,1t,tx,1ne,f1ne,can,ccath,
1  xitotj,txoj,son,etas)
c
233  format(1 atfer call phi, phistr=',lpell.4)
c
233  print 103
233  format(1 converged concentrations',
c
1  1 /' anode cathode anode cathode')
do 105 jx=1,nj

c
233  print 204,csurf(i an,1,jx),csurf(icath,1,jx)
c
204  format(lx,lpell.4,1x,ell.4)
c
105  continue

c
214  print 114
214  format(1 converged currents',1 anode cathode')
do 205 jx=1,nj

c
214  print 214,xi(i an,jx),xi(icath,jx)
c
214  format(lx,lpell.4,1x,ell.4)
c
205  continue

c
232  format(1 relative delta phi criterion=',lpell.4)
c
if(inrit.ne.1) go to 236
do 235 idec=1,ndec
do 235 ielec=i an,icath
do 235 jx=1,nj
235  svdist(ielec,jx,idec) = p0mpsi(ielec,jx,idec)
smsq = O.d+00

do 210 ielec=i an,icath
    if(ielec.eq.ian) print 231,ielec
231  format(1, potential distr. on electrode',i2,1,
' jx     old     new ')
do 210 idec=1,ndec
do 210 jx=l,nj
    if(idec.eq.ndec) print 230,jx,pomps0(ielec,jx,ndec),
       poms0(ielec,jx,ndec)
230   format(lx,i2,3x,lepell.4,3x,ell.4)
   sumsq = sumsq + (poms0(ielec,jx,ndec)
   - poms0(ielec,jx,ndec))**2
   poms0(ielec,jx,idec)=poms0(ielec,jx,idec) + (1.-damp)*
   (poms0(ielec,jx,idec)-poms0(ielec,jx,idec))
   if(jx.eq.nl) print *,' elec',ielec,' sumsq=' ,sumsq
   if(jx.eq.nl .or. jx.eq.nj) go to 210
   if(it.ne.itmaxo) go to 210
   if(dabs((poms0(ielec,jx,ndec)-poms0(ielec,jx,ndec))/poms0(
      ielec,jx,ndec))
     .ge.phicrt) go to 210
   if(sumsq.lt.O.OOld+OO) go to 258
      print 259
259   format(' would have converged')
      go to 210
258   if(ielec.eq.icath .and. jx.eq.njml) go to 193
210  continue
   call phiint(poms0)
c
   print 211,damp,it,inrit,sumsq
211   format(' damp=",lepell.4,,' outer it=' ,i2,' sub it=",i2,/
   ' sum of squares=",lepell.4)
c
   if(inrit.eq.nl) go to 11
   if(inrit.eq.nl) go to 22
   if(inrit.eq.nl) go to 33
      y(irep)=sumsq
      go to 37
   if(y(2)=sumsq
     yold=y(2)
     irep=2
37    continue
    n = 3
    m = 2*n + 1
    do 35 i=1,n
35    fit(i)=1.d+00
    call abc(n,m,x,y,fit,idump)
   if(.not.idump) go to 40
      call dump(phistr,xiav,xi,poms0)
print 42  
format(' bombed in abc')  
stop  

40  
damp = -fit(2)/(2.*fit(1))  
print 198,damp  
if(inrit.gt.3 .and. dabs(y(irep)-yold)/y(irep)) .le.  
   1.0d-03) goback=.false.  
mid=1  
if( (x(3) - x(2))*x(1) - x(2)) .le. 0.d+00) mid=2  
if( (x(1) - x(3))*x(2) - x(3)) .le. 0.d+00) mid=3  
if(mid.eq.1 .and. y(1).lt.y(2) .and. y(1).lt.y(3))  
   goback=.false.  
if(mid.eq.2 .and. y(2).lt.y(1) .and. y(2).lt.y(3))  
   goback=.false.  
if(mid.eq.3 .and. y(3).lt.y(2) .and. y(3).lt.y(1))  
   goback=.false.  
if(x(mid).ge.1.0d+00) goback=.true.  
if(.not.goback) go to 36  
print 38,mid  
c  
38  
format(' mid=',i2)  
i3=0  
if(fit(1).le.0.d+00) go to 39  
irep=1  
if((dabs(y(2)) .gt. dabs(y(1))) irep=2  
   if((dabs(y(3)) .gt. dabs(y(irep))) irep=3  
   x(irep)=damp  
i3=1  
go to 36  
c  
c  
39  
n=2  
m=2*n+1  
c  
if(y(1).gt.y(2)) go to 200  
icomp=2  
y1=y(1)  
x1=x(1)  
go to 212  
200  
icomp=1  
y1=y(2)  
x1=x(2)  
212  
continue  
c  
if(y(3).gt.y(icomp)) go to 220  
y2=y(3)  
x2=x(3)  
go to 240
220  y2=y(icomp)
x2=x(icomp)
240  continue

c  if(x1.1e.x2) go to 101
   xt=x1
   yt=y1
   x1=x2
   y1=y2
   x2=xt
   y2=yt
101  if(y1.gt.y2) damp=2.*x2-x1
    if(y2.ge.y1) damp=2.*x1-x2
    y(1)=y1
    y(2)=y2
    x(1)=x1
    x(2)=x2
    print 198,damp
    irep=3
    x(3)=damp
    goback = .true.

c
36   x3=damp
    yold=y(irep)

c     do 430 idec=1,ndec
     do 430 ielec=ian,icath
     do 430 jx=1,nj
        p0mpsi(ielec,jx,idec) = old(ielec,jx,idec) + (1.-damp)*
1    (svdist(ielec,jx,idec)-old(ielec,jx,idec))
     continue
     damp=0.d+00
     go to 189

c
22   y(3)=sumsq
    damp=x(2)
    print 198,damp
    do 330 idec=1,ndec
    do 330 ielec=ian,icath
    do 330 jx=1,nj
330   p0mpsi(ielec,jx,idec) = old(ielec,jx,idec) + (1.-damp)*
1   (svdist(ielec,jx,idec)-old(ielec,jx,idec))
    damp=0.d+00
    goback = .true.
    go to 189

c
11   y(1)=sumsq
damp=0.d+00
goback = .true.
c
189 continue
c call phiint(p0mpsi)
c if(goback .and. inrit.lt.7) go to 300
c
ibest=1
if(x(1).ge.1.0d+00) ibest=2
if(y(2).lt.y(1) .and. x(2).lt.1.0d+00) ibest=2
if(y(3).lt.y(ibest) .and. x(3).lt.1.0d+00 .and. i3.eq.0) ibest=3
save=x(ibest)
c
x(1)=1.0d+00
x(2)=save
x(3)=x3
print 194,x(1),x(2),x(3)
194 format(’ x(1,2,3)=’,3el2.4)
190 continue
c
print 320,itmaxo
320 format(’ no conv in outer loop after’,i2,’ iterations’)
193 continue
c
c call prtout(coef,nj,ndec,icut,cref,porq,uref,nspec,nrxn,
c ciocr,xi,j,xi,csurf,p0mpsi,xiav,phistr,tox,fine,can,ccath,
c xitotj, toxj, etas)
c
stop
c
end
c
c subroutine phi(phiin,xiavin,xitotc,xitota,
c p0mpsi,eta,csurf,xij,xi,phistr,xiav,phistr,xiav,phistr,xiav,phistr,
c xitotj, toxj, son,etas)
c
implicit double precision (a-h,o-z)
dimension csurf(2,10,101),xintgd(101),
c xij(2,5,101),xi(2,101),p0mpsi(2,101,5),eta(2,101,5),
c xiint(2,96,5),tox(2,101),fine(2,5,101),can(5,10,101),
c ccath(5,10,101),xitotj(2,5),toxj(2,101,5),etas(2,101)
c
common/evprob/xlamda(3),ai(3),z,diodr(10)
c
common/co/coeff,xn
c
common/rxnpars cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
c nspec,nrxn,one(2,5),rho0
c
common/matcom/b(20,20),d(20,41)
common/kinpars/xj(5), ratson(10,5), ason(2,5), ifor, iback
common/cdata/cnctol, ftol, xint0(10,101,5),
1   xint1(10,101,5), n2, nd, isec
common/elecs/v(2), ian, icath
common/geom/pi, hol, nj, ndec, icut
common/gauss/xp(96), gw(96), ngt, xielec(201),
1   xiothr(201), njmore, xx(201)
common/nrdata/phicrt, totcrt, crit, vcell

c   njml = nj - 1

c
ccccccccccccccccccccccccccccc

c
i<avg> loop

c
ccccccccccccccccccccccccccc

c
   call iavlp (phiin, xiavin, xitotc, xitota,
1   p0mps, eta, csurf, xij, xi, phistr, xiav, it, tox, fine, can, ccath,
1   xitotj, toxj, son, etas)

c
c   print 251, phistr

c
251 format(' after call iavlp, phistr=' , lpell.4)

c
c
   print *, 'currents'
   njmore = 0
   xmax = -1.
   do 249 idec = 1, ndec
      h = dfloat(icut)**(idec - ndec)/dfloat(nj - 1)
   do 249 jx = 1, nj
      x = dfloat(jx - 1)*h
      if (x.gt.xmax) then
         xmax = x
         njmore = njmore + 1
         xx(njmore) = x
         xielec(njmore) = fine(1, idec, jx)/xiav
         xiothr(njmore) = fine(2, idec, jx)/xiav
      endif
   249 print 268, x,
1   fine(1, idec, jx), fine(2, idec, jx), fine(1, idec, jx) +
1   fine(2, idec, jx)
268 format(1pe11.4, lx, 2(1pe20.13, lx), lx, 1pe11.4)
   do 248 jx = 1, nj
      x = dfloat(jx - 1)*h
   248 print 247, x, tox(1, jx) + tox(2, jx)
247 format(2(1pe11.4, lx))
   print *, 'normalize i by ', xiav
   do 250 ielec = ian, icath


do 250 jx=1,nj
   xi(ielec,jx)=xi(ielec,jx)/xiav

calculate phi 0 - phi * (p0mps) from solution to Laplace's eqn.

do 200 jx=1,nj
   jreal = jx
   xol = dfloat(jx-1)/dfloat(nj-1)
   call eqfive(xol,phian,phicath,p2,p3,p5)
   print 181,xol,p2,p3,phicath,p5,phian
if(jreal.ge.1 .and. jreal.le.nj) p0mps(ian,jreal,ndec) = phian
if(jreal.ge.1 .and. jreal.le.nj) p0mps(icath,jreal,ndec) = phicath
   format(0pf8.4,x,l5e12.4)
200 continue

call phiint(p0mps)
if(ndec.eq.1) go to 202
   do 201 idec=1,ndec-1
   do 201 jx=1,nj
      xol = dfloat(jx-1)/dfloat(nj-1)/icut**(ndec-idec)
      if(xol.gt.0.02) then 
        call eqfive(xol,phian,phicath,p2,p3,p5)
        p0mps(ian,jreal,idec) = phian
        p0mps(icath,jreal,idec) = phicath
      end if
   continue
202 print *, 'p0mps'
   print 269,((dfloat(jx-1)/dfloat(nj-1),p0mps(1,jx,ndec),
   1 p0mps(2,jx,ndec)),jx=1,nj)
269 format(3(lpe11.4,lx))
   istop = 0
   if(istop.ne.0) stop
   return
end

subroutine abc(n2,nd,x,y,coeff,idump)

implicit double precision (a-h,o-z)
logical idump
common/matcom/b(20,20),d(20,41)
dimension x(3),y(3),coeff(3)

idump = .false.
   print 122,(x(i),i=1,n2)
122 format(' x(i) = ',3e12.4)
print 222,(y(i),i=1,n2)
print 222,format(' y(i)=' ,3e12.4)
print 223,(coeff(i),i=1,n2)
print 223,format(' coeff(i)=' ,3e12.4)
c print 224,n2,nd
c 224 format(' n2=', i2,' nd=', i2)
c
n2pl=n2+1
do 100 i=1,n2
do 110 j=1, n2
   b(i,j)=0.d+00
d(i,j) = 0.d+00
do 100 j=n2pl,nd
d(i,j)=0.d+00
do 120 j=1, n2
   b(i,j)=x(i)**(n2-j)
do 127 i=-l,n2
d(i,1)=0.d+00
do 128 j=1, n2
   d(i,1)=d(i,1)-coeff(j)*x(i)**(n2-j)
d(i,1)=d(i,1)+y(1)
do 124
do 123,b(i,j),j=1,n2
do 123,format(3e12.4)
c continue
c
print 136
print 136,format(' bij')
do 124 i=1,n2
do 123,(b(i,j),j=1,n2)
do 123,format(3e12.4)
c continue
c
c
c
if(determ.eq.0.d+00) print 121
format(' zero determinant')
if(determ.ne.0.d+00) go to 131
idump=.true.
return
c
131 continue
CHANNEL.FOR

```fortran
   do 117 i=1,n2
   coeff(i) = coeff(i) + d(i,1)
   c
   c   xx = -coeff(2)/(2.*coeff(1))
   c   yy = 0.d+00
   c   do 119 j=1,n2
   c 119   yy = yy + coeff(j)*xx**(n2-j)
   c
   print 125,y(1),y(2),y(3),yy,xx,coeff(1),coeff(2),coeff(3)
   c 125   format(’ y1,y2,y3=’,3e12.4,’ y=’,e12.4,’/’,’ x=’,ipell.4,’/’,
   c         ’ aa,bb,cc=’,3e12.4)
   c
   print 130
   c 130   format(’ coefficients’)
   c
   print 129,(coeff(i),i=1,n2)
   c 129   format(5e12.4)
   c
   return
   end
   subroutine iguess(iunifc,ilimc,iprimc,iunifp,izerop,iprimp,
   1   xi,p0mps,xiend)
   c
   implicit double precision (a-h,o-z)
   dimension iunifc(2),ilimc(2),iprimc(2),sign(2),xi(2,101),
   1   p0mps(2,101,5),xiint(2,96,5)
   c
   common/elecs/v(2),ian,icath
   common/geom/pi,hol,nj,ndec,icut
   common/gauss/xp(96),gw(96),ngt,xielec(201),
   1   xiothr(201),njmore,xx(201)
   c
   ndm1=ndec-1
   icertx=-0
   sign( ian)=1.d+00
   sign(icath)=-1.d+00
   c
   do 100 ielec=ian,icath
       if(iunifc(ielec).ne.0) go to 10
       11
       if(ilimc(ielec).ne.0) call ilim(ielec,xi,xiint)
       100   if(iprimc(ielec).ne.0) call primry(iprimc,ielec,xi,x,icertx,
       1   xix,xiint,xiend)
   c
   if(iprimp.ne.0) go to 20
   12
   if(iunifp.ne.0) go to 30
   13
   if(izerop.ne.0) go to 40
   14
   if(iprimp.ne.0 .or. iunifp.ne.0 .or. izerop.ne.0) return
```
```plaintext
CHANNEL.FOR

! 

! 

! 

do 200 jx=1,nj
jreal = jx
xol = dfloat(jx-1)/dfloat(nj-1)
call eqfive(xol,phian,phicath,p2,p3,p5)
!
print 181,xol,p2,p3,phicath,p5,phian

if(jreal.ge.1 .and. jreal.le.nj) p0mps(ian,jreal,ndec) = phian
if(jreal.ge.1 .and. jreal.le.nj) p0mps(icath,jreal,ndec)

181 format(0pf8.4,x,lp5el2.4)
200 continue
!
call phiint(p0mps)
creturn
!
uniform current
!
do 120 jx=1,nj

120 xi(ielec,jx)=sign(ielec)
if(iunifp.ne.0) go to 11
do 121 idec=1,ndec
do 121 jg=1,ngt

121 xiint(ielec,jg,idec)=sign(ielec)
go to 11
!
cprimary distribution
!
do 130 ielec=ian,icath
do 130 jx=1,nj
p0mps(ielec,jx,ndec)=sign(ielec)*hol/pi
if(hol.gt.1.0d+00) p0mps(ielec,jx,ndec)=sign(ielec)/pi*
1 (1.+dlog(hol))
if(ndec.eq.1) go to 130
do 131 idec=1,ndml
131 p0mps(ielec,jx,idec)=p0mps(ielec,jx,ndec)
130 continue
go to 12
!
cpotential distribution for uniform current distribution
!
do 140 ielec=ian,icath
do 140 jx=1,nj
p0mps(ielec,jx,ndec)=sign(ielec)*hol/pi
if(hol.gt.1.0d+00) p0mps(ielec,jx,ndec)=sign(ielec)/pi*
1 (1.+dlog(hol))
if(ndec.eq.1) go to 140
do 141 idec=1,ndml
```
CHANNEL.FOR

141  pOmps(ielec,jx,idec)=pOmps(ielec,jx,ndec)
140  continue
    call phiint(pOmps)
    go to 13

c
141  no ohmic drop
c

40  do 150 ielec=ian,icath
    do 150 jx=l,nj
    do 150 idec=l,ndec
150  pOmps(ielec,jx,idec)=0.d+00
    go to 14

c
    end
    c
    c
    subroutine ilim(ielec,xi,xiint)
    c
    implicit double precision (a-h,o-z)
    dimension xi(2,10l),xiint(2,96,5),sign(2)
    c
    common/geom/pi,hol,nj,ndec,icut
    common/gauss/xp(96),gw(96),ngt,xielec(201),
        xiothr(201),njmore,xx(201)
    c
    njml=nj-1
    denom=dfloat(njml)
    sign(1)=1.d+00
    sign(2)=-1.d+00
    c
    xi(ielec,1)=sign(ielec)*1.0e+20
    c
    do 100 jx=2,nj
        x=dfloat(jx-1)/denom
        xi(ielec,jx)=sign(ielec)*2./3.*x**(-1./3.)
    100  continue
    hold=0.d+00
    c
    do 141 idec=1,ndec
        h=dfloat(icut)**(idec-ndec)/denom
        do 149 jg=-1,ngt
            x=xp(jg)*(njml*h - hold) + hold
            xiint(ielec,jg,idec)=sign(ielec)*2./3.*x**(-1./3.)
    149  continue
        hold=h*njml
    141  continue
return
end

function iother(i)
  ! this function is used for generating the index of the 'other'
electrode
  iother=3-i
  return
end

subroutine totcj(jbegin,h,fj,savej,toxj,j)
  implicit double precision (a-h,o-z)
dimension save(2),fj(2,5,101),xintgd(101),tox(2,101),savej(2,5),
          toxj(2,101,5)
  common/geom/pi,hol,nj,ndec,icut
  common/elecs/v(2),ian,icath
  ! this subroutine integrates the function fj (call is inside
  idec loop and j loop)
  do 95 ielec=ian,icath
      save(ielec) = savej(ielec,j)
      print 94,ielec,j,savej(ielec,j)
      94 format(' elec',i2,' j=',i2,' savej input=',1pe11.4)
      do 363 jx=1,nj
          tox(ielec,jx) = toxj(ie1ec,jx,j)
          kend=(nj-1)/icut + 1
          jb=jbegin
          npts=nj-1
          if(jbegin.eq.2) go to 367
          jb=jbegin-1
          npts=nj-kend+1
          367 continue
          do 463 ielec=ian,icath
              do 363 jj=jb,nj
                  print 93,j,ielec,jj,fj(ielec,j,jx)
                  93 format(' rxn',i2,' elec',i2,' jj=',i2,' xintgd=',lpell.4)
                  xintgd(jj)=fj(ielec,j,jx)
              print *," tox='",tox(ielec,j)
          363 continue
      end
  95 continue
end
CHANNEL.FOR

call simps(xintgd,npts,h,save(ielec),ielec,jb, tox)
c  print *, ' save=', save(ielec)
continue
c
do 92 ielec=ian,icath
   savej(ielec,j)=save(ielec)
do 92 jx=1,nj
92 toxj(ielec,jx,j)=tox(ielec,jx)
if(jbegin.ne.2) return
c
do 100 ielec=ian,icath
tox(ielec,1) = 0.d+00
print 465
c 465 format( ' in totcurr initializing tox(1)' )
gl=fj(ielec,j,1)
g2=fj(ielec,j,2)
if(gl.eq.0.d+00 .or. g2.eq.0.d+00) go to 373
if(gl.ne.g2 .and. gl*g2/dabs(gl*g2).gt.0.d+00) go to 372
372 save(ielec)=save(ielec) + h/2.*(gl+g2)
tox(ielec,2) = h/2.*(gl+g2)
do 98 jx=3,nj
98 tox(ielec,jx)=tox(ielec,jx) + h/2.*(gl+g2)
go to 99
372 continue
save(ielec)=save(ielec) + 1.5*h*g2
tox(ielec,2) = 1.5*h*g2
do 97 jx=3,nj
97 tox(ielec,jx)=tox(ielec,jx) + 1.5*h*g2
99 continue
c print 466
c 466 format( ' in totcurr initializing tox(2)' )
100 continue
c
do 96 ielec=ian,icath
   savej(ielec,j) = save(ielec)
do 96 jx=1,nj
96 toxj(ielec,jx,j) = tox(ielec,jx)
c
c return
c end
c
c subroutine readin(iunifc,ilimc,iprimc,iunifp,izerop,iprmp,
1 xn,ujth,ciocr,gam,son,ng,root,itmaxo,d2,d3,
1 nre,irstrt,ifine,nc,xiend)
c
this subroutine reads and prints the input data for program channel
input data are in file
    channel.data

The parameter iwhite tells whether input data are dimensional
or dimensionless (iwhite=0 means dimensionless)

I. The input and output dimensionless parameters are:

1) Dimensionless parameters describing the problem:

   \[ xn = \frac{N - n_m^{2/3} s R_m^{2/3} F^2 D^{1/3} c_{R,b}}{RT k_b} \times (6v L^{2/3} (h D^{2/3}))^{2/3} \] - dimensionless
   limiting current density

   \[ \rho_{O} = \frac{\rho_{O}/c_{R,b}}{c_{R,b}} \] - solvent density / bulk concentration
   of main reactant

   \[ u_{jth}(j) = U_{j}[\theta] \times \frac{-n_m F}{(s R_m RT)} \] - dimensionless
   standard electrode potential for reaction j

   \[ c_{i,cr}(i) = \frac{c_{i,b}}{c_{R,b}} \] - dimensionless bulk concentration
   of species i

   \[ \gamma(i,j) \] - exponent in composition dependence of exchange
   current density

   \[ z, \text{diodr} \]

   \[ \text{hol} \]

   \[ \text{cre, cre, nspec, nrxn, one} \]

   \[ xj, \text{ratson, ason} \]

   \[ vcell \]

2) Variables related to the solution method

   \[ ng = 1/2 \] the number of Gaussian quadrature points to be
   used in the integrals for the potential distribution
   (maximum allowable=48)

   \[ \text{root}(jxp) = \text{roots for Gaussian quadrature (range from -1 to 1)} \]

   \[ \text{isec} = 0 \] if tertiary current distribution, nonzero if
   secondary

   \[ \text{itmaxo} = \text{maximum number of iterations allowed in outer loop} \]

   \[ \text{nre} = \text{which reaction is reference electrode reaction} \]

   \[ \text{xlambda, ai} \]

   \[ nj, \text{ndec} \]
The dimensional parameters (if ihwite.ne. 0) are:

- \( h \) = gap thickness (cm)
- \( x_1 \) = electrode length (cm)
- \( \text{vcell} \) = applied cell voltage (V)
- \( \rho_0 \) = solvent density (kg/cc)
- \( \text{vel} \) = average linear velocity through channel (cm/s)
- \( d_r \) = diffusion coefficient of main reactant (cm\(^2\)/s)
- \( c_r \) = feed concentration of main reactant (mol/cc)
- \( x_{m} = n_{m} \) = no. of electrons in main rxn
- \( s_r \) = stoichiometric coeff. of main reactant in main rxn
- \( a_{aj} \) = anodic transfer coefficient for reaction \( j \)
- \( a_{cj} \) = cathodic transfer coefficient for reaction \( j \)
- \( x_{j}(j) \) = exchange current density for reaction \( j \) (A/cm\(^2\))
- \( \text{uth}_{j} \) = standard electrode potential for reaction \( j \) (V)
- \( n_{se} \) = number of species in solution that do not participate in electrolyte reactions (for ex. supporting electrolyte species)
- \( d_{se}(i) \) = diffusion coefficient of nonparticipating species \( i \)
- \( c_{se}(i) \) = feed concentration of nonparticipating species \( i \)
- \( z_{se}(i) \) = charge number of nonparticipating species \( i \)
- \( d_{i} \) = diffusion coefficient of species \( i \)
- \( c_{ref}(i) \) = reference concentration of species \( i \) (mol/cc)
- \( c_{ref}(i) \) = concentration of species \( i \) in reference electrode compartment (mol/cc)
- \( c_{i} \) = feed concentration of species \( i \) (mol/cc)
- \( z_{i} \) = charge number of species \( i \)
- \( \text{gam}(i,j) \) = exponent for concentration dependence of rxn \( j \)
- \( \text{ratson}(i,j) \) = ratio of \( s_{i,j}/n_{j} \) (stoichiometric coeff. of species \( i \) in rxn \( j \) to number of electrons in rxn \( j \))
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,
1   iback
common/cdata/cnctol,ftol,xint0(10,101,5),
1   xint1(10,101,5),n2,nd,i,sec
common/geom/pi,hol,nj,ndec,icut
common/gauss/xp(96),gw(96),ngt,xielec(201),
1   xiothr(201),njmore,xx(201)
common/nrdata/phi,ctotct,crit,vcell
c
read *,ng
if(ng.gt.48 .or. ng.lt.1) print 600,ng
600  format(ng='',i2,' is incorrect input')
c
Read the Gaussian quadrature roots and weights for the interval
0 to 1. Then put in the roots and weights for the interval
-1 to 0.
c
do 101 k=-1,ng
   karg=ng+1-k
   kpng=k+ng
   read *,(root(karg),gw(karg))
   root(kpng)=root(karg)
   gw(kpng)=gw(karg)
101  root(karg)--root(karg)

c
read *,(xlamda(i),i=1,3)
read *,(ai(i),i=1,3)
read 670,ichar
670  read *,nj,ndec,irstrt,ifine,nc
    print 617,irstrt
   617  format(' restart=',i3)
   if(nj.gt.101 .or. nj.lt.1)
      print 615,nj
615  format(nj='',i3,' is incorrect input')
   if(ndec.gt.5 .or. ndec.lt.1) print 616,ndec
616  format(ndec='',i3,' is incorrect input')
c
read 670,ichar
read *,iwhite
if(iwhite.ne.0) print 619
619  format(' Input Ralph White dimensional parameters')
c
   if(iwhite.ne.0) go to 300
go to 299
300  read 670,ichar
read *,h,xl,vcell,rho,vel
read 670,ichar
read *, dr, cr, xnm, srm
print 621, h, xl, vcell, rho, vel, dr, cr
621 format(' h=', lpe1l.4, ' l=', lpe1l.4, ' vcell=', lpe1l.4, ' rho=',
1 lpe1l.4, ' vel=', lpe1l.4, '/' ' dr=', lpe1l.4, ' cr=', lpe1l.4)

hol=h/xl
z=3.*h**2*vel/(8.*dr*xl)
rho0=rho/(1000.*cr)
vcell=-vcell*xnm/(srm*0.0257)
go to 301

c
299 read 670, ichtar
read *, (iunifc(i), i=1, 2), (ilimc(i), i=1, 2), (iprimc(i), i=1, 2)

do 516 i=1,2
   if(iunifc(i).ne.0) print 514, i
514 format(' uniform current on electrode', i2)
516 continue

do 513 i=1,2
   if(ilimc(i).ne.0) print 512, i
512 format(' limiting current on electrode', i2)
513 continue

do 511 i=1,2
   if(iprimc(i).ne.0) print 509, i
509 format(' primary current on electrode', i2)
511 continue

c
read 670, ichtar
read *, izerop
iunifp=0
iprimp=0
if(iunifc(1).ne.0 .and. iunifc(2).ne.0) iunifp=1
if(iprimc(1).ne.0 .and. iprimc(2).ne.0) iprimp=1
if(izerop.ne.0) print 508
508 format(' no ohmic drop')

c
read 670, ichtar
read *, hol, icut, xn, pehol, rho0, vcell, son
print 620, nj, ndec, hol, icut, xn, pehol, rho0, vcell
620 format(' nj=', i3, ' ndec=', i3, ' hol=', lpe1l.4, ' icut=', i3,
1 ' xn=', lpe1l.4, ' pehol=', lpe1l.4,
1 ' rho0=', lpe1l.4, ' v(cell)=' ,1pe1l.4)

c
z=3./16. * pehol

c
301 read 670, ichtar
read *, isec, itmaxo, nspec, nrxn, nre, xiend
read 670, ichar
read *, cnctol, ftol, crit, totcrt, phicrt, d2, d3

if (isec.ne.0) print 629
format( ' secondary current distribution')
print 628, itmaxo, nspec, nrxn, nre, cnctol, ftol, crit, totcrt,
     phicrt, d2, d3
628 format( ' max iterations=', i3, /
     i3, ' species', i3, ' reactions, reference rxn is rxn', i3, /
     ' conc tolerance=', lpe1l.4, ' conc eqn tolerance=', lpe1l.4, /
     ' conv crit for i avg=', lpe1l.4,
     ' conv crit for phi star loop=',
     lpe1l.4, '/', ' conv crit for potl distribution=', lpe1l.4,
     '/', d2, 'lpe1l.4, d3=', lpe1l.4)

if (nspec.gt.10 .or. nspec.lt.1) print 625, nspec
format(' nspec=', i3, ' is incorrect input')
if (nrxn.gt.5 .or. nrxn.lt.1) print 626, nrxn
format(' nrxn=', i3, ' is incorrect input')
if (nre.gt.5 .or. nre.lt.1) print 627, nre
format(' nre=', i3, ' is incorrect input')

do 162 ielec=1,2
read 670, ichar
read *, (one(ielec,j), j=1,5)
print 631, (one(ielec,j), j=1,nrxn)
631 format(' one=', 5f2.0)
162 continue

if (iwhite.eq.0) go to 302

do 171 j=1,nrxn
read 670, ichar
read *, aaj, acj, xj(j), uthj
print 632, aaj, acj, xj(j), uthj
632 format(' aaj, acj=', lpe1l.4, 'x, lpe1l.4, xj=', lpe1l.4, ' uth=',
     lpe1l.4)
ason(1,j)=-aaj*srnm/xnm
ason(2,j)=-acj*srnm/xnm
uthj(j)=-uthj*xnm/(srnm*0.0257)
171 continue

if (nre.eq.nrxn) go to 502
read 670, ichar
read *, uthj
uthj(nre)=-uthj*xnm/(srnm*0.0257)
502  read 670,ichar
     read *,nse
     do 181 i=1,nse
         read 670,ichar
         read *,dse(i),cse(i),zse(i)
     181  continue
     c
     xk=0.d+00
     do 172 i=1,nspec
         read 670,ichar
         read *,di,crefi,crei,ci,zi
         print 636,di,crefi,crei,ci,zi
     636     format(' di=',lpe1.4,' cref=',lpe1.4,' cre=',lpe1.4,' ci=',
             1     lpe1.4,' zi=',lpe1.4)
         diodr(i)=di/dr
         cref(i)=crefi/cr
         cre(i)=crei/cr
         ciocr(i)=ci/cr
     172  xk=xk+zi**2*di*ci
     do 182 i=1,nse
     182  xk=xk+zse(i)**2*dse(i)*cse(i)
     xk=xk*96487./(0.0257)
     print 637,xk
     637  format(' xk=',lpe1.4)
     xn=(xnm/srm)**2*96487.*dr*cr/(0.0257*xk)*(6.*vel*x1**2/(h*dr))**l/3.
     do 173 j=1,nrxn
     173  xj(j)=xj(j)*(-xnm/srm)*x1/(0.0257*xk)
     do 174 i=1,nspec
         read 670,ichar
     174  read *,(gam(i,j),j=1,nrxn)
     do 176 i=1,nspec
         read 670,ichar
         read *,(ratson(i,j),j=1,nrxn)
     176  continue
     c
     do 175 j=1,nrxn
         print 641,ratson(i,j),i,j
         ratson(i,j)=ratson(i,j)/(srm/xnm)
         if(nre.gt.nrxn) go to 175
         print 642,gam(i,j),i,j
     642     format(' pij=',lpe1.4,' i=',i2,' j=',i2)
         gam(i,j)=gam(i,j)+ason(1,j)*ratson(i,j)
     175  continue
     176  continue
     c
CHANNEL.FOR

C
print 620,nj,ndec,hol,icut,xn,z,rho0,vcell
do 177 j=1,nrxn
177 print 630,ason(1,j),ason(2,j),xj(j),ujth(j)
do 178 i=1,nspec
178 print 635,diodr(i),cref(i),cre(i),ciocr(i)
do 179 j=1,nrxn
179 continue
C
go to 304
302 continue
C
do 163 j=1,nrxn
read 670,ichar
read *,ason(1,j),ason(2,j),xj(j),ujth(j)
print 630,ason(1,j),ason(2,j),xj(j),ujth(j)
630 format('ason(1)='lpe11.4,'ason(2)='lpe11.4,'xj='lpe11.4,'ujth='lpe11.4)
do 164 i=1,nspec
read 670,ichar
read *,diodr(i),cref(i),cre(i),ciocr(i)
print 635,diodr(i),cref(i),cre(i),ciocr(i)
635 format('diodr='lpe11.4,'cref='lpe11.4,'cre='lpe11.4,'ciocr='lpe11.4)
do 166 i=1,nspec
read 670,ichar
read *(gam(i,j),j=1,nrxn)
if(cref(i).lt.1.0d-15 .and. gam(i,j).ne.0.) print *,’ incorrect input
for gam(i,j)’
do 199 j=1,nrxn
print 640,gam(i,j),i,j
640 format(’gam='lpe11.4,’i=’,i2,’j=’,i2)
do 199 continue
166 continue
do 168 i=1,nspec
   read 670,ichar
   read *,(ratson(i,j),j=1,nrxn)
   do 201 j=1,nrxn
      print 641,ratson(i,j),i,j
   641 format(' ratson=',1pel1.4,' i=',i2,' j=',i2)
   201 continue
   168 continue

end

subroutine prtout(coeff,nj,ndec,icut,cref,porq,uref,nspec,nrxn,
   ciocr,xij,xi,csurf,p0mps,xiav,phistr,tox,fine,can,ccath,
   xtotj,toxj,etas)

This subroutine prints the final results of program channel
implicit double precision (a-h,o-z)
dimension csurf(2,10,101),ciocr(10),xij(2,5,101),
   uref(5),porq(2,10,5),tox(2,101),fine(2,5,101),
   xi(2,101),cref(10),p0mps(2,101,5),can(5,10,101),xitotj(2,5),
   toxj(2,101,5),ccath(5,10,101),etas(2,10)

   print 139,coeff,xiav,phistr
   139 format(/,' coeff=','1pel1.4,' xiav=','1pel1.4,' phistr=','1pel1.4)
   print 138,(xitotj(1,j),j=1,nrxn)
   138 format(' xitotj, anode',2x,1pel1.4,4(lx,ell.4))
   print 137,(xitotj(2,j),j=1,nrxn)
   137 format(' xitotj, cathode',1pel1.4,4(lx,ell.4))
   do 145 j=1,nrxn
      print 146,j,uref(j)
   146 format(' j=','i2,' ujref=','1pel1.4)
   145 continue
   do 144 i=1,nspec
      print 143,i,(j,porq(1,i,j),j=1,nrxn)
   143 format(' i=','i2,5(' j=','i2,' pij=','1pel1.4))
   print 142,i,(j,porq(2,i,j),j=1,nrxn)
   142 format(' i=','i2,5(' j=','i2,' qij=','1pel1.4))
   144 continue
   do 149 i=1,nspec
      print 147,i
   147 format(/,' species','i3,/',' x ccath can')
do 149 jx=1,nj
  x=dfs(jx-1)/dfs(nj-1)
  print 148,x,csurf(2,i,jx),csurf(1,i,jx)
148 format(1x,lp9.2,1x,lpell.4,2x,lpell.4)
149 continue

do 150 j=1,nrxn
  print 151,j
151 format(/,' reaction',i2,/
    x xicj xiaj ')

do 150 jx=1,nj
  x=dfs(jx-1)/dfs(nj-1)
  print 121,x,xij(2,j,jx),xij(1,j,jx)
121 format(1x,lp9.2,1x,1p11.4,2x,1p11.4)
150 continue

c
print 170

c 170 format(/,' total currents',/
  x ian icath ')

do 187 jx=1,nj
  x=dfs(jx-1)/dfs(nj-1)
  print 188,x,xi(1,jx),xi(2,jx)
188 format(1x,1p9.2,1x,1p11.4,2x,1p11.4)
187 continue

c
print 191
191 format(/,' axial currents',/ x iaxial ')
do 192 idec=1,ndec
  h=dfs(icut)**(idec-ndec)/dfs(nj-1)
do 192 jx=1,nj
  x=h*dfs(jx-1)
  print 193,x,fine(1,idec,jx)+fine(2,idec,jx),fine(1,idec,jx),
    fine(2,idec,jx)
193 format(1x,1p9.2,1x,1p11.4,4(1x,e11.4))
192 continue

c
m=(nj+1)/2

print 180,m,xi(1,1)/xi(1,m)
180 format(' midpt=',i2,' i(1)/i(m)=',1p11.4)

do 16 idec=1,ndec
  h=dfs(icut)**(idec-ndec)/dfs(nj-1)
  print 479,idec
479 format(' decade=',i2,/' x c anode(x) ')
do 16 k=1,nj
  x=h*dfs(k-1)
  print 480,x,(can(idec,i,k),i=1,nspec)
480 format(1x,1p9.2,1x,1p11.4,4(1x,e11.4))
16   continue
   do 17 idec=1,ndec
       h=dfloat(icut)**(idec-ndec)/dfloat(nj-1)
       print 481,idec
       format(' decade',i2,' x c cathode(x)')
   do 17 k=1,nj
       x=h*dfloat(k-1)
       print 480,x,(ccath(idec,i,k),i=1,nspec)
   continue

write (7,184),xiav,phistr
184 format(1x,1pe9.2,1x,1pe9.2,1x,1pe9.2,1x,1pe9.2)
168 format(/,
1     ' x ian/iavg icath/iavg p0mps p0mpsc
1     etasa etasc')
   do 152 jx=1,nj
       x=dfloat(jx-1)/dfloat(nj-1)
       print 123,x,xi(1,jx),xi(2,jx),p0mps(1,jx,ndec),
       p0mps(2,jx,ndec),etas(1,jx),etas(2,jx)
       write (7,122),x,xi(1,jx),xi(2,jx),p0mps(1,jx,ndec),
       p0mps(2,jx,ndec)
122 format(1x,1pe9.2,1x,1pe9.2,1x,1pe9.2,1x,1pe9.2)
123 format(1x,1pe9.2,6(1x,1pe9.2))
152 continue

return
end

subroutine rstrt(ifine,nc,xiav,phistr,xiold,p0mps)

implicit double precision (a-h,o-z)
dimension xi(2,101),p0mps(2,101,5),coarse(2,101),xiold(2,101),
1     p0mpso(2,101)
common/geom/pi,hol,nj,ndec,icut
common/elecs/v(2),ian,icath

nf=nj
nj=nf
if(ifine.eq.0) go to 102
nj=nc
102 continue
print 103
103 format(' restart file')
read (4,*,xiav,phistr
CHANNEL.FOR

100  format(e11.4,2x,e11.4)
    print 100,xiax,phistr

c    do 105 jx=1,nj
       read (4,*)x,xi(1,jx),xi(2,jx),p0mps(1,jx,ndec),p0mps(2,jx,ndec)
       print 106,x,xi(1,jx),xi(2,jx),p0mps(1,jx,ndec),
           p0mps(2,jx,ndec)
106  format(1x,1pe9.2,1x,1pel1.4,2x,1pel1.4,2x,1pel1.4,2x,1pel1.4)
    do 104 ielec=ian,icath
       p0mpso(ielec,jx)=p0mps(ielec,jx,ndec)
    xiold(ielec,jx)=xi(ielec,jx)
    continue

c    if(ifine.eq.0) go to 107
    print 108
108  format(‘ going to finer points’)

c    do 120 ielec=ian,icath
       do 120 jx=1,nc
120   coarse(ielec,jx)=xi(ielec,jx)

c    call finer(coarse,nc,xi,nf)

c    do 130 ielec=ian,icath
       do 130 jx=1,nc
130   coarse(ielec,jx)=p0mpso(ielec,jx)

c    call finer(coarse,nc,p0mpso,nf)

c    do 140 ielec=ian,icath
       do 140 jx=1,nf
          p0mps(ielec,jx,ndec)=p0mpso(ielec,jx)
140   xiold(ielec,jx)=xi(ielec,jx)

c    107  call phiint(p0mps)
    nj=nf
    return
end

c subroutine finer(coarse,nc,fine,nf)

c implicit double precision (a-h,o-z)
dimension coarse(2,101),fine(2,101),xintgd(101)

c print 49,nf,nc
49  format(‘ in finer, nf=’,i2,’ nc=’,i2)
c
  do 50 ielec=1,2
  do 50 jx=1,nc
    print 51,ielec,jx,coarse(ielec,jx)
  51 format(' elec',i2,' jx=',i2,' coarse=',lpe1.4)
  50 continue

  x1 = 0.d+00
  xn = 1.d+00
  h = 1./dfloat(nc-1)
  m = 3

  do 100 ielec=1,2
    do 105 jx=1,nc
      xintgd(jx)=coarse(ielec,jx)
  105 x = dfloat(jx-1)/dfloat(nc-1)
    call interp1(xl,xn,h,nc,xintgd,x,m,yout)
    fine(ielec,jx)=yout
    print 101,ielec,jx,x,fine(ielec,jx)
  101 format(' elec',i2,' jx=',i2,' x=',lpe1.4,' fine=',lpe1.4)
  100 continue

  return
end

subroutine phiint(p0mps)

implicit double precision (a-h,o-z)
dimension xintgd(101),p0mps(2,101,5)

common/geom/pi,hol,nj,ndec,icut
common/elecs/v(2),ian,icath

if(ndec.eq.1) go to 290
  ndml=ndec-1
  njml=nj-1
  hc=1./dfloat(njml)

use Lagrange interpolation to
find p0mps at the Gaussian quadrature points

  do 312 ielec=ian,icath
    do 311 jx=1,nj
      xintgd(jx)=p0mps(ielec,jx,ndec)
  311 continue
  do 313 idec=1,ndml
decend=dfloat(icut)**(idec-ndec)
h=decend/dfloat(njml)
do 313 jx=1,nj
   x=dfloat(jx-1)*h
call interpl(0.d+00,1.d+00,hc,nj,xintgd,x,3,
   p0mps(ielec,jx,idec))
   continue
  312 continue
  290 continue
return
end

subroutine dump(phistr,xiav,xi,p0mps)
implicit double precision (a-h,o-z)
dimension xi(2,101),p0mps(2,101,5)
common/geom/pi,hol,nj,ndec,icut
print 139,xiav,phistr
139 format(' xiav=',1pell.4,' phistr=',1pe11.4)
p:print 168
168 format(/,
  '   x ian/iavg icath/iavg p0mpsa p0mpsc')
do 152 jx=1,nj
   x=dfloat(jx-1)/dfloat(nj-1)
   print 122,x,xi(1,jx),xi(2,jx),p0mps(1,jx,ndec),
   p0mps(2,jx,ndec)
122 format(1x,1pe9.2,lx,1pel1.4,2x,1pe11.4,2x,lpell.4,2x,lpell.4)
152 continue
return
end

subroutine iavlp (phiin,xiavin,xitotc,xitota,
  p0mps,eta,csurf,xij,xi,phistr,xiav,it,tox,fine,can,ccath,
  xitotj, toxj, son,etas)
implicit double precision (a-h,o-z)
logical secset,yloset,yhiset
dimension csurf(2,10,101),tox(2,101),fine(2,5,101),
  xij(2,5,101),xi(2,101),p0mps(2,101,5),eta(2,101,5),
  can(5,10,101),ccath(5,10,101),xitotj(2,5),toxj(2,101,5),
1  etas(2,101)

common/evprob/xlamda(3),ai(3),z,diodr(10)
common/co/coeff,xn
common/rxn pars cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
1  nspec,nrxn,one(2,5),rho0
common/matcom/b(20,20),d(20,41)
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/cdata/cnctol,ftol,xint0(10,101,5),
1  xintl(10,101,5),n2,nd,ise c
common/elecs/v(2),ian,icath
common/geom/pi,hol,nj,ndec,icut
common/gauss/xp(96),gw(96),ngt,xielec(201),
1  xiothr(201),njmore,xx(201)
common/nrdata/phicrt,totcrt,crit,vcell

c 105  format(/,' iavg iteration',i3,' x=' ,lpe23.16,' y=' ,lpe23.16)
1 106  format(' xofyhi=' ,lpe23.16,' xofylo=' ,lpe23.16,' yhi=' ,lpe11.4,
1 1  ylo=' ,lpe11.4)
1 114  format(' x=' ,lpe11.4,
1 1  y=' ,lpe11.4,' xbest=' ,lpe11.4,' ybest=' ,lpe11.4)
1 115  format(' iavlp',/,' x=' ,lpe11.4,
1 1  y=' ,lpe11.4,' xsec=' ,lpe11.4,' ysec=' ,lpe11.4,' xbest=' ,
1 1  lpe11.4,' ybest=' ,lpe11.4)
1 215  format(' beware y worse than ylo and yhi')
1 335  format(' iavlp no convergence in',i3,' iterations',/,
1 1  x=' ,lpe11.4,' y=' ,lpe11.4)
1 405  format(/,' converged in',i3,' iterations',/,' x=' ,lpe11.4,' y=',
1 1  lpe11.4)

c  ycrit = 1.0d-06
1  xcrit = 1.0d-06
1  itmax = 20

c  itmax = 1
1  slope = -1.d+00
1  xguess = xiavin
1  deltax = xiavin/100.
1  phistr = phii n

c  --- the following can be initialized to anything since the
1  --- flags will tell whether they have really been intialized
1  ybest=1.0d+100
1  ysec=1.0d+100
1  yhi=1.0d+100
1  xofyhi=-1.0d+100
1  ylo=-1.0d+100
1  xofylo=1.0d+100
xbest=0.d+00

--- make sure it knows nothing has been initialized ---

secset = .false.
yhiset = .false.
yloset = .false.

xnew = xguess
iter = 0

100 iter = iter + 1
if(iter.gt.itmax) stop
phiold = phistr

---- evaluate y and z at xnew ----

call evali(ynew,xnew,phiold,
1 xitotc,xitota,p0mps,eta,csurf,xij,xi,phistr,it,tox,fine,can,
1 ccath,xitotj,toxj,son,etas)

print 108,phistr

108 format(' after call evali, phistr=\\',lpell,4)

if(yhiset.and.yloset) print 106,xofyhi,xofylo,yhi,ylo

print 114,xnew,ynew,xbest,ybest

if (dabs(ynew).lt.ycrit .or. (dabs(xnew-xbest).le.xcrit))
  go to 400

---- update best, second-best ----

if (iter .eq. 1) go to 130
if (dabs(ynew) .lt. dabs(ybest)) go to 120
  if (secset .and. (dabs(ynew).gt.dabs(ysec)) .and.
1 (ynew*ysec).gt.0.d+00) go to 110
    ysec = ynew
    xsec = xnew
    secset = .true.
    go to 140

110 continue

print 315

315 format(' in iavlp ynew worse than ysec')

istop=1
if(istop.ne.0) stop
if( ynew*ybest .gt. 0.d+00 .and. ynew*ysec .gt. 0.d+00)
go to 317
    call dump(phistr,xnew,xi,p0mps)
    stop
317 xnew = 2.*xbest - xnew
    print 115,xnew,ynew,xsec,ysec,xbest,ybest
    if(iter.lt.itmax) go to 100
    print 316,itmax
316 format(' in iavlp it=itmax=',i2)
    call dump(phistr,xnew,xi,p0mps)
    stop

ysec = ybest
xsec = xbest
secset = .true.
ybest = ynew
xbest = xnew
continue

---- update bounds ----

if ( (ynew .gt. 0.d+00) .or. 
    ((dabs(ynew) .gt. dabs(ylo)) .and. yloset) ) go to 200
    xmove = slope*deltax
    if(iter.eq.1) print 131,xmove,deltax
131 format(' xmove=',lpell.4,' deltax=',lpell.4)
ylo = ynew
    yloset = .true.
xofylo = xnew
200 if ( (ynew .lt. 0.d+00) .or. 
    ((dabs(ynew) .gt. dabs(yhi)) .and. yhiset) ) go to 210
    xmove = -slope*deltax
    if(iter.eq.1) print 131,xmove,deltax
    yhi = ynew
    yhiset = .true.
xofyhi = xnew
210 if ((dabs(ynew) .gt. dabs(ylo)) .and. (dabs(ynew).gt.dabs(yhi))
    .and. yhiset .and. yloset ) print 215

if (iter .eq. 1) go to 320

--- try ynew ---

xnew = extrpl(ybest,ysec,xbest,xsec,slope,deltax)
    if(dabs(xnew-xbest).gt.10.d+00) xnew=xbest+10.*(xnew-xbest)/
    dabs(xnew-xbest)
CHANNEL.FOR

print 800,xnew,xbest,dabs((xnew-xbest)/xbest)

format(' iavlp at try y,xnew=',1pe11.4,' xbest=',1pe11.4,
1 /,' rel diff=',1pe11.4)
if (.not. (yhiset.and.yloset)) go to 330
if ((xnew-xofyhi) * (xnew-xofylo) .gt. 0.d+00) go to 310
go to 330

c --- try bisect ---
c
310 continue
c
print 311
c
311 format(' bisect')
print 801,xnew,xbest,dabs((xnew-xbest)/xbest)
format(' iavlp at bisect,xnew=',1pe11.4,' xbest=',1pe11.4,
1 /,' rel diff=',1pe11.4)
if (.not.(yloset.and.yhiset)) go to 320
xnew = 0.5 * (xofyhi + xofylo)
if(dabs(xnew-xbest).gt.10.d+00) xnew=xbest+10.*(xnew-xbest)
go to 330

c --- move x ---
c
320 xnew = xnew + xmove
print 321,xmove
321 format(' move x, xmove=',1pe11.4)
c
330 if (iter .lt. itmax) go to 100
print 335,itmax,xnew,ynew
call dump(phistr,xnew,xi,p0mps)
print 318
318 format(' too many iterations in iavlp')
stop
c
400 print 405,iter,xnew,ynew
c
400 continue
xiav = xnew
c
temp
iprt=0
call newt(xiav,phistr,p0mps,eta,csurf,xij,xi,xitotc,xitota,
1 tox,fine,can,ccath,iprt,xitotj,toxj,son,etas)
return
end

c function extrpl(yb,ys,xb,xs,slope,deltax)

c
CHANNEL.FOR

implicit double precision (a-h,o-z)
if ((yb .eq. ys) .or. (xb .eq. xs)) go to 100
extrpl = xb - yb*(xb-xs)/(yb-ys)

10 format(' extrpl=',lpe23.16,' dy=',lpell.4,' dx=',
        lpe11.4)
return

100 if (yb .gt. 0.d+00) extrpl = xb - slope*deltax
if (yb .le. 0.d+00) extrpl = xb + slope*deltax
print 20,extrpl
format(' decided to step, extrpl=',lpe23.16)
return
end

subroutine evali(yfcn,xiavin,phiin,
1 xitotc,xitota,p0mps,eta,csurf,xij,xi,phistr,it,tox,fine,can,
1 ccath,xitotj,toxj,son,etas)
implicit double precision (a-h,o-z)
dimension csurf(2,10,10l),tox(2,101),fine(2,5,10l),
1 xij(2,5,10l),xi(2,101),p0mps(2,101,5),eta(2,101,5),
1 can(5,10,101),ccath(5,10,101),xitotj(2,5),toxj(2,101,5),
1 etas(2,101)

common/evprob/xlamda(3),ai(3),z,diodr(10)
common/co/coeff,xn
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
1 nspec,nrxn,one(2,5),rho0
common/matcom/b(20,20),d(20,41)
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/cdata/cncdtol,ftol,xint0(10,101,5),
1 xint1(10,101,5),n2,nd,iseq
common/elecs/v(2),ian,icath
common/geom/pi,hol,nj,ndec,icut
common/gauss/xp(96),gw(96),ngt,xielec(201),
1 xiothr(201),njmore,xx(201)
common/nrdata/phicrt,totcrt,crit,vcell

call philoop(phiin,xiavin,xitotc,xitota,p0mps,eta,csurf,
1 xij,xi,phistr,it,tox,fine,can,ccath,xitotj,toxj,son,etas)

108 format(' after call philoop, phiin=',lpe11.4)
yfcn = xitota - xiavin
return
end

subroutine philoop (phiin,xiavin,xitotc,xitota,
1 p0mps,eta,csurf,xij,xi,phistr,it,tox,fine,can,ccath,
1 xitotj, toxj, son, etas)

---- This program searches for the zero of a monotonic function
---- It also allows for an alternate function to be used to
---- speed the convergence

implicit double precision (a-h,o-z)
logical secset,setz(3),yloset,yhiset,blowup
logical usez

integer best,secbst

dimension csurf(2,10,101), tox(2,101), fine(2,5,101),
1 xij(2,5,101), xi(2,101), p0mps(2,101,5), eta(2,101,5),
1 can(5,10,101), ccath(5,10,101), xitotj(2,5), toxj(2,101,5),
1 etas(2,101)

common/evprob/xlamda(3), ai(3), z, diodr(10)
common/co/coeff, xn
common/rxnpars/cref(10), cre(10), porq(2,10,5), ip, iq, uref(5),
1 nspec, nrxn, one(2,5), rho0
common/matcom/b(20,20), d(20,41)
common/kinpars/xj(5), ratson(10,5), ason(2,5), ifor, iback
common/cdata/cnctol, ftol, xint0(10,101,5),
1 xintl(10,101,5), n2, nd, isecl
common/elects/v(2), ian, icath
common/geom/pi, hol, nj, ndec, icut
common/gauss/xp(96), gw(96), ngt, xielec(201),
1 xiothr(201), njmore, xx(201)
common/nrdata/phi, totcrt, crit, vcell
common/diagn/iterp

data best/1/ , secbst/2/ , new/3/

105 format(/,' phistr iteration',i3,' x=',lp23.16,' y=',lp23.16)
106 format(' xofyhi=',lp23.16,' xofylo=',lp23.16,' yhi=',lpell.4,
1   ' ylo=',lpell.4)
114 format(' x=',lpell.4,
1   ' y=',lpell.4,' xbest=',lpell.4,' ybest=',lpell.4)
115 format(' philoop',/,' x=',lpell.4,
1   ' y=',lpell.4,' xsec=',lpell.4,' ysec=',lpell.4,' xbest=',
CHANNEL.FOR

1  lpell.4,' ybest=',lpell.4)
215  format(' beware y worse than ylo and yhi')
335  format(' philoop no convergence in',i3,' iterations',/,
1  x='' ,lpell.4,' y='' ,lpell.4)
405  format(/,' converged in',i3,' iterations',/,' x=' ,lpell.4,' y=',
1  lpell.4)

c  ycrit = 1.0d-06
xcrit = 1.0d-08
itmax = 40
itmax = 1
slope = -1.0d+00
xguess = phiin
deltax = 2.0d+00
usez = .true.
ipass=0
slopeo=0.d+00

c  ybest=1.0d+100
ysec=1.0d+100
yhi=1.0d+100
xofyhi=1.0d+100
ylo=-1.0d+100
xofylo=1.0d+100
xbest=0.d+00

c  --- make sure it knows nothing has been initialized ---
c
secset = .false.
yhiset = .false.
yloset = .false.
c
20 do 20 i=best,new
20  setz(i) = .false.
c
c  xnew = xguess
iterp = 0

c  c
100  iterp = iterp + 1
if(iterp.gt.itmax) stop

c  c

c  ---- evaluate y and z at xnew ----
c
  call evalp(ynew,znew,setz(new),xnew,xiavin,
CHANNEL.FOR

1  xitotc,xitota,p0mps,eta,csurf,xij,xi,tox,fine,can,ccath,
1  xitotj,txoj,son,etas)
print 105,interp,xnew,ynew
print *,xitota',xitota',xitotc',xitotc
print *,',jx ian icath'
do 401,jjx=1,nj
print 401,jjx,xi(l,jjx),xi(2,jjx)
format(1x,i2,1x,1pel1.4,1x,ell.4)
continue
print 106,xofyhi,xofylo,yhi,ylo
print 114,xnew,ynew,xbest,ybest
if (dabs(ynew).lt.ycrit .or. (dabs(xnew-xbest).le.xcrit))
  go to 400
---- update best, second-best ----
if (interp .eq. 1) go to 130
if (dabs(ynew) .lt. dabs(ybest)) go to 120
  (ynew*ysec).gt.0.d+00) go to 110
    ysec = ynew
    xsec = xnew
    secset = .true.
    setz(secbst) = setz(new)
    go to 140
110 continue
print 115,xnew,ynew,xsec,ysec,xbest,ybest
print 315
format(' in philoop ynew worse than ysec')
if( ynew*ybest .gt. 0.d+00 .and. ynew*ysec .gt. 0.d+00 )
  go to 317
    call dump(xnew,xiavin,xi,p0mps)
    stop
317 xnew = 2.*xbest - xnew
print 115,xnew,ynew,xsec,ysec,xbest,ybest
if(interp.lt.itmax) go to 100
print 316, itmax
format(' in philop it=itmax=',i2)
call dump(xnew,xiavin,xi,p0mps)
stopp
120 ysec = ybest
xsec = xbest
secset = .true.
setz(secbst) = setz(new)
130 ybest = ynew
xbest = xnew
setz(best) = setz(new)

continue

---- update bounds ----

print 141,ynew,ylo,yhi
format(' ynew=',1pe11.4,' ylo,yhi=',1pe11.4,1x,e11.4)

if ( (ynew .gt. 0.d+00) .or. 
1   ((dabs(ynew) .gt. dabs(ylo)) .and. yloset) ) go to 200
  xmove = slope*deltax
  ylo = ynew
  yloset = .true.
  xofylo = xnew
200 if ( (ynew .1t. 0.d+00) .or. 
1   ((dabs(ynew) .gt. dabs(yhi)) .and. yhiset) ) go to 210
  xmove = -slope*deltax
  yhi = ynew
  yhiset = .true.
  xofyhi = xnew

if ((dabs(ynew) .gt. dabs(ylo)) .and. (dabs(ynew).gt.dabs(yhi)) 
1   .and. yhiset .and. yloset ) print 215

if (iterp .eq. 1) go to 320

--- try znew ---

temp
go to 310

--- try ynew ---

xnew = extrpl(ybest,ysec,xbest,xsec,slope,deltax)
print 306,xnew,xofyhi,xofylo,xbest,xsec
format(' at try y, extrpl=',lpe23.16,/, ' xofyhi=', 
1    lpe23.16,' xofylo=',lpe23.16,/, ' xbest=',lpe11.4,' xsec=', 
1    lpe11.4)
if(blowup(nrxn,p0mps,xiavin,xnew)) print 341
format(' blows up at try y')
if(blowup(nrxn,p0mps,xiavin,xnew)) go to 310
if(dabs(xnew-xbest).gt.10.d+00) xnew=xbest+10.*(xnew-xbest)
1 /dabs(xnew-xbest)
if (.not. (yhiset.and.yloset)) go to 330
if ((xnew-xofyhi) * (xnew-xofylo) .gt. 0.d+00) go to 310
   go to 330
--- try bisect ---
310 continue
if(ybest*ysec .le. 0.d+00) ipass=1
if(ipass.ne.0) go to 300
if(.not.(yhiset.and.yloset)) go to 320
xnew = 0.5 * (xofyhi + xofylo)
if(dabs(xnew-xbest).gt.10.d+00) xnew=xbest+10.*(xnew-xbest)
   /dabs(xnew-xbest)
   print 800,xnew,xbest,dabs((xnew-xbest)/xbest)
800 format(' philoop at bisect,xnew=',lpell.4,' xbest=',lpell.4,
   1 '/,' rel diff=',lpell.4)
   if(blowup(nrxn,p0mps,xiavin,xnew)) print 312,xnew
312 format(' blows up at bisect, xnew=',lpell.4)
   if(.not. blowup(nrxn,p0mps,xiavin,xnew)) go to 330
   call dump(phistr,xiavin,xi,p0mps)
   print 319
   format(' blowup in philoop')
   stop
--- move x ---
320 xnew = xnew + xmove
   print 321,xnew
321 format(' move x,xnew=',lpell.4)
   if(.not. blowup(nrxn,p0mps,xiavin,xnew)) go to 330
   xnew = xbest + xmove
   print 321,xnew
   if(blowup(nrxn,p0mps,xiavin,xnew)) print 322
322 format(' blows up at move x')
   if(.not. blowup(nrxn,p0mps,xiavin,xnew)) go to 330
   call dump(phistr,xiavin,xi,p0mps)
   print 319
   stop
330 if (iterp .lt. itmax) go to 100
   print 335,itmax,xnew,ynew
   call dump(xnew,xiavin,xi,p0mps)
   stop
   print 405,iterp,xnew,ynew
400 continue
phistr = xnew
return
end

subroutine evalp(yfcn, zfcn, zinrng, phiin, xiavin,
1 xitotc, xitota, p0mps, eta, csurf, xij, xi, tox, fine, can, ccath,
1 xitotj, toxj, son, etas)

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

dimension csurf(2, 10, 101), tox(2, 101), fine(2, 5, 101),
1 xij(2, 5, 101), xi(2, 101), p0mps(2, 101, 5), eta(2, 101, 5),
1 can(5, 10, 101), ccath(5, 10, 101), xitotj(2, 5), toxj(2, 101, 5),
1 etas(2, 101)

c common/evprob/xlamda(3), ai(3), z, diodr(10)
c common/co/coeff, xn
common/rxnparse cref(10), cre(10), porq(2, 10, 5), ip, iq, uref(5),
1 nspec, nrnxn, one(2, 5), rho0
common/matcom/b(20, 20), d(20, 41)
c common/kinpars/xj(5), ratson(10, 5), ason(2, 5), ifor, iback
common/cdata/cnctol, fto1, xint0(10, 101, 5),
1 xintl(10, 101, 5), n2, nd, isec
common/elecs/jv(2), ian, icath
common/geom/pi, hol, nj, ndec, icut
common/gauss/xp(96), gw(96), ngt, xielec(201),
1 xiothr(201), njmore, xx(201)
c common/nrdata/phicrt, totcrt, crit, vcell
c common/diagn/iterp

c print *, ' at call newt, iterp=', iterp
iprt = 0
call newt(xiavin, phiin, p0mps, eta, csurf, xij, xi, xitotc, xitota,
1 tox, fine, can, ccath, iprt, xitotj, toxj, son, etas)

c 102 print 102, xitotc, xitota
c 103 format(' xitotc=', lpe11.4, ' xitota=', lpe11.4)

c 104 format(1x, lpe11.4, lx, ell.4, 3x, ell.4, lx, ell.4)
logical function blowup(nrxn,p0mps,xiavin,phistr)
implicit double precision (a-h,o-z)
common/geom/pi,hol,nj,ndec,icut
common/kinpars/xj(5),ratson(10,5),ascon(2,5),ifor,iback
common/elecs/v(2),ian,icath
dimension p0mps(2,101,5)
print 100,phistr
format(' in blowup, phistr=',lpell.4)
expmax=2.303d+00*290.d+00
blowup=.false.
do 270 idec=1,ndec
do 270 jx=1,nj
do 270 ielec=ian,icath
  eta=v(ielec)-(p0mps(ielec,jx,idec)*xiavin+phistr)
do 271 j=1,nrxn
  if(idec.eq.1 .and. jx.eq.1) print 101,ascon(ifor,j)*eta,
    ascon(iback,j)*eta
format(' in blowup, alp f eta=',lpell.4,/', alp b eta=',
    lpell.4)
  if(dabs(ascon(ifor,j)*eta) .lt. expmax .or.
    dabs(ascon(iback,j)*eta) .lt. expmax)
    go to 271
  blowup=.true.
  return
continue
271 continue
270 continue
return
end
subroutine primry(iprimc, ielec, xi, x, icertx, xix, xiint, xiend)

This subroutine calculates the primary distribution in a channel flow cell (except at the end points)

Input variables

pi, hol, nj
ian, icath

Output variables

xi(ielec, jx) = normalized current density on electrode ielec at location jx

implicit double precision (a-h,o-z)
dimension xi(2,101), sign(2), iprimc(2), xiint(2,96,5)

common/elecs/v(2), ian, icath
common/geom/pi, hol, nj, ndec, icut
common/gauss/jxp(96), gw(96), ngt, xielec(201),
1 xiothr(201), njmore, xx(201)
	sign(ian) = 1.d+00
	sign(icath) = -1.d+00

xi(ielec, 1) = xiend*sign(ielec)
xi(ielec, nj) = xi(ielec, 1)*sign(ielec)

a0 = 1.3862944d+00
a1 = 0.1119723d+00
a2 = 0.0725296d+00
b0 = 0.5d+00
b1 = 0.1213478d+00
b2 = 0.0288729d+00

eps = pi/(2.*hol)
xml = 1.-(dtanh(eps))**2
njml = nj - 1
term = dlog( (dexp(2.*eps) + 2. + dexp(-2.*eps)) / 4.)

xk = a0 + a1*xml + a2*xml+xmll + (b0 + b1+xmll + b2+xmll+xmll)*
1 term

xnum = eps*dcosh(eps)/xk

if(icertx.ne.0) go to 101
do 100 jx=2,njml
    x=dfloat(jx-1)/dfloat(njml)
100   xi(ielec,jx)=sign(ielec)*xnum/dsqrt((dsinh(eps)**2 -
               (dsinh((2.*x-1.)*eps))**2)

go to 102

101 xix=xnum/dsqrt((dsinh(eps)**2 - (dsinh((2.*x-1.)*eps))**2)
102 if(iprimc(1).ne.0 .and. iprimc(2).ne.0) return
denom=dfloat(njml)
hold=0.d+00

do 141 idec=1,ndec
    h=dfloat(icut)**(idec-ndec)/denom
    do 149 jg=1,ngt
       x=xp(jg)*(njml*h - hold) + hold
       xiint(ielec,jg,idec)=sign(ielec)*xnum/
                      dsqrt((dsinh(eps)**2 - (dsinh((2.*x-1.)*eps))**2)
    149   continue
    hold=h*njml
141   continue

return
end

subroutine intcxi(nspec,xint0,xint1)

This subroutine calculates xint0(i,n-k+1,idec), 1/dx times the
integral from (k-1)dx to kdx of dC/dxi, where C is the
solution to the asymmetric Graetz problem in channel flow and
xi is the dimensionless normal variable, y\h.

Input variables
    nspec = number of species
    xlamda,ai,z,diodr
    nj,ndec

Output variables
    xint0(i,n-k+1,idec)     (described above)
    xint1(i,n-k+1,idec)

implicit double precision (a-h,o-z)
dimension xint0(10,101,5),xint1(10,101,5),array(5)
common/evprob/xlamda(3),ai(3),z,diodr(10)
common/geom/pi,hol,nj,ndec,icut

a=1.35659745d+00
b=0.2d+00
c=0.060733452d+00

aa=0.9594d+00
bb=0.6069d+00
cc=0.4512d+00
dd=0.276d+00

njml=nj-1
n=njml
h=dfloat(icut)**(1-ndec)/dfloat(njml)

do 1000 idec=1,ndec
    do 999 ixi=1,2
    do 999 i=1,nspec
        do 999 k=1,n
            nmk=n-k
            nmkp1=n-k+1
            den=z/(h*diodr(i))
            zeta=nmk/den
            zeta11=(nmkp1)/den
            if(ixi .eq. 2) go to 500
            xi=0
            if(zeta .lt. 0.11d+00 .and. zeta11.gt.0.11d+00) go to 600
            if(zeta.ge.0.11d+00) go to 100
            xint0(i,nmkp1,idec)=1.5*den**(1./3.)*a*
            1 (nmk**(2./3.)-(nmk+1)**(2./3.))
            1 + b -0.75*den**(-1./3.)*c*(nmk**(4./3.)-(nmk+1)**(4./3.))
            go to 999

100    sum=0.0d+00
    do 110 j=1,3
110    sum=sum+dabs(ai(j))/xlamda(j) * (dexp(-xlamda(j)*zeta) - dexp
1      (-xlamda(j)*zeta11))
    xint0(i,nmkp1,idec)=-1. - 2.*den*sum
    go to 999

600    xint0(i,nmkp1,idec)=1.5*den**(1./3.)*a*(nmk**(2./3.) -
1 (0.11*den)**(2./3.)) + b*(0.11*den-nmk) - 0.75*den**(-1./3.)*c*
1 (nmk**(4./3.) - (0.11*den)**(4./3.))
    sum=0.0d+00
do 610 j=1, 3
   sum=sum+dabs(ai(j))/xlamda(j) * (dexp(-xlamda(j)*0.11) -
   dexp(-xlamda(j)*zetall))
   sum=0.11*den - nmkp1 - 2.*den*sum
   xint0(i,nmkp1,idec)=xint0(i,nmkp1,idec)+sum
   go to 999
   c
   c
   xi=1
   c
   500 if(zeta<le.0.18d+00 .and. zetall.ge.0.18d+00) go to 620
   if(zeta.gt.0.18d+00) go to 510
   f0=0.d+00
   if(n.eq.k) go to 505
   f0=-dexp(aa - bb/zeta - cc*dexp(-dd/zeta))
   505 xint1(i,nmkp1,idec)=(-dexp(aa - bb/zetall -
   1 cc*dexp(-dd/zetall)) + f0)/2.
   go to 999
   c
   510 continue
   sum=0.d+00
   do 520 j=1, 3
      520 sum=sum+ai(j)/xlamda(j)*(dexp(-xlamda(j)*zetall) -
          dexp(-xlamda(j)*zetall))
   xint1(i,nmkp1,idec)=-1.+2.*den*sum
   go to 999
   c
   620 zll=0.18d+00
   f0=0.d+00
   if(n.eq.k) go to 640
   f0=-dexp(aa - bb/zeta - cc*dexp(-dd/zeta))
   640 xint1(i,nmkp1,idec)=(-dexp(aa - bb/zll -
   1 cc*dexp(-dd/zll)) + f0)/2.
   *(-nmk+zll*den)
   sum=0.0d+00
   do 660 j=1,3
      sum=sum+ai(j)/xlamda(j) * (dexp(-xlamda(j)*0.18) - dexp(1-
          -xlamda(j)*zetall))
   sum=2.*den*sum + zll*den-nmkp1
   xint1(i,nmkp1,idec)=xint1(i,nmkp1,idec) + sum
   999 continue
   h=h*dfloat(icut)
   1000 continue
   c
   return
   end
   c
   subroutine oldint(islec,nj,yarray,x,y)
**CHANNEL.FOR**

```plaintext
implicit double precision (a-h,o-z)
common/elecs/v(2),ian,icath
dimension yarray(nj),ydum(101)
njml=nj-1
h=1./dfloat(njml)
xl=0.d+00
xn=1.d+00

unrefined:
al=0.3d+00
ar=0.2d+00
if(ielec.eq.icath) ar=0.5d+00
eps1 = (al/yarray(1))**2
eps2 = (ar/yarray(nj))**2
do 100 jx=1,nj
   xx=dfloat(jx-1)/dfloat(njml)
   ydum(jx) = yarray(jx) * dsqrt( (xx+eps1)*(1.-xx+eps2) )
100 continue
call interpl(xl,xn,h,nj,ydum,x,3,yout)
y = yout / dsqrt( (x+eps1)*(1.-x+eps2) )
return
end

subroutine interpl(xl,xn,h,n,y,x,m,yout)

subroutine to interpolate in a table of uniformly spaced values.
parameters are -
xl,xn  beginning and ending x values
h      delta x - the uniform spacing
n      number of entries in the table
y      array of function values
x      value at which y is to be interpolated
m      degree of interpolating polynomial. The subroutine
       will handle up to 10th degree, but usually the
       degree will be less than 10 to avoid round-off
       errors.
yout   the interpolated y value returned to the caller.

An array d is used in the subroutine to hold the delta y's.
```

```
implicit double precision (a-h,o-z)
dimension y(n),d(l0)

First find proper subscript for y0 so that x is centered in the domain as well as possible. This subscript value is j.

fm = m + 1
j = (x - x1)/h - fm/2. + 2.
if (x .le. x1 + fm/2.*h) j = 1
if (x .ge. x1 - fm/2.*h) j = n - m
fj = j
x0 = x1 + (fj - 1.)*h
y0 = y(j)

compute the differences that are needed

do 10 i=1,m
   d(i) = y(j+1) - y(j)
   j = j + 1
10 continue

if (m .eq. 1) go to 25

do 20 j=2,m
   do 15 i=j,m
      k = m - i + j
      d(k) = d(k) - d(k-1)
15 continue
20 continue

compute s value

25 s = (x - x0)/h

compute interpolated y value

yout = y0
fnum = s
den = 1.d+00

do 30 i=1,m
   fi = i
   yout = yout + fnum/den*d(i)
   fnum = fnum*(s - fi)
30 continue
```
den = den*(fi + 1.)
continue
return
end

subroutine extrap(n,x,f,xin,fout)
This subroutine uses a Lagrange formula to extrapolate a function

Input variables

n = number of points to be used in extrapolation
x = array of values at which function is given
f = array of function values
xin = value at which function is to be extrapolated

Output variables

fout = value of function at x=xin

implicit double precision (a-h,o-z)
dimension x(n:),f(n)

fout=0.d+00

do 100 i=1,n
  if(xin.eq.x(i)) go to 999
  prodn=1.d+00
  prodd=1.d+00
  do 110 j=1,n
    if(i.eq.j) go to 110
    prodn=prodn*(xin-x(j))
    prodd=prodd*(x(i)-x(j))
  110 continue
  100 fout=fout + f(i)*prodn/prodd

go to 120

999 print 1000,xin
1000 format(' wrong input for xin, xin=’,1pe11.4)

120 return
end

SUBROUTINE MATINV(N,M,DETERM)

This subroutine solves a matrix equation of the form BX=D. The answer X is put into the first column of D. Note that D should be dimensioned nxm, where m=2n+1 and that all columns of D (except the first) should be initialized with zeros.

Input variables

n = dimension of square b matrix
m = 2n+1
b = matrix in equation bx=d
d : first column contains vector d in bx=d

Output variables

d : first column contains vector x in bx=d
determ = 0. if matrix has zero determinant

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

DETERM=1.0
DO 1 I=1,N
1     ID(I)=0
     DO 18 NN=1,N
     BMAX=1.1
     DO 6 I=1,N
     IF(ID(I).NE.0) GOTO 6
     BNEXT=0.0
     BTRY=0.0
     DO 5 J=1,N
     IF(ID(J).NE.0) GOTO 5
     IF(ABS(B(I,J)).LE.BNEXT) GOTO 5
     BNEXT=ABS(B(I,J))
     IF(BNEXT.LE.BTRY) GOTO 5
     BNEXT=BTRY
     BTRY=ABS(B(I,J))
     JC=J
5     CONTINUE
     IF(BNEXT.GE.BMAX*BTRY) GOTO 6
     BMAX=BNEXT/BTRY
     IROW=I
     JCOL=JC
6     CONTINUE
     IF(ID(JC).EQ.0) GOTO 8
     DETERM=0.0
RETURN
8  ID(JCOL)=1
  IF(JCOL.EQ.IROW) GOTO 12
  DO 10 J=1,N
  SAVE=B(IROW,J)
  B(IROW,J)=B(JCOL,J)
10  B(JCOL,J)=SAVE
  DO 11 K=1,M
  SAVE=D(IROW,K)
  D(IROW,K)=D(JCOL,K)
11  D(JCOL,K)=SAVE
12  F=1.0/B(JCOL,JCOL)
  DO 13 J=1,N
  13  B(JCOL,J)=B(JCOL,J)*F
  DO 14 K=1,M
  14  D(JCOL,K)=D(JCOL,K)*F
  DO 18 I=1,N
  IF(I.EQ.JCOL) GO TO 18
  F=B(I,JCOL)
  DO 16 J=1,N
  16  B(I,J)=B(I,J)-F*B(JCOL,J)
  DO 17 K=1,M
  17  D(I,K)=D(I,K)-F*D(JCOL,K)
  CONTINUE
18  RETURN
END

subroutine newt(xiav,phistr,p0mps,eta,csurf,xij,xi,xitotc,
  xitota,tox,fine,can,ccath,iprt,xitotj,toxj,son,etas)
  run with routines
curnt
other
fcns
matinv
bigtrm
simps
This subroutine solves for \( <i,cath>(x) \) and \( <i,an>(x) \) using
a multidimensional Newton-Raphson method. The equations to be
solved result from equating the Fick's-Law and Faraday's-Law
expressions for the flux. The Fick's-Law expression
incorporates the method of Acrivos and Chambre to discretize
the integral equation (Ind. Eng. Chem., 49, 1025(1957))
Input variables
  isec = 0 if tertiary current distribution, nonzero
if secondary

\[
\text{xiav} = i^{avg} \left( -n^{<m>/s<Rm>} \right) \text{FL}/(RT K^{<b>})
\]

\[
\phi_{\text{istr}} = (\phi^* \left( -n^{<m>/F/(s<Rm> RT)} \right)
\]

\[
p0mps(ielec,jx,idec) = (\phi^0 - \phi^*) K^{<b>} / (i^{avg} L)
\]

csurf (input at jx=1 only)

\[
\text{diodr}
\]

\[
\text{coeff}
\]

\[
nj, ndec
\]

cref, porq, ip, iq, uref, nspec, nrxn, one

\[
\text{xj, ratson, ason, ifor, iback}
\]

cnctol, fto1, xint0, xint1, n2, nd

\[
v, ian, icath
\]

Intermediate variables in common

b, d

Output variables

\[
\text{eta}(ielec,jx,idec) = (V^{<ielec>} - \phi^0)^* \left( -n^{<m>/F/(s<Rm> RT)} \right)
\]

\[
\text{csurf}(ielec,i,jx) = \text{dimensionless surface concentration of species i on electrode ielec at mesh point jx}
\]

\[
\text{can}(idec,i,jx) = \text{dimensionless anode surface conc of species i at mesh point jx in decade idec}
\]

\[
\text{ccath}(idec,i,jx) = \text{dimensionless cathode surface conc of species i at mesh point jx in decade idec}
\]

\[
\text{xij}(ielec,j,jx) = -n^{<m>/s<R,m>} (\text{FL}/(RT K^{<b>})) i<j, elec>
\]

\[-\text{partial current density of reaction j on electrode ielec at location jx}
\]

\[
\text{xi}(ielec,jx) = \text{current density on electrode ielec at mesh point jx}
\]

\[
\text{xitotc} = \text{integral of current along cathode}
\]

\[
\text{xitota} = \text{integral of current along anode}
\]

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

logical limcur(2)

dimension csurf(2,10,101),eta(2,101,5),fs(20),tf(2,5,101),
1 tb(2,5,101),big(4),p0mps(2,101,5),xij(2,5,101),xi(2,101),
1 xintgd(101),xitot(2),save(2),sum0(2,10),suml(2,10),tox(2,101),
CHANNEL.FOR

1   fine(2,5,101),can(5,10,101),xitotj(2,5),savej(2,5),savem(2),
1   toxj(2,101,5),ccath(5,10,101),etas(2,101),
1   farpot(2),savdum(2)

1   common/evprob/xlamsda(3),ai(3),z,diodr(10)
1   common/co/coeff,xn
1   common/geom/pi,hol,nj,ndec,icut
1   common/rxmpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
1   nspec,nrnn,one(2,5),rho0
1   common/matcom/b(20,20),d(20,41)
1   common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
1   common/cdata/cnctol,ftol,xint0(10,101,5),
1   xint1(10,101,5),n2,nd,isecc
1   common/elecs/v(2),ian,icath
1   common/diagn/interp

1   istop=0

1   limcur(ian)=.true.
1   limcur(icath)=.true.
1   rtonf=8.3143*298.*son/96487.
1   if(iprt.ne.0) print 158
1   158 format(' big printout goes here')
1   icount = 0
1   n2=nb+n2+1
1   expmax=2.303d+00*290.d+00

1   damping factor if concentrations go negative

1   cdamp=1.0d-06

1   calculate V - phi 0

1   do 270 idec=1,ndec
1     do 270 jx=1,nj
1       do 270 ielec=ian,icath
1         eta(ielec,jx,idec)=v(ielec)-(p0mps(ielec,jx,idec)*xiav+phistr)
1           if(dabs(eta(ielec,jx,idec)*rtonf).lt.2.5d0)
1             limcur(ielec)=.false.
1     enddo 270 jx
1   enddo 270 idec
1   do 271 j=1,nrxn
1     if(dabs(ason(ifor,j)*eta(ielec,jx,idec)) .lt. expmax .or.
1       dabs(ason(iback,j)*eta(ielec,jx,idec)) .lt. expmax)
1       go to 271
1     enddo 271 j
1   call dump(phistr,xiav,xi,p0mps)
print 276
  format(' stopped in newt because exp blows up')
stop

continue

if(limcur(ian)) print *, ' limiting current assumed on anode'
if(limcur(icath)) print *, ' limiting current assumed on cath'
if(iprt.ne.0) print 370
  format(' eta anode')
  if(iprt.ne.0) print 371, ((eta(ian,jx,ndec)), jx=1,nj)
  format(5el2.4)

if(iprt.ne.0) print 372
  format(' rtonf eta anode')
  if(iprt.ne.0) print 373, ((eta(ian,jx,ndec)*rtonf), jx=1,nj)
  format(Sel2.4)

if(iprt.ne.0) print 374
  format(' eta cathode')
  if(iprt.ne.0) print 375, ((eta(icath,jx,ndec)), jx=1,nj)
  format(5el2.4)

if(iprt.ne.0) print 376
  format(' rtonf eta cathode')
  if(iprt.ne.0) print 377, ((eta(icath,jx,ndec)*rtonf), jx=1,nj)
  if(istop.ne.0) stop

get currents at first mesh point

jx=1
idec = ndec
do 376 ielec=ian,icath
  call curr(ielec,csurf,eta,tf,tb,xi(jx),xi,idec,etas)
  if(iprt.ne.0) print 375, jx,xi(ian,1),xi(icath,1)
  format(' jx=',i2, ' ian=',lpell.4,' icath=',lpell.4)
  kend=(nj-1)/icut + 1
  njml=nj-1
  do 260 ielec=ian,icath
    tox(ielec,1)=0.d+00
    xitot(ielec)=0.d+00
farpot(ielec) = 0.d0
do 260 j=1,nrxn
toxj(ielec,1,j)=0.d+00
260 xitotj(ielec,j)=0.d+00
c
   jbegin=2
do 115 idec=1,ndec
   temp
c
   print 116,idec
   c 116
   c 117
   c 118
   print 116,can(l),ccath,can(2),ccath'
c
   do 118 jx=1,nj
   print 117,csurf(ian,l,jx),csurf(icath,l,jx),csurf(ian,2,jx),
c
   csurf(icath,2,jx)
c
   format(1x,lpe11.4,lx,e11.4,1x,e11.4,1x,ell.4)
c
   continue
   h= dfloat(icut)**(idec-ndec)/dfloat(njml)
c
   do 170 jx=jbegin,nj
   if(jx.eq.2) go to 179
   jxm2 = jx - 2
   jxml = jx - 1
   if(jx.eq.3) go to 184
   do 180 i=1,nspec
      csurf(ian,i,jx) = 2.*csurf(ian,i,jxml) - csurf(ian,i,jxm2)
      csurf(icath,i,jx) = dexp(2.*dlog(csurf(icath,i,jxm2))
      if(csurf(ian,i,jx).le.0.d+00) csurf(ian,i,jx) = dexp(2.1
      dlog(csurf(ian,i,jxml)) - dlog(csurf(ian,i,jxm2))
      istop=0
      if(csurf(ian,i,jx).le.0.d+00 .or.
      csurf(icath,i,jx).le.0.d+00) istop=1
      if(istop.eq.1) print 181,jx,idec,csurf(ian,i,jx),
c
      csurf(icath,i,jx)
181 format(’ at jx=’,i2,’ decade’,i2,’ can,cath=’,lpe11.4,lx,
c
      ell.4)
      if(istop.ne.1) go to 180
      call dump(phistr,xiav,xi,p0mps)
      print 183
5 format(’ stopped in newt because predicted conc neg.’)
      stop
180 continue
go to 179
184   do 185 i=1,nspec
185   do 185 ielec=ian,icath
186   csurf(ielec,i,jx) = csurf(ielec,i,jxml)
187   continue
177   do 172 i=1,nspec
172   continue
171   if(jx.eq.2) go to 172
170   n=jx-1
169   nml=n-1
168   do 171 k=1, nml
171     kpl=k+1
171     nmkpl=n-k+1
167     if(ielec.eq.ian,icath)
166     sum0(ielec,i)=sum0(ielec,i) + (csurf(ielec,i,kpl)
165       -csurf(ielec,i,k))*xint0(i,nmkpl,idec)
165     sum1(ielec,i) = sum1(ielec,i) + (csurf(ielec,i,kpl)
164       -csurf(ielec,i,k))*xint1(i,nmkpl,idec)
164     continue
163   print 178,jx,sum0(ian,l),sum0(icath,l),sum1(ian,l),
162       sum1(icath,l),
161     format(' jx=',i2,' sum0 an,cath=',lpell.4,lx,ell.4,
160       ' sum1 an,cath=',lpell.4,lx,ell.4)
161     iexit=0
160     itmax = 40
159   do 130 iter=1,itmax
158     initialize current to zero for summing
157     if(iseq.eq.0.d+00) go to 377
156     do 378 ielec=ian,icath
155     call curnt(ielec,csurf,eta,tf,tb,xij,xi,jx,idec,etas)
154     if(iprt.ne.0 .and. isec.ne.0) print 375,jx,xi(ian,jx),
153       xi(icath,jx)
153     if(isec.ne.0) go to 170
152     do 50 jcol=1,n2
do 50 irow=1,n2
   b(irow,jcol)=0.d+00
   d(irow,jcol)=0.d+00
50   do 51 jcol=n2p1,nd
   do 51 irow=1,n2
51   d(irow,jcol)=0.d+00
ibefor=0
   if(iexit.ne.0) ibefor=1
   call fcns(sum0,sum1,idec,csurf,jx,eta,fs,b,big,xij,xi,etas)
   if(idec.eq.ndec) print 375,jx,xi(ian,jx),xi(icath,jx)
   do 100 irow=1,n2
100  d(irow,i)=fs(irow)
112 format(4(lx,e11.4))
   call matinv(n2,nd,determ)
   if(determ.ne.0.d+00) go to 119
   print 110,idec,jx
110 format(27h zero determinant in matinv,' idec=' ,i2,' jx=' ,i2)
   do 121 i=1,n2
      print 113,(b(i,j),j=1,n2)
113 format(' b matrix',lx,lpe11.4,3(lx,ell.4))
   continue
   stop
119 iexit=0
   do 120 i=1,nspec
      do 120 ielec=ian,icath
         ishift=0
         if(ielec.eq.icath) ishift=nspec
         irow=i+ishift
         if((d(irow,1)+csurf(ielec,i,jx)).le.0.d+00) print *,
            ' d+surf.le.0, go to 150'
             d+surf.le.0, go to 150
         if((d(irow,1)+csurf(ielec,i,jx)).le.0.d+00) go to 150
         csurf(ielec,i,jx)=d(irow,1)+csurf(ielec,i,jx)
         if((dabs(d(irow,1)/csurf(ielec,i,jx)).ge.cnctol .and.
            c 1  csurf(ielec,i,jx).gt.cnctol)) print *,
            ' d/c.ge.cnctol, go to 219,d,csurf',d(irow,1),
            c 1  csurf(ielec,i,jx)
          if((dabs(d(irow,1)/csurf(ielec,i,jx)).ge.cnctol .and.
             c 1   csurf(ielec,i,jx).le.cnctol)) print *,
              ' warning in newt'
             if((dabs(d(irow,1)/csurf(ielec,i,jx)).ge.cnctol .and.
                c 1     csurf(ielec,i,jx).le.cnctol)) go to 219
CHANNEL.FOR

150 go to 120
   csurf(ielec,i,jx)=csurf(ielec,i,jx)*cdamp
c   if((dabs(d(irow,1)/csurf(ielec,i,jx)).ge.cnctol)) go to 219
219 iexit=1
120 continue
c make sure it converges by N-R method and not by damping
c   if(iexit.ne.0) print *, 'iexit.ne.0',iexit
c   if(ibefor.ne.0) print *, 'ibefor.ne.0',ibefor
c   if(iexit.ne.0 .or. ibefor.ne.0) go to 130
c make sure that original equation is satisfied
c   call bigtrm(big,biggst)
c dangerous
c   if(ftol*biggst.le.1.d-10) go to 163
c do 160 irow=1,n2
   if(dabs(fs(irow)).gt.ftol*biggst) print *, 'fs.gt.ftol*biggst at irow',irow,fs(irow),ftol*biggst
   if(dabs(fs(irow)).gt.ftol*biggst) go to 130
160 continue
c 163 continue
icount = icount + iter
c   if(iter.ge.20) print 162,iter,idec,jx
162 format('newt took',i3,'iterations for dec',i2,'jx=',i2)
   if(iter.ge.20 .and. jx.gt.3) stop
go to 170
130 continue
c   if(itmax.ne.1) print 140,itmax,idec,jx,csurf(ian,1,jx),
1 csurf(icath,1,jx)
140 format('no convergence in newt after',i4,
1 'iterations at decade',i2,'jx=',i3,'can,cath=',lpell.4,
1 lx,ell.4)
   if(itmax.eq.1) go to 170
call dump(phistr,xiav,xi,pOmps)
stop
c 170 continue
c call totcurr(jbegin,h,xi,savdum, tox)
call gaussq(1,xi,jbegin,idec,save(ian), save(icath),tox)
call gaussq(2, xi, jbegin, ielec, savem(ian), savem(icath), tox)
print *, 'decade', idec, 'savem', savem(1)
c
do 473 ielec=ian, icath
    xitot(ielec)=xitot(ielec)+save(ielec)
    farpot(ielec)=farpot(ielec)-savem(ielec)/(2.*hol)
    do 474 jx=jbegin,nj
        tox(ielec,jx) = tox(ielec,jx) + tox(ielec,jbegin-1)
    474 continue
    473 continue

do 15 k=1,nj
    do 15 ielec=ian, icath
        fine(ielec,idec,k)=xi(ielec,k)
    15 continue

do 16 i=1,nspec
    do 16 k=1,nj
        ccath(idec,i,k) = csurf(icath,i,k)
    16 continue

cath(icath,i,k) = csurf(ian,i,k)
c if(idec.eq.ndec) go to 115

do 12 k=1,kend
    jj = k*icut - icut + 1
    do 12 ielec=ian, icath
        xi(ielec,k) = xi(ielec,jj)
        tox(ielec,k) = tox(ielec,jj)
        etas(ielec,k) = etas(ielec,jj)
        tox(ielec,k) = tox(ielec,jj)
        do 14 j=1,nrxn
            toxj(ielec,k,j)=toxj(ielec,jj,j)
        14 continue
        do 12 i=1,nspec
            csurf(ielec,i,k)=csurf(ielec,i,jj)
        12 continue

jbegin=kend+1
continue

c xitotc=xitot(icath)
xitota=xitot(ian)
print *, 'far potential', farpot(1)
c
print 116,icount

format(' total newt iterations',i5)
c
return
end

subroutine bigtrm(big, biggst)

implicit double precision (a-h,o-z)
dimension big(4)

biggst = 0.d+00
do 100 i=1,4
   absb = dabs(big(i))
   if(absb.gt.biggst) biggst = absb
100 continue

return
end

subroutine fcns(sumo, suml, idec, csurf, jx, eta, fs, dfdc, big, xij, xi, etas)

run with

  current
  f
  dfdc
  iother
  irow
  for
  back
  delfcn

This subroutine calculates the functions that are needed to solve for the surface concentrations, and it calculates the partial current densities for the given guessed surface concentrations.

Input variables:

  idec = index for which decade
  csurf(ielec,i,jx) = dimensionless surface concentration of species i on electrode ielec at mesh point jx
  jx = current mesh point
  eta(ielec,jx,idec) = (V<ielec> - phi 0) (-n<m> F/(s<Rm> RT))
  diodr
  coeff
c       cref,porq,ip,iq,uref,nspec,nrxn,one

xj,ratson,ason,ifor,iback

xint0,xintl,n2

ian,icath

Output variables

fs(irow) = dimensionless equation for Fick’s Law flux minus Faraday’s Law flux
dfdcs(irow,jcol) = derivative of f with respect to dimensionless surface concentrations
big(l-4) = terms in equation f
xij(ielec,j,jx) = -n<m>/s<R,m> (FL/(RT K<b>)) i<j,ielec>

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

common/evprob/xlamda(3),ai(3),z,diodr(10)
common/co/coeff,xn
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),nspec,nrxn,one(2,5),rho0
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/cdata/cnctol,ftol,xint0(10,101,5),
xintl(10,101,5),n2,nd,isecc
common/elecs/jv(2),ian,icath
common/diagn/iterp
dimension csurf(2,10,101),eta(2,101,5),fs(20),etas(2,101),
dfdcs(20,20),prod(2),big(4),xij(2,5,101),sum0(2,10),
suml(2,10),tf(2,5,101),tb(2,5,101),bignew(4),xi(2,101)

n-jx-1
npl-jx

do 90 ib=1,4

90 big(ib) = 0.d+00
do 91 ielec=ian,icath
91 call curnt(ielec,csurf,eta,tf,tb,xij,xi,jx,idec,etas)
do 100 ielec=ian,icath
do 100 i=1,nspec
   call fsub(n,npl,ielec,i,jx,idec,csurf,sum0,sum1,
   tf,tb,f,bignew)
do 105 ib=1,4
if (dabs(bignew(ib)) .gt. dabs(big(ib))) big(ib) = bignew(ib)  
fs(irow(i,ie1ec)) = f  
do 110 irow(i,ie1ec) = f  
do 110 irow(i,ie1ec) = f  
do 110 irow(i,ie1ec) = f  
dfdcs(irow(ir,ie1ecr),irow(ic,ie1ec)) = dfdc(ir,ic,ie1ecr,  
     ielecc,jx,idec,csurf,tf,tb)  
print 111,irow(ir,ie1ecr),irow(ic,ie1ec),dfdc(irow(ir,ie1ecr),  
     irow(ic,ie1ec))  
111 format(1x,' dfdcs(',i2,',',i2,')=',1pe11.4)  
continue  
return  
end  
function dfdc(ir,ic,ie1ecr,ie1ecc,jx,idec,csurf,tf,tb)  
run with  
iother  
delfcn  
implicit double precision (a-h,o-z)  
common/evprob/xlambda(3),ai(3),z,diodr(10)  
common/co/coef,xn  
common/cdata/cnctol,ftol,xint0(10,101,5),  
     xint1(10,101,5),n2,nd,isecc  
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),  
     nspec,nrxn,one(2,5),rho0  
common/kinpars/xj(5),ratson(10,5),ascon(2,5),ifor,iback  
dimension csurf(2,10,101),tf(2,5,101),tb(2,5,101)  
iothr = iother(ie1ecr)  
sumj = 0.d+00  
if(ie1ecr.ne.ie1ecc) go to 99  
do 110 j=1,nrxn  
     sumj = sumj + ratson(ir,j)*(tf(ie1ecr,j,jx)*porq(ip,ic,j)/  
     csurf(ie1ecr,ic,jx) - tb(ie1ecr,j,jx)*porq(iq,ic,j)/  
     csurf(ie1ecr,ic,jx))  
1 if(j.eq.1) print *, ' f=',tf(ie1ecr,j,jx)*porq(ip,ic,j)/  
c     csurf(ie1ecr,ic,jx),' b=',tb(ie1ecr,j,jx)*porq(iq,ic,j)/  
c     csurf(ie1ecr,ic,jx)  
110 continue  
dfdc = delfcn(ir,ic)*coef*diodr(ir)*(-xint0(ir,1,idec)*
CHANEL.FOR

1 delfcn(ielecr,ielecc) + xintl(ir,lc,idec)
1 delfcn(iothr,ielecc))

print 112,dfdc,delfcn(ir,ic),delfcn(ielecr,ielecc),
1 delfcn(iothr,ielecc)

1 format('lx,' dfdc','1plell.4,' del(ir,ic)' ,lpe11.4,/,
1 ' del(ielecr,ielecc),1plell.4,' del(iothr,ielecc),
1 lplell.4)

100 continue

1 print *, ir',ir, ic',ic',iothr',iothr,' ielecc',ielecc
1 print 111,jx,ielecr,ielecc,ir,ic,sumj,dfdc

1 format(jx','i2,ielecr=' ,i2,' ielecc=' ,i2,ir','i2, ic','i2,
1 ' sumj=' ,lplell.4,' dfdc=' ,lplell.4)

1 return
1 end

function delfcn(i,j)

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

delfcn=0.d00

if (i.eq.j) delfcn=1.

return
end

subroutine newtlc(xiav,phistr,p0mps,eta,csurf,xij,xi,xitotc,
1 xitota,tx,fine,can,ccath,iprt,xitotj,txoj,son,limcur,etas)

subroutine newt for limiting current

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

logical limcur(2)

dimension csurf(2,10,101),eta(2,101,5),fs(20),tf(2,5,101),
1 tb(2,5,101),big(4),p0mps(2,101,5),xij(2,5,101),xi(2,101),
1 xintgd(101),xitot(2),save(2),sum0(2,10),suml(2,10),tox(2,101),
1 fine(2,5,101),can(5,10,101),xitotj(2,5),savej(2,5),
1 toxj(2,5,101),ccath(5,10,101),etas(2,101)

common/eveprob/xlamda(3),ai(3),z,diodr(10)
common/co/coeff,xn
common/geom/pi,hol,nj,ndec,icut
common/rxnpars cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
1 nspec,nrxn,one(2,5),rho
common/matcom/b(20,20),d(20,41)
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/cdata/cnctol,ftol,xint0(10,101,5),
xint1(10,101,5),n2,nd,isej
common/elecs/v(2),ian,icath
common/diagn/interp
if(iprt.ne.0) print 158
format( ' big printout goes here')
icount = 0
n2pl=n2+1
expmax=2.303d+00*290.d+00
damping factor if concentrations go negative
cdamp=1.0d-06
get currents at first mesh point
jx=1
idec = ndec
do 374 ielec=ian,icath
374 call curnt(ielec,csurf,eta,tf,tb,xij,xj,jx,idec,etas)
if(iprt.ne.0) print 375,jx,xi(ian,1),xi(icath,1)
375 format( ' jx=',i2,' ian=',lpe1.4,' icath=',lpe1.4)
kend=(nf-1)/icut + 1
njml=nj-1
do 260 ielec=ian,icath
tox(ielec,1)=0.d+00
xitot(ielec)=0.d+00
do 260 j=1,nrxn
toxj(ielec,1,j)=0.d+00
260 xitotj(ielec,j)=0.d+00
jbegin=2
do 115 idec=1,ndec
h=dfloat(icut)**(idec-ndec)/dfloat(njml)
do 170 jx=jbegin,nj
if(limcur(ian).and.son.gt.0.) go to 116
go to 117
116 ielim=ian
iolim=icath
if(limcur(icath).and.son.lt.0.) go to 118
   go to 123
118  ielim=icath
   iolim=ian
123  continue
do 124 i=1,nspec
   if(ratson(i,1).le.0.) go to 124
   limsp=i
   csurf(ielim,i,2)=0.
124  continue
   if((limcur(ian).and.son.lt.0.) .or. (limcur(icath).and.
   son.gt.0.)) print *,' newtlc not appropriate'
   if((limcur(ian).and.son.lt.0.) .or. (limcur(icath).and.
   son.gt.0.)) stop
   if(jx.eq.2) go to 179
   jxm2 = jx - 2
   jxml = jx - 1
   if(jx.eq.3) go to 186
do
180  i=1,nspec
   do 184 ielec=ian,icath
   1
184  csurf(ielec,i,jx) = .9.*csurf(ielec,i,jxml) -
   csurf(ielec,i,jxm2)
   if(i.eq.limsp) csurf(ielim,i,jx) = 0.
   if(csurf(iolim,i,jx).le.0.d+00) csurf(ielim,i,jx) =
   dexp( 2.*
   dlog(csurf(iolim,i,jxml)) - dlog(csurf(iolim,i,jxm2)) )
   istop=0
   if(csurf(iolim,i,jx).le.0.d+00) istop=1
   if(istop.eq.1) print 181,jx,i dec,csurf(ielim,i,jx),
   csurf(ielim,i,jx)
   format(' at jx=' ,i2,' decade' ,i2,
   ' cothr,celec=' ,lpell.4,lx,ell.4)
   if(istop.ne.1) go to 180
   call dump(phistr,xiav,xi,p0mps)
   print 183
   format(' stopped in newtlc because predicted conc neg.' )
   stop
180  continue
go to 179
186  do 185 i=1,nspec
   do 185 ielec=ian,icath
185  csurf(ielec,i,jx) = csurf(ielec,i,jxml)
c
179 continue
CHANNEL.FOR

do 172 i=1,nspec

do 173 ielec=ian,icath
   sum0(ielec,i)=0.d+00
   suml(ielec,i)=0.d+00
173
if(jx.eq.2) go to 172
n=jx-1
nml=n-1

do 171 k=1,nml
   kp1=k+1
   nmkp1=n-k+1
   do 171 ielec=ian,icath
      sum0(ielec,i) = sum0(ielec,i) + (csurf(ielec,i,kp1)
1 -csurf(ielec,i,k))*xint0(i,nmkl,idec)
      suml(ielec,i) = suml(ielec,i) + (csurf(ielec,i,kp1)
1 -csurf(ielec,i,k))*xintl(i,nmkl,idec)
   171
continue
iexit=0
itmax = 40
itmax = 10

do 130 iter=1,itmax

initialize current to zero for summing

if(isec.eq.0.d+00) go to 216
   do 217 ielec=ian,icath
      call curnt(ielec,csurf,eta,tf,tb,xij,xi,jx,idec,etas)
216
216 if(iprt.ne.0 .and. isec.ne.0) print 375,jx,xi(ian,jx),
1   xi(icath,jx)
if(isec.ne.0) go to 170
do 50 jc=1,n2
do 50 ir=1,n2
   b(ir,jc)=0.d+00
50
d(ir,jc)=0.d+00
do 51 jc=n2p1,nd
   d(ir,jc)=0.d+00
51
ibefor=0
if(iexit.ne.0) ibefor=1

if(iterp.eq.4 .and. jx.eq.2) print *, 'call fcnsfc(lims,io)'
call fcnsfc(limsp,iolim,sum0,sum1,idec,csurf,jx,eta,fs,b,
big,xij,xi,etas)

n2ml = n2 - 1
ndml = 2*n2ml + 1
do 100 irowi=1,n2ml

if(jx.eq.2) print *, ' fs(irowi)', irowi, fs(irowi)
d(irowi,1)=-fs(irowi)

100

if(nspec.eq.1) d(irow(1,iolim),1)=-fs(irow(1,iolim))/
b(irow(1,iolim),irow(1,iolim))

if(nspec.eq.1) go to 119
if(iterp.eq.4) print *, ' before call matinv, fs'
if(iterp.eq.4) print 112,(fs(irowi),irowi=1,n2)

112 format(4(1x,e11.4))

if(iterp.eq.4) print *, ' before call matinv'
do 122 i=1,n2
if(iterp.eq.4) print 112,(b(i,j),j=1,n2)

122 continue

if(iterp.eq.4) print *, ' can', (csurf(ian,i,jx),i=1,nspec)
if(iterp.eq.4) print *, ' ccath', (csurf(icath,i,jx),
i=1,nspec)
call matinv(n2ml,ndml,determ)

if(jx.eq.2) print *, ' d', (d(i,1),i=1,3)
if(determ.ne.0.d=00) go to 119
print 110,idec,jx

110 format(27h zero determinant in matinv', idec=' ,i2,
' jx=' ,i2)
do 121 i=1,n2

121 print 113,(b(i,j),j=1,n2)

113 format(' b matrix',lx,lpe1.4,3(lx,ell.4))
continue
stop

119 iexit=0
do 120 i=1,nspec

do 120 ielec=ian,icath
if(i.eq.limsp .and. ielec.eq.ielim) go to 120
ishift=0
if(ielec.eq.icath) ishift=nspec
irsm=1+ishift
if(irsm.ge.irow(limsp,ielim)) irsm = irsm - 1
if(ielec.eq.icath .and. i.eq.2) print *, ' irsm', irsm
if((d(irsm,l)+csurf(ielec,i,jx)).le.O.d+0.0d+00) go to 150

CHANNEL.FOR

csurf(ielec,i,jx)=d(irsm,l)+csurf(ielec,i,jx)

if((dabs(d(irsm,l)/csurf(ielec,i,jx)).ge.cnctol)) go to 219
if(dabs(d(irsm,l)/csurf(ielec,i,jx)).ge.cnctol .and.
csurf(ielec,i,jx).gt.cnctol) print *,
'd/c.ge.cnctol, go to 219,d,csurf',d(irsm,l),
csurf(ielec,i,jx)
if((dabs(d(irsm,l)/csurf(ielec,i,jx)).ge.cnctol .and.
csurf(ielec,i,jx).le.cnctol) .or. cdamp>0)
if((dabs(d(irsm,l)/csurf(ielec,i,jx)).ge.cnctol .and.
csurf(ielec,i,jx).gt.cnctol)) go to 219
go to 120

150
csurf(ielec,i,jx)=csurf(ielec,i,jx)*cdamp
if((dabs(d(ir,l)/csurf(ielec,i,jx)).ge.cnctol)) go to 219
iexit=1
120 continue
if(iterp.eq.4) print *,
'make sure it converges by N-R method and not by damping
1
162 format(' newtlc took',i3,' iterations for dec',i2,' jx=',i2)

163 continue
icount = icount + iter
if(iter.ge.20) print 162,iter,idec,jx

if(iter.ge.20 .and. jx.gt.3) stop

go to 170

CHANNEL.FOR

130 continue

c
140 if(itmax.ne.1) print 140,itmax,idec,jx,csurf(ian,1,jx),
1      csurf(icath,1,jx)
150 format(' no convergence in newtlc after',i4,
1      ' iterations at decade',i2,'jx=',i3,' can, cath=',1pel1.4,
1      1x,ell.4)
160 if(itmax.eq.1) go to 170
170 call dump(phistr,xiav,xi,pomps)
170 continue

c
180 call dump(phistr,xiav,xi,pomps)
170 continue

c
190 call dump(phistr,xiav,xi,pomps)
200 continue

c
210 continue

c
220 continue

c
230 continue

c
240 continue

c
250 continue

c
260 continue

c
270 continue

c
280 continue

c
290 continue

c
300 continue

c
310 continue

c
320 continue

c
330 continue

c
340 continue

c
350 continue

c
360 continue

c
370 continue

c
380 continue

c
390 continue

c
400 continue

c
410 continue

c
420 continue

c
430 continue

c
440 continue

c
450 continue

c
460 continue

c
470 continue

c
480 continue

c
490 continue

c
500 continue

c
510 continue

c
520 continue

c
530 continue

c
540 continue

c
550 continue

c
560 continue

c
570 continue

c
580 continue

c
590 continue

c
600 continue

c
610 continue

c
620 continue

c
630 continue

c
640 continue

c
650 continue

c
660 continue

c
670 continue

c
680 continue

c
690 continue

c
700 continue

c
710 continue

c
720 continue

c
730 continue

c
740 continue

c
750 continue

c
760 continue

c
770 continue

c
780 continue

c
790 continue

c
800 continue

c
810 continue

c
820 continue

c
830 continue

c
840 continue

c
850 continue

c
860 continue

c
870 continue

c
880 continue

c
890 continue

c
900 continue

c
910 continue

c
920 continue

c
930 continue

c
940 continue

c
950 continue

c
960 continue

c
970 continue

c
980 continue

c
990 continue

c
1000 continue

c
1010 continue

c
1020 continue

c
1030 continue

c
1040 continue

c
1050 continue

c
1060 continue

c
1070 continue

c
1080 continue

c
1090 continue

c
1100 continue

c
1110 continue

c
1120 continue

c
1130 continue

c
1140 continue

c
1150 continue

c
1160 continue

c
1170 continue

c
1180 continue

c
1190 continue

c
1200 continue

c
1210 continue

c
1220 continue

c
1230 continue

c
1240 continue

c
1250 continue

c
1260 continue

c
1270 continue

c
1280 continue

c
1290 continue

c
1300 continue

c
1310 continue

c
1320 continue

c
1330 continue

c
1340 continue

c
1350 continue

c
1360 continue

c
1370 continue

c
1380 continue

c
1390 continue

c
1400 continue

c
1410 continue

c
1420 continue

c
1430 continue

c
1440 continue

c
1450 continue

c
1460 continue

c
1470 continue

c
1480 continue

c
1490 continue

c
1500 continue

c
1510 continue

c
1520 continue

c
1530 continue

c
1540 continue

c
1550 continue

c
1560 continue

c
1570 continue

c
1580 continue

c
1590 continue

c
1600 continue

c
1610 continue

c
1620 continue

c
1630 continue

c
1640 continue

c
1650 continue

c
1660 continue

c
1670 continue

c
1680 continue

c
1690 continue

c
1700 continue

c
1710 continue

c
1720 continue

c
1730 continue

c
1740 continue

c
1750 continue

c
1760 continue

c
1770 continue

c
1780 continue

c
1790 continue

c
1800 continue

c
1810 continue

c
1820 continue

c
1830 continue

c
1840 continue

c
1850 continue

c
1860 continue

c
1870 continue

c
1880 continue

c
1890 continue

c
1900 continue

c
1910 continue

c
1920 continue

c
1930 continue

c
1940 continue

c
1950 continue

c
1960 continue

c
1970 continue

c
1980 continue

c
1990 continue

c
2000 continue

c
2010 continue

c
2020 continue

c
2030 continue

c
2040 continue

c
2050 continue

c
2060 continue

c
2070 continue

c
2080 continue

c
2090 continue

c
2100 continue

c
2110 continue

c
2120 continue

c
2130 continue

c
2140 continue

c
2150 continue

c
2160 continue

c
2170 continue

c
2180 continue

c
2190 continue

c
2200 continue

c
2210 continue

c
2220 continue

c
2230 continue

c
2240 continue

c
2250 continue

c
2260 continue

c
2270 continue

c
2280 continue

c
2290 continue

c
2300 continue

c
2310 continue

c
2320 continue

c
2330 continue

c
2340 continue

c
2350 continue

c
2360 continue

c
2370 continue

c
2380 continue

c
2390 continue

c
2400 continue

c
2410 continue

c
2420 continue

c
2430 continue

c
2440 continue

c
2450 continue

c
2460 continue

c
2470 continue

c
2480 continue

c
2490 continue

c
2500 continue

c
2510 continue

c
2520 continue

c
2530 continue

c
2540 continue

c
2550 continue

c
2560 continue

c
2570 continue

c
2580 continue

c
2590 continue

c
2600 continue

c
2610 continue

c
2620 continue

c
2630 continue

c
2640 continue

c
2650 continue

c
2660 continue

c
2670 continue

c
2680 continue

c
2690 continue

c
2700 continue

c
2710 continue

c
2720 continue

c
2730 continue

c
2740 continue

c
2750 continue

c
2760 continue

c
2770 continue

c
2780 continue

c
2790 continue

c
2800 continue

c
2810 continue

c
2820 continue

c
2830 continue

c
2840 continue

c
2850 continue

c
2860 continue

c
2870 continue

c
2880 continue

c
2890 continue

c
2900 continue

c
2910 continue

c
2920 continue

c
2930 continue

c
2940 continue

c
2950 continue

c
2960 continue

c
2970 continue

c
2980 continue

c
2990 continue

c
3000 continue

if(idec.eq.ndec) go to 115
do 12 k=1,kend

CHANNEL.

jj = k*icut - icut + 1

do 12 ielec=ian,icath
    xi(ielec,k) = xi(ielec,jj)
    etas(ielec,k) = etas(ielec,jj)
    tox(ielec,k) = tox(ielec,jj)

do 14 j=1,nrxn
    toxj(ielec,k,j) = toxj(ielec,jj,j)

14
    xij(ielec,j,k) = xij(ielec,j,jj)

do 12 i=1,nspec

12
    csurf(ielec,i,k) = csurf(ielec,i,jj)

jbegin=kend+1

continue

xitotc=xitot(icath)

xitota=xitot(ian)

return

end

subroutine fcnslc(limsp, iolim, sumO, suml, idec, csurf, jx, eta, fs,
1 dfdcs, big, xij, xi, etas)

subroutine fcns for limiting current

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

logical same

common/evprob/xlamda(3),ai(3),z,diodr(10)
common/co/coeff,xn
common/rxnpars/ref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
1 nspec,nrxn,one(2,5),rho0
common/knkpars/xj(5),ratson(10,5),asen(2,5),ifor,iback
common/cdata/cntol,ftol,xint0(10,101,5),
1 xintl(10,101,5),n2,nd,isee
common/elecs/jv(2),ian,icath
common/diagn/iterp

dimension csurf(2,101,5),eta(2,101,5),fs(20),etas(2,101),
1 dfdcs(20,20),prod(2),big(4),xij(2,5,101),sum0(2,10),
1 suml(2,10),tf(2,5,101),tb(2,5,101),bignew(4),xi(2,101)

c

ielim=iother(iolim)
n=jx-1
npl = jx

CHANNEL.FOR

do 90 ib = 1, 4
90  big(ib) = 0.d0

if (iterp.eq.4 .and. jx.eq.2) print *, ' call currnt(iolim)'
call currnt(iolim, csurf, eta, tf, tb, xij, xi, jx, idec, etas)

if (iterp.eq.4 .and. jx.eq.2) print *, ' call fsub(ielim, limsp)',
1  ' tf', tf(1, 1, jx), tf(1, 2, jx), ' tb', tb(1, 1, jx), tb(1, 2, jx)
call fsub(n, npl, ielim, limsp, jx, idec, csurf, sum0, sum1, tf, tb, f, bignew)
  if (jx.eq.2) print *, ' ielim, limsp'
c  if (jx.eq.2) print 378, jx, (bignew(ii), ii = 1, 4), bignew(1)
  + bignew(2), bignew(3) + bignew(4)
  format(' jx', i2, ' big(i)=' , 4el2.4, /, ' 1+2,3+4' , 2el2.4)
print *, ' limsp', limsp
do 115 j = 1, nrxn
  print *, ' j', j, ', ratson', ratson(limsp, 1)
  xij(ielim, j, jx) = -(bignew(l) + bignew(2))/ratson(limsp, 1)
  xi(ielim, jx) = xij(ielim, j, jx)
115  if (iterp.eq.4 .and. jx.eq.2) print *, ' tf, tb change:'
do 95 j = 1, nrxn
  tf(ielim, j, jx) = xij(ielim, j, jx)
  tb(ielim, j, jx) = 0.
95  continue

do 100 i = 1, nspec
do 100  ielec = ian, icath
  if (i.eq.limsp .and. ielec.eq.ielim) go to 100
call fsub(n, npl, icath, i, jx, idec, csurf, sum0, sum1, tf, tb, f, bignew)
    if (iterp.eq.4 .and. jx.eq.2) print *, ' call fsub, ielec',
      ielec, ' species', i
  if (jx.eq.2) print 378, jx, (bignew(ii), ii = 1, 4), bignew(1)
  + bignew(2), bignew(3) + bignew(4)
do 105 ib = 1, 4
105  if (dabs(bignew(ib)) .gt. dabs(big(ib))) big(ib) = bignew(ib)
    ismall = irow(i, ielec)
    if (ismall.ge.irow(limsp, ielim)) ismall = ismall - 1
    fs(ismall) = f
100  continue

do 110 ielecc = ian, icath
do 110  irel = 1, nspec
110  continue
**CHANNEL.FOR**

```fortran
irmsm = irow(ir,ielecr)
icsm = irow(ic,ielecc)

if(irmsm.eq.irow(limsp,ielim) .or. icsm.eq.irow(limsp,ielim))
   go to 110
   if(irmsm.ge.irow(limsp,ielim)) irsm = irsm - 1
   if(icsm.ge.irow(limsp,ielim)) icsm = icsm - 1
dfdcs(irms,icsm) = dfdclc(ielim,ir,ic,ielecr,ielecc,jx,idec,csurf,tf,tb)
110 continue

c return
c end

c subroutine currnt(ielec,csurf,eta,tf,tb,xij,xi,jx,idec,etas)
  run with
  for
  back

  implicit double precision (a-h,o-z)
  common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
     nspec,nrnx,one(2,5),rho0
  common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
  common/elec:v(2),ian,icath
  common/diag./interp

  dimension tf(2,5,101),tb(2,5,101),xij(2,5,101),csurf(2,10,101),
     eta(2,101,5),xi(2,101),etas(2,101)

  xj(ielec,jx) = 0.d+00

do 100 j=1,nrnx
  call for(csurf,eta,ielec,j,jx,idec,tf(ielec,j,jx),etas)
  call back(csurf,eta,ielec,j,jx,idec,tb(ielec,j,jx),etas)
  xij(ielec,j,jx) = tf(ielec,j,jx) - tb(ielec,j,jx)
  xi(ielec,jx) = xi(ielec,jx) + xij(ielec,j,jx)
100 continue

  return
  end

c subroutine for(csurf,eta,ielec,j,jx,idec,tf,etas)

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

common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),nspec,nrxn,one(2,5),rho0

CHANNEL.FOR

common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/diagn/iterp

c
dimension csurf(2,10,101),eta(2,101,5),etas(2,101)
c
prod = 1.d+00
c
do 120 k=1,nspec
   if(csurf(ielec,k,jx).eq.0..and. porq(ip,k,j).eq.0.) go to 120
   if(csurf(ielec,k,jx).ne.0.) go to 118
   prod = 0. go to 120
118 if(cref(k).lt.1.0d-15) go to 119
   prod = prod*(csurf(ielec,k,jx)/cref(k))**porq(ip,k,j)
   go to 120
119 prod = prod*(csurf(ielec,k,jx)/rho0)**porq(ip,k,j)
   continue
c
if(j.eq.1) etas(ielec,jx) =
1   eta(ielec,jx,idec)-uref(j)
   eterm=dexp(ason(ifor,j)*(eta(ielec,jx,idec)-uref(j)))
   tf = one(ielec,j)*prod*xj(j)*eterm
c
return
end
c
c subroutine back(csurf,eta,ielec,j,jx,idec,tb,etas)
c
implicit double precision (a-h,o-z)
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),nspec,nrxn,one(2,5),rho0
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/diagn/iterp

c
dimension csurf(2,10,101),eta(2,101,5),etas(2,101)
c
prod = 1.d+00
c
do 120 k=1,nspec
   if(csurf(ielec,k,jx).eq.0..and. porq(iq,k,j).eq.0.) go to 120
   if(csurf(ielec,k,jx).ne.0.) go to 118
   prod = 0. go to 120
118 if(cref(k).lt.1.0d-15) go to 119
   prod = prod*(csurf(ielec,k,jx)/cref(k))**porq(iq,k,j)
go to 120
prod = prod*(csurf(ielec,k,jx)/rhoO)**porq(iq,k,j)
CHANNEL.FOR

120 continue
if(j.eq.1) etas(ielec,jx) =
1 eta(ielec,jx,idec)-uref(j)
eterm=dexp(-ason(iback,j)*(eta(ielec,jx,idec)-uref(j)))
tb = one(ielec,j)*prod*xj(j)*eterm
return
end

subroutine fsub(n,np1,ielec,i,jx,idec,csurf,sumO,suml,tf,tb,f,big)
run with
iother
implicit double precision (a-h,o-z)
common/evprob/lambda(3),ai(3),z,diodr(10)
common/co/coeff,xn
common/cdata/cnctol,ftol,xint0(10,101,5),
xint1(10,101,5),n2,nd,isec
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,uref(5),
spec,nrxn,one(2,5),rhoO
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
common/diagn/iterp
dimension csurf(2,10,101),tf(2,5,101),tb(2,5,101),sumO(2,10),
suml(2,10),big(4)
iothr=iother(ielec)
sumlf = -(csurf(ielec,i,np1)-csurf(ielec,i,n))*xint0(i,1,idec)
1 -sumO(ielec,i)
sum2f = (csurf(iothr,i,np1)-csurf(iothr,i,n))*xintl(i,1,idec)
1 +suml(iothr,i)
big(1) = coeff*diodr(i)*sumlf
big(2) = coeff*diodr(i)*sum2f
big(3)=0.d+00
big(4)=0.d+00

do 100 j=1,nrxn
big(3) = ratson(i,j)*tf(ielec,j,jx) + big(3)
big(4) = -ratson(i,j)*tb(ielec,j,jx) + big(4)
100 continue

CHANNEL. FOR

f = big(1) + big(2) + big(3) + big(4)
return
end

function dfdclc(ielim,ir,ic,ielecr,ielecc,jx,idec,csurf,tf,tb)
function dfdc for limiting current
implicit double precision (a-h,o-z)
common/evprob/xlamda(3),ai(3),z,diodr(10)
common/co/coef,xf
common/cdata/cnctol,ftol,xint0(10,101,5),
in(xintl(10,101,5),n2,nd,ise)
common/ncpars/crel(10),cre(10),porq(2,10,5),ip,iq,uref(6),
nspec,nx,one(2,5),rho
common/kinpars/xj(5),ratson(10,5),ason(2,5),ifor,iback
dimension csurf(2,10,101),tf(2,5,101),tb(2,5,101)
iothr = .iother(ielecr)

if(csurf(ielecc,ic,jx).eq.0.) print *,' ielecc,ic,jx',ielecr,
  ic,jx
sumj = 0.d+00
if(ielecr.ne.ielecr .or. ielecr.eq.ielimin) go to 99
  do 110 j=1,nx
  sumj = sumj + ratson(ir,j)*(tf(ielecr,j,jx)*porq(ip,ic,j)/
  csurf(ielecr,ic,jx) - tb(ielecr,j,jx)*porq(iq,ic,j)/
  csurf(ielecr,ic,jx))
110 continue
99 dfdclc = delfcn(ir,ic)*coffe*diodr(ir)*(-xint0(ir,1,idec)*
delfcn(ielecr,ielecc) + xintl(ir,1,idec)*
delfcn(iothr,ielecc))
100 continue

return
end

function irow(i,ielec)
implicit double precision (a-h,o-z)
common/ncpars/crel(10),cre(10),porq(2,10,5),ip,iq,uref(6),
nspec,nrxn,one(2,5),rho0
common/elecs/v(2),ian,icath

CHANNEL.FOR

if(ielec.eq.ian) irow = i
if(ielec.eq.icath) irow = i + nspec
return
end

SUBROUTINE GAUSSQ(INTEGR,XI,JBEGIN,IDEC,SAVEA,SAVEC,TOX)

Gaussian quadrature integration
integr = 1 is for integrating current
integr = 2 is for integrating to find \phi(-\infty)

implicit double precision (a-h,o-z)
dimension xi(2,10l),xl(101),x2(101),yl(101),y2(101),tox(2,101)
common/geom/pi,hol,nj,ndec,icut
common/gauss/xp(96),gw(96),ngt,xielec(201),
1 xiothr(201),njmore,xx(201)
common/elecs/v(2),ian,icath

h=dfloat(icut)**(idec-ndec)/dfloat(nj-1)
j s = jbegin
if(jbegin.eq.2) js = 1
savea = 0.d0
savec = 0.d0

set up limits of integration
nf = nj
if(integr.eq.1) nf = js
do 577 n = nj,nf,-1
do 577 ielec=ian,icath
print *,'electrode',ielec
xmin1 = dfloat(js-1)*h
xmax1 = dfloat(n-1)*h
xmin2 = xmax1
jxm = n
do 572 jx = js,n
print *,'diff',dabs(xi(ielec,jx))-dabs(xi(ielec,jx+1))
if((idec.lt.ndec .and.
1 dabs(xi(ielec,jx)) - dabs(xi(ielec,jx+1)) .gt. 0.)
1 .or. (idec.eq.ndec .and .jx.lt.n .and.
1 dabs(xi(ielec,jx)) - dabs(xi(ielec,jx+1)).gt.0.) ) then
   go to 572
else
\[
\text{xmax1} = \text{dfloat(jx-1)\text{*}h} \\
\text{xmin2} = \text{xmax1}
\]

CHANNEL FOR

\[
\text{jxm} = \text{jx} \\
\text{go to 573}
\]

end if

572 continue

573 \text{xmax2} = \text{dfloat(n-1)\text{*}h} \\
\text{c print 100,xmin1,xmax1,xmin2,xmax2,jxm} \\
\text{c 100 format('xmin1',lpe1.4,'xmax1',lpe1.4,'xmin2',lpe1.4,} \\
\text{c 1 'xmax2',lpe1.4,'jxm',12)} \\
\text{c nx1 = jxm - js + 1} \\
\text{nx2 = n - jxm + 1} \\
\text{do 574 jx = js,jxm} \\
\text{x1(jx+1-js) = dfloat(jx-1)\text{*}h} \\
574 \text{y1(jx+1-js) = xi(ielec,jx)} \\
\text{do 575 jx = jxm,n} \\
\text{x2(jx+1-jxm) = dfloat(jx-1)\text{*}h} \\
575 \text{y2(jx+1-jxm) = xi(ielec,jx)} \\
\text{save1 = 0.d0} \\
\text{save2 = 0.d0} \\
\text{c print *,'nx1',nx1,'nx2',nx2} \\
\text{c print *,'xprml fl flx xprm2 f2 f2x'} \\
\text{do 576 jxp = 1,ngt} \\
\text{xprml = (xmax1-xmin1)*xp(jxp) + xmin1} \\
\text{xprm2 = (xmax2-xmin2)*xp(jxp) + xmin2} \\
\text{call interp(1,x1,y1,xprml,f1,nx1,3)} \\
\text{call interp(0,x2,y2,xprm2,f2,nx2,3)} \\
\text{c if(integr.eq.2) print 578,xprml,f1,f1*dsqrt} \\
\text{c 1 (xprml)*(xprml-0.5),xprml2,} \\
\text{c 1 f2,f2*dsqrt(xprml)*(xprml2-0.5)} \\
\text{c 578 format(6(lpe1.4,lx))} \\
\text{if(integr.eq.1) then} \\
\text{save1 = save1 + f1*dsqrt(xprml)*gw(jxp)} \\
\text{save2 = save2 + f2*dsqrt(xprml)*gw(jxp)} \\
\text{else} \\
\text{save1 = save1 + f1*dsqrt(xprml)*(xprml-0.5)*gw(jxp)} \\
\text{save2 = save2 + f2*dsqrt(xprml2)*(xprml2-0.5)*gw(jxp)} \\
\text{end if}
\]

576 continue

\text{save1 = save1*(dsqrt(xmax1) - dsqrt(xmin1))} \\
\text{save2 = save2*(dsqrt(xmax2) - dsqrt(xmin2))} \\
\text{if(ielec.eq.ian.and.n.eq.nj) savea = save1 + save2} \\
\text{if(ielec.eq.icath.and.n.eq.nj) savec = save1 + save2} \\
\text{if(integr.eq.1) tox(ielec,n) = save1 + save2} \\
\text{c if(integr.eq.2) print *,'elec',ielec,'savea',savea,'savec',savec} \\
\text{if(integr.eq.2) savea = savea + savec} \\
\text{c if(integr.eq.2) print *,'save1',save1,'save2',save2}
577 continue

return

end

SUBROUTINE INTERP(ileft,x,y,xj,f,nx,m)

Lagrangian interpolation subroutine adapted from Parrish
y(x)*sqrt(x) is assumed to be linear in sqrt(x)
v = y sqrt(x), u = sqrt(x)
y is the array of known values at the evenly-spaced x points
y(zj) = f is sought. nx is the number of evenly-spaced x points
(x ranges between 0 and 1)

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

dimension y(101),x(101),u(101),v(101)

if(ileft.eq.1) then
    zj = dsqrt(xj)
else
    zj = dsqrt(1.-xj)
end if

print *, ' u  v  x  y'
do 100 ix=1,nx
    if(ileft.eq.1) then
        u(ix) = dsqrt(x(ix))
v(ix) = y(ix) * u(ix)
   else
        u(1+nx-ix) = dsqrt(1.-x(ix))
v(1+nx-ix) = y(ix) * u(1+nx-ix)
   end if
continue
print 110,((u(ix),v(ix),x(ix),y(ix)),ix=1,nx)
format(4(1pe11.4,1x))

kp = 1
n = m + 1
ne = 1
if( (n/2)*2 .eq. n) ne = 0

do 20 k = kp,nx
    if(u(k)-zj) 20,60,24
```
CHANNEL.FOR

20  continue

22  jjs = nx - n + 1
    jjf = nx
    go to 30

24  kp = k
    jjs = kp - n/2 - ne
    jjf = kp + n/2 - 1
    if (jjf .ge. nx) go to 22
    if (jjs .gt. 0) go to 30
    jjs = 1
    jjf = n

30  f = 0.d0
    if (jjs lt 1) jjs = 1
    if (jjf gt nx) jjf = nx
    print *, 'jjs', jjs, 'jjf', jjf
    do 50 k = jjs, jjf
        c = 1.d0
        do 40 kk = jjs, jjf
            if (kk .eq. k) go to 40
            c = c * (zj - u(kk)) / (u(k) - u(kk))
        40          continue
    50          f = f + c*v(k)
    print *, 'raw f', f, 'zj', zj
    go to 80

60  f = y(k)
    return

80  f = f/zj
    return
end

subroutine totcurr(jbegin,h,f,save, tox)

implicit double precision (a-h,o-z)
dimension save(2), f(2,101), xintgd(101), tox(2,101)

common/geom/pi, hol, nj, ndec, icuf
common/elecs/v(2), ian, icath

this subroutine integrates the function f (call is inside idec loop

do 95 ielec = ian, icath
  print 94, ielec, save(ielec)
```
CHANNEL.

kend=(nj-1)/icut + 1
jb=jbegin
npts=nj-1
if(jbegin.eq.2) go to 367
jb=jbegin-1
npts=nj-kend+1
367 continue

do 463 ielec=ian,icath
   do 363 jx=jb,nj
      jj=jx-jb+l
      print 93,ielec,jj,f(ielec,jx)
      format(' elec',i2,' jj=',i2,' xintgd=',lpell.4)
      xintgd(jj)=f(ielec,jx)
   print *, ' tox=', tox(ielec,jb)
   call simps(xintgd,npts,h,save(ielec),ielec,jb,tox)
   print *, ' save=', save(ielec)
   continue
   if(jbegin.ne.2) return
   do 100 ielec=ian,icath
      tox(ielec,1) = 0.d+00
   print 465
   c 465 format(' in totcurr initializing tox(1)')
   gl=f(ielec,1)
   g2=f(ielec,2)
   if(gl.eq.0.d+00 .or. g2.eq.0.d+00) go to 373
   if(gl.ne.g2 .and. gl*g2/dabs(gl*g2).gt.0.d+00) go to 372
   save(ielec)=save(ielec) + h/2.*(gl+g2)
   tox(ielec,2) = h/2.*(gl+g2)
   do 98 jx=3,nj
      tox(ielec,jx)=tox(ielec,jx) + h/2.*(gl+g2)
      go to 99
   continue
   save(ielec)=save(ielec) + 1.5*h*g2
   tox(ielec,2) = 1.5*h*g2
   do 97 jx=3,nj
      tox(ielec,jx)=tox(ielec,jx) + 1.5*h*g2
      continue
   print 466
   c 466 format(' in totcurr initializing tox(2)')
   continue
   return
end
subroutine simps(f,n,h,result,ielec,jb,tox)

This routine performs Simpson's rule integration of a function defined by an array of equispaced values. This routine is adapted from Curtis F. Gerald, Applied Numerical Analysis, 2nd edition.

THIS ROUTINE HAS BEEN MODIFIED TO HANDLE SMALL N

Parameters are -
- f array of values of the function
- n number of points
- h the uniform spacing between x values
- result estimate of the integral that is returned to caller

implicit double precision (a-h,o-z)
dimension f(101),tox(2,101)

Check to see if number of panels is even. Number of panels is n-1

```
npanel=n-1
nhalf=npanel/2
nbeg=1
result=0.d+00
save=tox(ielec,jb)
tox(ielec,jb) = 0.d+00
print 302,jb
format(' in simps initializing tox(',i2,')')
```

if((npanel-2*nhalf).eq.0) go to 5

Number of panels is odd. Use 3/8 rule on first three panels, 1/3 rule on the rest of them.

```
if(npanel.eq.1) go to 100
result=3.*h/8.*(f(1) + 3.*f(2) + 3.*f(3) + f(4))
tox(ielec,jb+3) = result
tox(ielec,jb+2) = (2./3.) * result
tox(ielec,jb+1) = (1./3.) * result
print 300,jb+1,jb+2,jb+3
```
c 300 format(' in simps initializing tox(,3i2,)')
if(n.ne.4) go to 6

CHANNEL.FOR

tox(ielec,jb)=save
return

6 nbegin=4

apply 1/3 rule - add in first, second, last values.
result=result + h/3.*(f(nbegin) + 4.*f(nbegin+1) + f(n))
nbegin=nbegin+2
if(nbegin.ne.n) go to 110
    tox(ielec,n+jb-1) = result
    tox(ielec,n+jb-2) = 0.5*(tox(ielec,n+jb-1)+tox(ielec,n+jb-3))
    print 301,n+jb-1,n+jb-2

301 format(' in simps initializing tox(,2i2,)')
tox(ielec,jb)=save
return

The pattern after nbegin+2 is repetitive. Get nend, the place to stop
110 nend=n-2

tox(ielec,nbegin+jb-1) = result + h/3.*(f(nbegin)-f(n))
tox(ielec,nbegin+jb-2) = 0.5*(tox(ielec,nbegin+jb-1)+
tox(ielec,nbegin+jb-3))
print 301,nbegin+jb-2,nbegin+jb-1

do 10 i=nbegin, nend, 2
    result=result + h/3.*((2.*f(i) + 4.*f(i+1))
    tox(ielec,i+jb+1) = result + h/3.*(f(i+2)-f(n))
    tox(ielec,i+jb) = 0.5*(tox(ielec,i+jb-1) + tox(ielec,i+jb+1))
    print 301,i+jb,i+jb+1
10 continue

tox(ielec,jb)=save
return

100 result=(f(1)+f(2))/2.*h
tox(ielec,jb+1) = result
print 302,jb+1
tox(ielec,jb)=save
return
end

SUBROUTINE NINTERP(x,y,xj,f,nx)
c y(x)*sqrt(x) is assumed to be linear in sqrt(x)
CHANNEL.FOR
v = y sqrt(x), u = sqrt(x)
c y is the array of known values at the x points
c y(xj) = f is sought. nx is the number of x points
(c x ranges between 0 and 1)
c implicit double precision (a-h,o-z)
c
dimension y(201),x(201)
c ileft = 1
if(xj.ge.0.5) ileft = 0
do 20 k = 1,nx
    kp = k
    if(x(k)-xj) 20,60,30
20 continue
30 if(kp.ne.nx .or. y(kp)/y(kp-1).le.1.) go to 40
   b = 1./( (y(kp)/y(kp-1))**2 -1.)
   a = y(kp)*dsqrt(b)
   f=a/dsqrt(b+ (x(kp)-xj)/(x(kp)-x(kp-1)))
   return
40 if(kp.ne.2 .or. y(1)/y(2).le.1.) go to 50
   b = 1./( (y(1)/y(2))**2 -1.)
   a = y(1)*dsqrt(b)
   f = a/dsqrt(b+xj/x(2))
   return
50 if(ileft.eq.1)then
   ul=dsqrt(x(kp-1))
   u0=dsqrt(x(kp))
   zj = dsqrt(xj)
else
   ul=dsqrt(1.-x(kp-1))
   u0=dsqrt(1.-x(kp))
   zj = dsqrt(1.-xj)
endif
   f=(y(kp-1)*ul*(u0-zj)+y(kp)*u0*(zj-ul))/(u0-ul)/zj
   return
60 f = y(k)
   return
end
SUBROUTINE EQFIVE(XOL,PHIAN,PHICATH,P2,P3,P5)
This subroutine calculates the dimensionless form of the
CHANNEL.FOR

((phi0) - (phi *))/i<avg> given by equation 5 in Parrish
and Newman, JES, 117, 43-48, 1970. The singularity in
the equation was eliminated by the method of Kantorovich
and Krylov

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

c
common/geom/pi,hol,nj,ndec,icuts
common/elecs/v(2),ian,icath
common/gauss/xp(96),gw(96),ngt,xielec(201),
1 xiothr(201),njmore,xx(201)

ielec=ian
iothr = icath

an = 0.5d0

if(xol.gt.0.d0 .and. xol.lt.1.d0) then
  xmin = xol/2.
  xmax = xmin + 0.5
  call ninterp(xx,xielec,xol,xie,njmore)
  call ninterp(xx,xiothr,xol,xio,njmore)
else
  xmin = 0.5
  xmax = xmin
  xie = 0.
  xio = 0.
end if

sumra = 0.d0
sumrb = 0.d0
sumrc = 0.d0

sumrf = 0.d0
sumrs = 0.d0
sumrt = 0.d0

do 180 iregn=1,4
  go to (101,102,103,104),iregn
101 xll = 0.d0
  xul = xmin**an
  go to 150
102 if(xol.ge.1.d0 .or. xol.le.0.d0) go to 180
  if(xol.ge.0.75) then
    xul = (1.-xmin)**an
    xll = (1.-xol)**an
  else
    xll = xmin**an
  end if
xul = xol**an
end if

CHANNEL.FOR

103 if(xol.le.0.d0 .or. xol.ge. 1.d0) go to 180
if(xol.le.0.25) then
   xll = xol**an
   xul = xmax**an
else
   xul = (1.-xol)**an
   xll = (1.-xmax)**an
end if

104 go to 150
xul = (1.-xmax)**an
xll = 0.d0

150 suma = 0.d0
sumb = 0.d0
sumc = 0.d0
cf sumf = 0.d0
c sums = 0.d0
c sumt = 0.d0
do 170 jxp=1,ngt
   xprm = xp(jxp)*(xul-xll) + xll
   z = xprm**(l./an)
   if (iregn.eq.3.and.xol.gt.0.25) .or. iregn.eq.4 .or. (iregn.eq.2.and.xol.gt.0.75) z = 1. z
   y = c*(xol-z)
   sinh2 = (dexp(y) - dexp(-y))*2/4.0
   if(dabs(y).lt.0.01) sinh2 = (y*(1. + y**2/6.*(1. + y**2/20.*(1. + y**2/42.))))**2
   cosh2 = 1. + sinh2
   if(y.ne.0) alnth2=dlog(sinh2/cosh2)
c
   ileft = 1
   if(z.gt.0.5) ileft = 0
   call ninterp(xx,xielec,z,aa,njmore)
call ninterp(xx,xiothr,z,bb,njmore)
c
   bb = bb + aa
   Y2 = xprm**(l./an - 1.) / an
   clnch2 = dlog(cosh2)*Y2
   c
   if(y.ne.0.) suma = suma + gw(jxp)*(aa*alnth2 - xie*dlog(y**2))*Y2
   if(y.ne.0.) sumb=sumb+gw(jxp)*((bb-aa)*alnth2-xio*dlog(y**2))*Y2
   sumc = sumc + gw(jxp)*bb*clnch2
cf sumf = sumf + gw(jxp)*bb*(z-0.5)*Y2
   sums = sums + gw(jxp)*bb*Y2
   sumt = sumt + gw(jxp)*clnch2

170 continue
sumra = suma/2.*(xul-xll) + sumra
sumrb = samb/2.*(xul-xll) + sumrb
sumrc = summar/2.*(xul-xll) + sumrc

cf sumrf = sumr/2.*(xul-xll) + sumr
sumrs = sumr/2.*(xul-xll) + sumr
sumrt = sumr/2.*(xul-xll) + sumr

c
180 continue

c
g2 = 0.d0
if(xol.lx 1.d0 .and. xol.gt.0.d0) g2 = dlog(c**2) +
1 2.*(1.-xol)*(dlog(1.-xol) - 1.) + 2.*xol*(dlog(xol) - 1.)

c
cf p1 = sumrf/(2.*h0)
p2 = -sumra/(2.*pi)
p3 = -(g2*xie+sumra)/(2.*pi)
p5 = -(sumrc-sumrs*sumrt)/(2.*pi)

phicath = -(g2*xio + sumrb)/(2.*pi)+p5
phian = -(g2*xie + sumra)/(2.*pi)+p5

c
return
end
Appendix G. Program NEWCHAN

G.1. Program Description

Program NEWCHAN solves for the current, concentration, and potential distribution in a channel flow cell by a method of collocation\textsuperscript{22} applied to the partial current densities. Current functions (four special functions plus a set of Legendre polynomials) are input to the linear equations — Faraday’s law, the superposition integrals for the flux, and Laplace’s equation — to produce concentrations and potentials for substitution into the Butler-Volmer equation. The program solves for the set of coefficients that minimizes the error between the series representation of the current and the Butler-Volmer current at the specified collocation points.

A variation of the method is used to improve convergence at the limiting current; instead of minimizing the current errors on the limiting electrode, we require the surface concentrations to be zero.

Another feature of the program is the continuation method, which may be used to improve the convergence. The continuation variable may be either the cell voltage or the exchange current density of one of the reactions. The cell voltage is the recommended continuation variable.

The program may be run with or without a restart by setting $\text{IRSTRT} = 1$ or $0$, respectively. In either case, a restart file is generated from every run. If the parameter $\text{IALL}$ is set equal to 0, the current and/or concentration errors are calculated only at the collocation points, but one can set $\text{IALL} = 1$ to provide the detailed distribution of errors at the $\text{NJ}$ evenly-spaced mesh points. This detailed information is needed to assess the accuracy of the results.

Upstream and downstream potentials may be calculated by adding extra mesh points. The points are 8 times more closely spaced in the intervals immediately before
and after the electrode edges.

*Input Variables:*

ICHAR = character string for identifying variables in the data file

NG = half the number of Gaussian quadrature points to be used in the integrals for the potential distribution (maximum allowable = 48)

ROOT(JXP) = roots for Gaussian quadrature (input the roots from 0 to 1)

GW(JXP) = Gaussian quadrature weights

XLAMDA(I) = eigenvalues from the asymmetric Graetz problem

AI(I) = coefficients from the asymmetric Graetz problem

NJ = number of evenly-spaced mesh points

NDEC = number of “decades” or regions with finely-spaced mesh points. Decades start from \( x=0 \) and are nested inside each other

ITSTRT = 0 if the run is not a restart, nonzero otherwise

HOL = aspect ratio, \( h/L \)

ICUT = specifies size of each “decade” of finely-spaced points. A decade starts from \( x=0 \) and ends at \( x=1/ICUT \)

\[
XN = N = \left( \frac{n_m^2} {s_{Rm}^2} \right) \frac{F^2 D_R c_{R, ref}}{(RT \kappa_{fced})} \left[ (6 v L^2 /(hD_R)) \right]^{1/3} = \text{dimensionless limiting current density}^1 
\]

PEHOL = \( Pe \ h/L = 2 <\nu> h^2 / (DL) = \) dimensionless parameter characterizing boundary layer thickness (directly proportional to Graetz number)

RHO0 = \( \rho_v / c_{R, ref} \) = solvent density / reference concentration of main reactant

VCELL = \( V_{cell}(-n_m F) / (s_{Rm} RT) \) = dimensionless cell voltage

VSTART = starting dimensionless cell voltage for continuation method (set VSTART = VCELL if the continuation method is not to be used)

^1 This interpretation holds only for thin boundary layers.
XJJ = starting dimensionless exchange current density for continuation method

JRXN : continuation may be applied to the exchange current of reaction JRXN

IVOLTS = nonzero if continuation is on cell voltage, zero otherwise

DELTAP = initial step size of continuation variable

ERRTST = relative convergence criterion for error in currents

MAXIT = maximum number of iterations for continuation method

IR = index for principal reactant

M = index for main reaction

ISEC = nonzero if running secondary current distribution

NSPEC = number of species

NRXN = number of reactions

NRE = index for reference-electrode reaction

UJTH(NRE) = dimensionless standard electrode potential of the reference-electrode reaction

ALPHA(1,J) = anodic transfer coefficient for reaction J

ALPHA(2,J) = cathodic transfer coefficient for reaction J

NSUBJ(J) = number of electrons transferred in reaction J

XJ(J) = \( J_j = -n_m FL_i_o/(s_{Rm}RT\kappa\infty) \) = dimensionless exchange current density for reaction J

UJTH(J) = \( U_j^f(-n_mF)/(s_{Rm}RT) \) = dimensionless standard electrode potential for reaction J

DIODR(I) = \( D_i/D_R \) = dimensionless diffusion coefficient of species I

CREF(I) = dimensionless reference concentration of species I

CRE(I) = dimensionless concentration of species I in the reference-electrode compartment

CIOCR(I) = \( c_{i,feed}/c_{R,ref} \) = dimensionless feed concentration of species I
SIJ(I,J) = stoichiometric coefficient of species I in reaction J
NPTS = 1 + number of equations on an electrode
ICOLL(IELEC,IC) = mesh-point index for collocation points on electrode IELEC
IA LL = 1 for computing the errors at all of the mesh points, 0 for calculating errors at
the collocation points only
DELT A1 = parameter for limiting-current trial function (see equation D-15)
DELT A2 = parameter for trial function given in equation D-16
DELT A3 = parameter for trial function given in equation D-17
NPEXTR = total number of extra mesh points at which potentials are calculated

Output Variables:
XIAV = dimensionless average current density
PHISTR = integration constant in solution to Laplace's equation for the potential
UREF(J) = dimensionless theoretical open-circuit potential for reaction J, evaluated at
the reference concentrations
PORQ(IP,I,J) = $p_{ij}$ = anodic reaction order of species I in reaction J
PORQ(IO,I,J) = $p_{ij}$ = cathodic reaction order of species I in reaction J
CSURF(IAN,I,JX) = dimensionless anode surface concentration of species I at mesh
point JX
XIJ(IAN,J,JX) = dimensionless anodic partial current density from reaction J at mesh
point JX
XI(IAN,JX) = $i_{an}(-n_{m}/s_{Rm})FL/(RT\kappa_{\infty})$ = dimensionless anodic current density
P0MPS(IAN,JX) = $(\phi_{an}^{*}(-n_{m}F)/(s_{Rm}RT))$ = potential calculated from Laplace's
equation
NITS = last iteration count
G.2. Major Subroutines

G.2.1. Subroutine GAUSSQ

This subroutine calculates by Gaussian-quadrature integration the total currents on both electrodes and, if desired, the potentials at $+\infty$ and $-\infty$. The integrals are divided into regions to improve the accuracy.

G.2.2. Subroutine INTXI

This subroutine calculates $XINT0$ (I, N-K+1, IDEC), which is

$$\frac{1}{\Delta z_{(k-1)}} \int_{1}^{k\Delta z} C_i(n\Delta z - x') dz',$$

where $C$ is the flux calculated from the asymmetric Graetz problem.

G.2.3. Subroutine ERR

Iterates to find the coefficients that minimize at the collocation points the errors between the Butler-Volmer currents and the approximate currents. The equation $<i>_{an} = <i>_{cath}$ is also included. If a "limiting electrode" (LIMELEC) is specified, the concentrations are set equal to zero on that electrode, the coefficient for the limiting-current trial function on the anode is set equal to zero, and the coefficient for the modified limiting-current function on the cathode is set equal to zero.

If the continuation method is being used, the subroutine is called twice for evaluating the derivatives with respect to the continuation variable. When the subroutine is called for the second time, there is no iteration on the coefficients.

Subroutines Required:

SUMIK
CFROMI
IKP (function subprogram)
Function IKP collapses three subscripts into one (K, J, and IELEC)

G.2.4. Subroutine EQFIVE

Calculates the dimensionless form of the \((\Phi^o - \Phi^i)\) given by equation 5 in Parrish and Newman\(^{21}\) (equation 1-6 in this work). The singularity in the equation was eliminated by the method of Kantorovich and Krylov\(^{64}\) as discussed in section 1.4.3. The subroutine is written to accept trial current functions to produce an array of potentials.

G.2.5. Subroutine CURRENT

Calculates the current from the Butler-Volmer equation (unless the surface concentration is zero) and evaluates the derivatives of the partial current densities with respect to the current coefficients. The derivatives of the concentrations and potentials with respect to the coefficients must be provided (they are calculated in subroutines CFROMI and EQFIVE, respectively).

G.2.6. Subroutine CFROMI

Calculates concentration functions from the current functions, using Faraday’s law (subroutine NFROMI) and the superposition integrals (subroutine CFROMN).

G.2.7. Subroutine CFROMN

Calculates concentrations from fluxes by solving the superposition-integral equation with the method of Acrivos and Chambre.\(^{59}\) The Stieltjes contributions are calculated separately (subroutines FLAK and FLCK). To improve the accuracy for the current-
step problem, a correction factor is added to provide the correct concentration at the second finely-spaced mesh point.
PROGRAM NEWCHAN

This program solves for the current, concentration, and potential distribution in a channel flow cell by a method of collocation. Current functions (three special functions plus a set of Legendre polynomials) are input to the linear equations -- Faraday's law, the superposition integrals, and Laplace's equation -- to produce the inputs for the Butler-Volmer equation, which gives the currents. The program solves for the set of coefficients that minimizes the error between the series representation of the current and the Butler-Volmer current at the specified collocation points. For problems that do not converge, the continuation method has been added. The continuation may be on either the cell voltage or the exchange current of a given reaction.

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

dimension csurf(2,10,101),xj(5),
1 xij(2,5,101),save(2),cik(40,40),xijnew(2,5,101),
1 xi(2,101),ujth(5),xijew(2,101),limsp(2),
1 root(96),error(2,5,101),delc(2,10,101),bdum(40,40),
1 xitotjk(41,2,5),array(2,101),csave(2,10,101),
1 nsubj(5),ratio(2),tryr(2),tox(2,101),xitot(2),
1 p0mps(2,101),xijka(41,5,404),xitotj(2,5),cord(41),
1 p0mpsk(41,2,101),v(2),errsav(2,5,101),
1 xijkc(41,5,404),cank(41,10,101),ccathk(41,10,101),
1 co(41,2,5),cold(41,2,5),delco(41,2,5),b(40,40),
1 d(40,3),dxij(41,5,101),dxijkc(41,5,101),
1 fluxak(41,10,404),pfine(41,2,229),ptemp(2,229)

c
common/evprob/xlambda(3),ai(3),z,diorl(10),ciocr(10)
common/co/coeef,xn
common/geom/pi,hol,nj,ndec,icut
common/rrnptes cref(10),cre(10),porg(2,10,5),ip,iq,
1 uref(5),npec,nrnx,rho0
common/rrpns/sij(10,5),ratson(10,5),ason(2,5),ifor,iback
common/cdata/xint0(10,101,5),xint1(10,101,5),n2,nd,isec
c
common/elecs/ian,icath
common/gauss/xp(96),gw(96),ngt,xx(201),xielec(201),xiochr(201),njmore
common/colloc/npts,icol(2,41),npeextr,npj
1 common/equalc/ikboth
common/params/deltel,delta2,delta3

data pi/3.14159265358979d+00/, ian/l/, icath/2/, ip/l/,
1 iq/2/, ifor/l/, iback/2/
call readin(xn,xj,ujth,nsubj,ng,root,nre,m,irstrt,vcell, \\
  vstart,jrxn,xjj,ivolts,deltap,errtst,maxit,iread)

nptstot = 2*(npts-1)*nrxn + 1
ncik = 2*(npts-1)

print *, ' points on anode', (icoll(ian,ik),ik=2,npts)
print *, ' points on cathode', (icoll(icath,ik),ik=2,npts)

n2 = 2*nspec
nd = 2*n2+1
coeff = -xn*(16.*z)**(-1./3.)
njml = nj - 1

Rearrange the Gaussian roots so they are in the interval
0 to 1 instead of -1 to 1.

ngt = 2*ng

do 10 jxp=1,ngt
10 xp(jxp) = (root(jxp)+l.)/2.

do 40 i=1,nspec
do 40 j=1,nrxn
  if(sij(i,j).gt.0.d0) then
    porq(ip,i,j) = sij(i,j)/nsubj(j)
    porq(iq,i,j) = 0.d0
  end if
  if(sij(i,j).lt.0.d0) then
    porq(ip,i,j) = 0.d0
    porq(iq,i,j) = -sij(i,j)/nsubj(j)
  end if
  if(sij(i,j).eq.0.d0) then
    porq(ip,i,j) = 0.d0
    porq(iq,i,j) = 0.d0
  end if
  print 30,i,j,porq(ip,i,j),porq(iq,i,j)
30 format(lx,' p(',i2,',,i2,')=',1pel1.4,' q=',1pel1.4)
40 continue

do 70 j=1,nrxn
  sum = 0.d0
  do 60 i=1,nspec
    if(ratson(i,j).eq.0.d0) go to 50
    if(cref(i) .eq. 0.d0) then
      print *, ' error in cref(i),i=',i,'j=',j
      stop
      continue
  end if
  print *, ' error in cref(i),i=',i,'j=',j
  stop
end if
sum = sum + ratson(i,j)*dlog(cref(i)/rhoO)

50 if(ratson(i,nre) .eq. 0.d0) go to 60
if(cref(i) .eq. 0.d0) then
  print *, 'error in cref(i), i=' , i, 'j=' , j
  stop
end if
sum = sum - ratson(i,nre)*dlog(cre(i)/rhoO)

60 continue
print *, 'j=', j, 'ujth(j), ujth(nre)'
uref(j) = ujth(j) - ujth(nre) + sum
print *, 'j=', j, 'ujref=' , uref(j)

70 continue

call intcxi(nspec,xint0,xint1)
c
istop=0
if(istop.ne.0) stop
v(icath) = 0.d0
v(ian) = vcell
c
phistr = vcell*ason(ian,1) / (ason(ian,1)+ason(icath,1))
1 + v(icath)
print *, 'first phistr=', phistr
print *, 'x current function'
do 100 idec=1,ndec
  h = dfloat(icut)**(idec-ndec)/dfloat(nj-1)
do 100 jx=1,nj
    jxfine = ifine(idec,jx)
x = (jx-1)*h
  do 100 j=1,nrxn
    do 100 ik=2,npts
      xijka(ik,j,jxfine) = curfcn(ik,ian,x)
      xijkc(ik,j,jxfine) = -xijka(ik,j,jxfine)
      if(ik.eq.npts) print 98,x,(xijka(ikd,j,jxfine),ikd=2,npts)
  98 format(6(1pe11.4,1x))
do 100 continue
ic
istop = 0
if(istop.ne.0) stop
do 140 ik = 2,npts
do 140 j = 1,nrxn
c
call gaussq(ik,xitot(ian),xitot(icath),dwnstr)
c
do 140 ielec=ian,icath
  xitotjk(ik,ielec,j) = xitot(ielec)
print 249, dwnstr, xitot(ian), xitot(icath)
249 format(‘downstream potential’, el2.4,’ xitota’, el2.4,’ xitotc’,
1 el2.4)
140 continue

continue

itmax = 20

limelec = 2
do 175 ic=2,npts
175 cord(ic) = 0.0
do 180 ik=1,npts
do 180 j=1,nrxn
180 delco(ik,ielec,j) = 0.0
p0 = vstart
pj0 = xjj
phistr = p0*ason(ian,1)/(ason(ian,1)+ason(icath,1))

phistr = 456.0
print *, ‘phistr=’,phistr,’p0’,p0,’ason(ian,1)’,ason(ian,1)
cold(1,ian,1) = phistr
cold(2,ian,1) = phistr
delco(2,ian,1) = 0.0
cold(2,icath,1) = 0.0
delco(2,icath,1) = 0.0
cold(2,icath,1) = 0.0

choose coefficients to give total current zero

sum = 0.0
do 185 ik=2,npts
185 if(ik.eq.2.and.ielec.eq.ian) go to 185
sum = sum + co(ik,ielec,1)*xitotjk(ik,ielec,1)
cold(1,ian,1) = -sum/xitotjk(2,ian,1)
cold(2,ian,1) = co(2,ian,1)

if (istop .ne. 0) stop

if(irstrt .ne. 0) then
read (4,190)
190 format(a72)

print *, ‘coeffs read in’
read (4,350),cold(1,1,1),((cold(ik,ielec,j),ielec=1,2)
1 ,ik=2,npts),j=1,nrxn))
print 350,cold(1,1,1),((cold(ik,ielec,j),ielec=1,2)
!NEWCHAN.FOR

1 ,ik=2,npts),j=1,nrxn))
read (4,190)
print *, 'delco read in'
read (4,350), (delco(1,1,1), ((delco(ik,ielec,j),
1 ielec=1,2), ik=2,npts), j=1,nrxn)
print 350, (delco(1,1,1), ((delco(ik,ielec,j),
1 ielec=1,2), ik=2,npts), j=1,nrxn)
print *, 'pOmpsk read in'
read (4,190)
do 200 j=1,nrxn
do 200 ik=2,npts
do 200 ieleck=ian,icath
do 200 jx=1,nj+28+npxextr
xol = xpri(jx,jreal)
read(4,360), (pfine( ikp(ik,ieleck,j), ielec, jx),
1 ielec=ian,icath)
if(jreal.ge.1 .and. jreal.1e.nj) then
pOmpsk(ikp(ik,ieleck,j),ian,jreal) =
1 pfine(ikp(ik,ieleck,j),ian,jx)
pOmpsk(ikp(ik,ieleck,j),icath,jreal) =
1 pfine(ikp(ik,ieleck,j),icath,jx)
end if
print 360, (pfine( ikp(ik,ieleck,j), ielec, jx),
1 ielec=ian,icath)
do 200 continue
read (4,190)
print *, 'conc functions read in'
do 210 j=1,nrxn
do 210 ik=2,npts
do 210 ieleck=ian,icath
do 210 i=1,nspec
do 210 jx=1,nj
read(4,360), cank( ikp(ik,ieleck,j), i, jx),
1 ccathk( ikp(ik,ieleck,j), i, jx)
print 360, cank( ikp(ik,ieleck,j), i, jx),
1 ccathk( ikp(ik,ieleck,j), i, jx)
do 210 continue
end if
do 340 iter=1,maxit
p0pd = 0.d0
pjpd = 0.d0
matcalc = 1
print *, 'try cell voltage', p0, 'deltap', deltap
print *, 'try exch current', pj0, 'deltap', deltap, 'errtst',
1 errtst
call err(cord, iter, ivolts, p0, p0pd, xj, pj0, pjpd, jrxn, error,
ERRSav, MATCALC, B, NITS, XIJKa, XIJKc, XIj, XI, CSURF,
PHDIST, XITOTJK, V, P0MPS, P0MPSK, PFINE, PTEMP, DELCO, COLD, CO, CANK,
CCATHK, NCIK, NPTSTOT, IISTRRT, ERRTST, IERR, IREAD)

PRINT *, NITS, 'iterations'
REWIND 7
WRITE(7, 220)
220 FORMAT('coefficients')
WRITE (7, 350), (CO1D(1, 1, 1), ((CO1D(1, 1, 1),
1 IELEC=1, 2), IK=2, NPTS), J=1, NRxn)
WRITE(7, 230)
230 FORMAT('delco')
WRITE (7, 350), (DELCO(1, 1, 1), ((DELCO(1, 1, 1),
1 IELEC=1, 2), IK=2, NPTS), J=1, NRxn)
WRITE(7, 240)
240 FORMAT('potential derivatives')
DO 250 J=1, NRxn
DO 250 IK=2, NPTS
DO 250 IELEC=1AN, ICATH
DO 250 JX=1, NJ+28+NPEXTR
WRITE(7, 360), (PFINE( IKP(1K, IELEC, J),
1 IELEC, JX), IELEC=1AN, ICATH)
250 CONTINUE
WRITE (7, 264)
264 FORMAT('concentration derivatives')
DO 270 J=1, NRxn
DO 270 IK=2, NPTS
DO 270 IELEC=1AN, ICATH
DO 270 I=1, NSPEC
DO 270 JX=1, NJ
WRITE(7, 360), CANK( IKP(1K, IELEC, J),
1 I, JX), CCATHK( IKP(1K, IELEC, J), I,
1 JX)
270 CONTINUE
WRITE (7, 271), NITS
271 FORMAT(i3)
C
IF(IERR.NE.0) THEN
PRINT *, 'no convergence in err'
GO TO 370
END IF
ISTOP=0
IF(ISTOP.NE.0) STOP
IF(I10VTS.EQ.1) THEN
IF(P0.GE.VCELL) THEN
PRINT *, 'continuation method converged, p0= ', P0,
'vcell',vcell
    go to 370
end if
else
  if(pj0.ge.xj(jrxn)) then
    print *, 'continuation method converged, pj0=',pj0,
    'xj',xj(jrxn)
    go to 370
  end if
end if

matcalc=0
if(ivolts.eq.1) then
  pOpd = pO + deltap*1.d-06
  print *, 'cell voltage',pO+deltap*1.d-06
else
  pjpd = pjO + deltap*1.d-06
  print *, 'exch current',pjO+deltap*1.d-06,'errtst',
  1 'errtst'
end if

call err(cord,iter,ivolts,pO,pOpd,xj,pj0,pjpd,jrxn,error,
  1 errsav,matcalc,bdum,nits,xijka,xijkc,xij,xi,csurf,
  1 phistr,xitotjk,v,p0mps,p0msk,pfine,ptemp,delco,cold,co,cank,
  1 ccathk,ncik,nptstot,irstrt,errtst,ierr,iread)

do 280 j=1,nrxn
  do 280 ik=1,npts
    do 280 ielec=ian,icath
      cold(ik,ielec,j) = co(ik,ielec,j)
      do 300 ielec=ian,icath
        do 300 j=1,nrxn
          do 300 ic=2,npts
            d( ikp(ic,ielec,j),-1 ) = -(error(ielec,j,icoll(ielec,ic))-
            1 errsav(ielec,j,icoll(ielec,ic)))/1.d-06
          300 continue
        d(1,1) = 0.d0
        print *, 'b matrix for continuation'
        print 320,((b(i,j),j=2,nptstot),i=2,nptstot)
        print *, 'row 1'
        print 310,(b(i,1),j=1,nptstot)
        print *, 'column 1'
        print 310,(b(1,j),j=1,nptstot)
        format(/,9(1x,lpel0.3))
        format(8(1x,lpel0.3))
        print *, 'd vector for continuation'
        print 320,(d(i,1),i=1,nptstot)
        call matinv(nptstot,nptstot,1,b,d,determ)
print *, 'd vector after continuation'
print 320, (d(i,1), i=1, nptstot)
if(determ.eq.0.d0) then
   print *, ' zero det in matinv for new p'
   stop
end if
do 330 j=1, nrxn
   do 330 ik=1, npts
      do 330 ielec=ian, icath
         delco(ik,ielec,j) = d( ikp(ik,ielec,j), 1 )
         if(nits.lt.3) deltap = deltap*3.
         if(nits.gt.3) deltap = deltap*0.56
         if(ivolts.eq.1) then
            po = po + deltap
            if(po.gt.vcell) po = vcell
         else
            pj0 = pj0 + deltap
            if(pj0.gt.xj(jrxn)) pj0 = xj(jrxn)
         end if
      end do
   end do
330 continue
print *, 'continuation method not converged'
350 format(lx, lpel5.8)
360 format(2(lx, lpel5.8))
370 call prtout(co,
            1 xij, xi, csurf, xiav, phistr, xitotjk, v, p0mps, ptemp)
stop
end

FUNCTION CURFCN(IFCN, IELEC, X)
if(ifcn.eq.2) then
   zeta = x*diodr(1)/z
   curfcn = coeff*diodr(1)*(flak(zeta) - flck(zeta))
   curfcn = -coeff*diodr(1)*flck(zeta)
end if

if(ifcn.eq.3) then
   curfcn = (x+deltal)**-1./3.
end if

if (ifcn.eq.4) then
   curfcn = 1.d0/dsqrt(x + delta2/(1./hol + 1.))
end if

if (ifcn.eq.5) then
   curfcn = 1.d0/dsqrt(1. - x + delta3/(1./hol + 1.))
end if

if(ifcn.ge.6) then
   do 90 ik-6,npts
      if(ifcn.eq.ik) then
         curfcn = p(ik-6,x)
      end if
   90 continue
end if

if(ielec.eq.icath) curfcn = -curfcn
return
end

FUNCTION IOTHER(I)
  this function is used for generating the index of the 'other' electrode
  iother = 3 - i
  return
end

SUBROUTINE PRTOUT(CO,XIJ,XI,CSURF,XIAV,PHISTR,1 XITOTJK,V,POMPS,PTEMP)
  This subroutine prints the final results
  implicit double precision (a-h,o-z)
dimension csurf(2,10,101), xij(2,5,101),
1  ptemp(2,101),
1  xi(2,101), xitotjk(41,2,5),
1  v(2), p0mps(2,101), co(41,2,5)
c
common/rxnpars/cref(10), cre(10), porq(2,10,5), ip, iq,
1  uref(5), nspec, nrxn, rho0
common/evprob/xlamda(3), ai(3), z, diodr(10), ciocr(10)
common/co/coefficient, xn
common/geom/pi, hol, nj, ndec, icut
common/elec/ion, icath
common/gauss/xp(96), gw(96), ngt, xx(201), xielec(201), xiothr(201), njmore
common/collcolloc/npts, icoll(2,41), npextr, np(7)
common/params/deltal, delta2, delta3
c
print 100, coeff, xiav, phistr
100 format(/,' coeff=',1pe11.4,' iavg=',1pe11.4,
1  ' phistr=',1pe11.4)
c
do 140 j=1,nrxn
print 130, j, uref(j)
130 format(' j=',i2,' ujref=',1pe11.4)
do 170 i=1,nspec
print 150, i, (j, porq(1,i,j), j=1, nrxn)
150 format(' i=',i2,5(' j=',i2,' pij=',1pel1.4))
do 160 i=1, (j, porq(2,i,j), j=1, nrxn)
160 format(' i=',i2,5(' j=',i2,' qij=',1pel1.4))
do 200 i=1,nspec
print 180, i
write(8,180), i
180 format(/,' species',i3,/', x ccath can')
do 200 jx=1,nj
x=dfloat(jx-1)/dfloat(nj-1)
write(8,190), x, csurf(2,i,jx), csurf(1,i,jx)
190 format(1x,1pe9.2,1x,1pe11.4,2x,1pe11.4)
do 230 j=1,nrxn
print 210, j
210 format(/,' reaction',i2, '/
1   x xicj xiaj ')
x=dfloat(jx-1)/dfloat(nj-1)
print 220,x,xij(2,j,jx),xij(1,j,jx)
220 format(1x,lpe9.2,lx,lpe9.2,lx,lpe9.2,lx,lpe9.2)
230 continue
c
print 390
390 format(/,
1 ' x     ian    icath    p0mpsa    p0mpsc'

c
print *,’potentials’
do 420 jx=1,nj+28+npeextr
   print 400,xprt(jx,jreal),ptemp(1,jx),ptemp(2,jx)
400 format(1x,0f8.4,lx,lpe9.2,lx,lpe9.2)
420 continue
c
print *,’partial currents’
do 430 j=1,nrxn
   print *,’reaction’ ,j
   do 430 jx=l,nj+14
      if(jx.le.9) xol = dfloat(jx-1)/dfloat(nj-1)/8.
      if(jx.gt.9.and.jx.le.nj+6) xol = dfloat(jx-8)/dfloat(nj-1)
      if(jx.gt.nj+6) xol-
1 + dfloat(jx-(nj+6))/dfloat(nj-1)/8.
   ani = 0.d0
   cathi = 0.d0
   do 425 ik=2,npts
      ani = ani + co(ik,ian,j)*curfcn(ik,ian,xol)
      cathi = cathi + co(ik,icath,j)*curfcn(ik,icath,xol)
425 continue
c
print 400,xol,ani,cathi
430 continue
c
return
end
c
SUBROUTINE CURRENT(XIJOLD,XIKA,XIJKC,IALL,YNEX,PHISTR,
1 XITOTC,XITOTA,POMPS,CSURF,CANK,CCATHK,XIJ,XI,
1 XITOTJK,POMPSK,DXIJKA,DXIJKC,V,XJ)
c
Calculates the current from the Butler-Volmer equation
c
implicit double precision (a-h,o-z)
c
dimension csurf(2,10,101),cank(41,10,101),
1 ccathk(41,10,101),xij(2,5,101),xi(2,101),p0mp(2,101),
1 eta(2,101),xitotjk(41,2,5),xitot(2),save(2),savej(2,5),

NEWCHAN.FOR

1 p0mpsk(41,2,10l),dprodf(41),dprodb(41),xj(5),
1 dxik(41,2,10l),dxijka(41,5,10l),dxijkc(41,5,10l),
1 dfor(41),dback(41),def(41),deb(41),v(2),
1 xijold(2,5,10l),xijka(41,5,404),xijkc(41,5,404)

c common/geom/pi,hol,nj,ndec,icut
common/kinpars/sij(l0,5),ratson(l0,5),ason(2,5),ifor,iback
common/rxnparsjcref(lO),cre(lO),porq(2,10,5),ip,iq,
1 uref(S),nspec,nrxn,rhoO
common/elecs/ian,icath
common/coll/coll/npts,icoll(2,4l),npextr,np(7)

c njml = nj - 1
kend = njml/icut + 1
expmax = 2.303d+00*270.d0

c if(iall.eq.0) then
  jx1 = 2
  jx2 = npts
else
  jx1 = 1
  jx2 = nj
end if

do 120 ic=jx1,jx2
do 120 ielec=ian,icath
if(iall.eq.0) then
  jx = icoll(ielec,ic)
else
  jx = ic
end if
eta(ielec,jx) = v(ielec) - (p0mpsk(ielec,jx) + phistr)
print 121,ielec,jx,eta(ielec,jx),phistr,p0mpsk(ielec,jx)
121 format('ielec',i2,'jx',i2,'eta',lpell.4,'phistr' ,lpell.4, 'p0mpsk',lpell.4)
c
do 110 j=-l,nrxn
  if(dabs(ason(ifor,j)*eta(ielec,jx)) .ge. expmax .or.
  1 dabs(ason(iback,j)*eta(ielec,jx)) .ge. expmax) then
  print 100
  100 format(' stopped in current because exp blows up')
  print *,'ielec',ielec
  print *,'jx','jx','eta',eta(ielec,jx),'v','v(ielec),
  'p0mpsk',p0mpsk(ielec,jx)
  stop
end if
110  continue

c  120 continue

c
   jbegin = 2
   h = 1./dfloat(njml)

do 280 ielec=ian,icath
   ilimc = 0
   if(iall.eq.0) then
      jx1 = 2
      jx2 = npts
   else
      jx1 = 1
      jx2 = nj
   end if
   do 280 ic=jx1,jx2
      if(iall.eq.0) then
         jx = icoll(ielec,ic)
      else
         jx = ic
      end if
   end do
   do 280 j=1,nrxn
      prodf = 1.d0
      prod2f = 1.d0
      prod2b = 1.d0
      prod2b = 1.d0
      do 160 k=1,nspec
         p = porq(ip,k,j)
         q = porq(iq,k,j)
         if(csurf(ielec,k,jx).eq.0.d0 .and. p.eq.0.d0) go to 140
         if(csurf(ielec,k,jx).gt.1.d-10) go to 130
         if(prodf.ne.0.d0) prdfoc=prodf
         prodf = 0.d0
         ilimc = 1
         go to 140
      end do
      130 continue
      prodf = prodf*(csurf(ielec,k,jx)/cref(k))**p
      prod2f = prod2f*(csurf(ielec,k,jx)/cref(k))**
         (p+ason(ifor,j)*ratson(k,j))
      140 continue
      if(csurf(ielec,k,jx).eq.0.d0 .and. q.eq.0.d0) go to 160
      if(csurf(ielec,k,jx).gt.1.d-10) go to 150
      if(prodb.ne.0.d0) prdboc = prodb
      ilimc = 1
      prodb = 0.d0
go to 160
prodb = prod2b*(csurf(ielec,k,jx)/cref(k))**q
prod2b = prod2b*(csurf(ielec,k,jx)/cref(k))**
(p+ason(ifor,j)*ratson(k,j))
continue

do 190 jk=1,nrxn
         do 190 ilec=Ian,icath
         do 190 ik=2,npts
            dprodf(ikp(ik,ieleck,jk)) = 0.d0
            dprodb(ikp(ik,ieleck,jk)) = 0.d0
         do 190 i=1,nspec
            p = porq(ip,i,j)
            q = porq(iq,i,j)
            ci = csurf(ielec,i,jx)
            if(ielec.eq.ian) dcidcok = cank(ikp(ik,ieleck,jk),i,jx)
            if(ielec.eq.icath) dcidcok = ccathk(ikp(ik,ieleck,jk),i,jx)
            if(p.eq.0.d0) go to 170
            if(ci.eq.0.d0) then
               print *, 'ci=0, ielec,i,jx',ielec,i,jx
               if(p.gt.1.d0) go to 170
               if(p.lt.1.d0) then
                  print *, 'c=0 and p<1 on elec',ielec,'spec',i
                  stop
               end if
               if(p.eq.1.d0) then
                  dprodf(ikp(ik,ieleck,jk)) = prdfoc*dcidcok
                  go to 170
               end if
            end if
            dprodf(ikp(ik,ieleck,jk)) = prod2f/ci * dcidcok
            + dprodf(ikp(ik,ieleck,jk))
         c
         continue
         if(q.eq.0.d0) go to 180
         c
         if(q.eq.0.d0) then
            if(q.gt.1.d0) go to 180
            if(q.lt.1.d0) then
               print *, 'c=0 and q<1 on elec',ielec,'spec',i
               stop
            end if
            if(q.eq.1.d0) then
               dprodf(ikp(ik,ieleck,jk)) = prdfob*dcidcok
               go to 170
            end if
            dprodf(ikp(ik,ieleck,jk)) = prdfob*dcidcok
            + dprodf(ikp(ik,ieleck,jk))
         end if
   end if
dprodb(ikp(ik,ieleck,jk)) = prodb*q/ci * dcidcok
1 + dprodb(ikp(ik,ieleck,jk))

180 continue
190 continue

etasjf = ason(ifor,j) * (eta(ielec,jx) - uref(j))
print *,'j,jx',j,jx,'etasjf',etasjf
etermf=dexp(ason(ifor,j) * (eta(ielec,jx) - uref(j)))

do 200 ieleck=ian,icath
do 200 ik=2,npts
do 200 jk=1,nrxn
200 def(ikp(ik,ieleck,jk)) = -ason(ifor,j)
1 *p0mpsk(ikp(ik,ieleck,jk),ielec,jx)*etermf

def(1) = -ason(ifor,j)*etermf
dprodf(1) = 0.d0
if(prod2f.eq.0.) then
print *,'prod2f=0'
go to 215
end if
if(dabs(etasjf/(xj(j)*prod2f)).le.0.01d0) then
print *, 'linear approx for forw term, elec',ielec,'jx',jx
etermf = (1. +
1 ason(ifor,j)*(eta(ielec,jx)-uref(j)))

do 210 ieleck=ian,icath
do 210 jk=1,nrxn
do 210 ik=2,npts
210 def(ikp(ik,ieleck,jk)) = -ason(ifor,j)
1 *p0mpsk(ikp(ik,ieleck,jk),ielec,jx)
def(1) = -ason(ifor,j)
end if
215 for = prodf*xj(j)*etermf
print *, 'elec',ielec,'jx',jx,'for',for

do 220 jk=1,nrxn
do 220 ieleck=ian,icath
do 220 ik=1,npts
   dfor(ikp(ik,ieleck,jk)) = xj(j) *
1 (prodf*def(ikp(ik,ieleck,jk))
1 + dprodf(ikp(ik,ieleck,jk))*etermf)
220 continue
etasjb = -ason(iback,j) * (eta(ielec,jx) - uref(j))
print *,'j,jx',j,jx,'etasjb',etasjb
etermb = dexp(-ason(iback,j) * (eta(ielec,jx) - uref(j)))
do 230 ieleck=ian,icath
do 230 jk=1,nrxn
do 230 ik=2,npts
230 deb(ikp(ik,ieleck,jk)) = ason(iback,j)
1 *p0mpsk(ikp(ik,ieleck,jk),ielec,jx)*etermb
de(1) = ason(iback,j)*etermb
dprodb(1) = 0.d0
if(prod2b.eq.0.) then
print *,'prod2b=0'
go to 245
end if
if(dabs(etasjb/(xj(j)*prod2b)).le.0.01d0) then
print *,'linear approx for back term, elec',ielec,'jx',jx
etermb = (1. -
1 ason(iback,j)*(eta(ielec,jx)-uref(j)))
do 240 ieleck=ian,icath
do 240 jk=1,nrxn
do 240 ik=2,npts
240 deb(ikp(ik,ieleck,jk)) = ason(iback,j)
1 *p0mpsk(ikp(ik,ieleck,jk),ielec,jx)
de(1) = ason(iback,j)
end if
245 back = prodb*xj(j)*etermb
print *,'elec',ielec,'jx',jx,'back',back
do 250 ieleck=ian,icath
do 250 jk=1,nrxn
do 250 ik=1,npts
dback(ikp(ik,ieleck,jk)) = xj(j) *
1 (prodb*deb(ikp(ik,ieleck,jk))
1 + dprodb(ikp(ik,ieleck,jk))*etermb)
250 continue
xij(ielec,j,jx) = for - back
if(ilimc .ne. 0 .and. ielec.eq.ian) xij(ielec,j,jx) =
xijold(ielec,j,jx)
if(ilimc .ne. 0 .and. ielec.eq.ian) print *,'lim curr on an'
if(ielec.eq.ian) then
do 260 ieleck=ian,icath
do 260 jk=1,nrxn
do 260 ik=1,npts
dxijka(ikp(ik,ieleck,jk),j,jx) = dfor(ikp(ik,
1 ielec,jk)) - dback(ikp(ik,ieleck,jk))
c if(ilimc.ne.0) dxijka(ikp(ik,ieleck,jk),j,jx) = xijka(c
 1 ik,j,ifine(ndec,jx))
260 continue
end if
if(ielec.eq.icath) then
  do 270 ieleck=ian,icath
    do 270 jk=1,nrxn
      do 270 ik=1,npts
        dxijkc(ikp(ik,ieleck,jk),j,jx) = dfor(ikp(ik,
           1 ieleck,jk)) - dback(ikp(ik,ieleck,jk))
      c if(ilimc.ne.0) dxijkc(ikp(ik,ieleck,jk),j,jx) = xijkc(c
      1 ik,j,ifine(ndec,jx))
      270 continue
    end if
    280 continue
  end do
  call sumi(xi,xij)
  do 290 ielec=ian,icath
    do 290 jk=1,nrxn
      do 290 ik=1,npts
290  call sumik(dxik,j,ikp(ik,ieleck,jk),ikp(ik,ieleck,jk),
                 1 dxijka,dxijkc,41,5,101,1)
  end do
  return
end

SUBROUTINE CFROMI(XIJKA,XIJKC,IK,J,IELECK,IKA,CANK,1 CCATHK)

Calculates concentration functions from the current functions

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

dimension cank(41,10,101),ccathk(41,10,101),
1 xijka(41,5,404),xijkc(41,5,404),fluxak(41,10,404),
1 fluxck(41,10,404)
c
common/evprob/xlamda(3),ai(3),z,diodr(10),ciocr(10)
common/co/coeff,xn
common/geom/pi,hol,nj,ndec,icut
common/rxnpars cref(10),cre(10),porq(2,10,5),ip,iq,
1 uref(5),nspec,nrxn,rho0
common/kinpars/sij(10,5),ratson(10,5),asen(2,5),ifor,iback
common/cdata/xint0(10,101,5),xintl(10,101,5),n2,nd,isecc
common/elects/ian,icath
common/gauss/xp(96),gw(96),ngt,xx(201),xielec(201),xiothr(20
SUBROUTINE NFROMI(XIJK, XIJKC, I, J, IKA, FLUXAK, FLUXCK)

Calculates flux from current using Faraday's law

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

dimension xijka(41,5,404), xijkc(41,5,404),
1 fluxak(41,10,404), fluxck(41,10,404)

call nfromi(xijka, xijkc, ik, j, ika, fluxak, fluxck)
call cfromn(fluxak, fluxck, cank, ccathk, ika, ieleck)
return
end

SUBROUTINE CFROMN(FLUXAK, FLUXCK, CANK, CCATHK, IK, IELECK)

Calculates concentrations from fluxes by solving the superposition-integral equation

implicit double precision (a-h,o-z)
dimension fluxak(41,10,404), fluxck(41,10,404),
1 cank(41,10,101), ccathk(41,10,101), sum0(2,10),
1 suml(2,10), b(40,40), d(40,1)

call cfromn(fluxak, fluxck, cank, ccathk, ika, ieleck)
return
end
common/elecs/ian,icath
common/rxn pars/cref(10),cre(10),porq(2,10,5),ip,iq,
1  uref(5),nspec,nrxn,rho0
common/kin pars/sij(10,5),ratson(10,5),ason(2,5),ifor,iback
common/geom/pi,hol,nj,ndec,icut
common/co/coeff,xn
common/evprob/xlamda(3),ai(3),z,diodr(10),ciocr(10)

cdamp=1.0d-06
n2 = 2*nspec
jbegin = 1
kend = (nj-1)/icut + 1

print *, 'in cfromn, ik=’,ik
do 100 jx=1, nj
do 100 i=1,nspec
  cank(ik,i,jx) = 0.d0
  ccathk(ik,i,jx) = 0.d0
100 if(isec.ne.0) return

do 190 idec=1,ndec
  h = dfloat(icut)**(idec-ndec)/dfloat(nj-1)
do 170 jx=jbegin,nj
  jxfine = ifine(idec,jx)
do 130 i=1,nspec
  zeta = (jx-1)*h*diodr(i)/z
do 110 ielec=ian,icath
    sum0(ielec,i) = 0.d0
 110 sum1(ielec,i) = 0.d0
  if(jx.1e.2) go to 130
  n = jx - 1
  nm1 = n - 1
  do 120 k=1,nm1
    kpl = k + 1
    nmkpl = n - k + 1
    sum0(ian,i) = sum0(ian,i) + (cank(ik,i,kpl)
     -cank(ik,i,k))**xint0(i,nmkpl,idec)
    sum0(icath,i) = sum0(icath,i) + (ccathk(ik,i,kpl)
     -ccathk(ik,i,k))**xint0(i,nmkpl,idec)
    sum1(ian,i) = sum1(ian,i) + (cank(ik,i,kpl)
     -cank(ik,i,k))**xint1(i,nmkpl,idec)
 120 sum1(icath,i) = sum1(icath,i) + (ccathk(ik,i,kpl)
     -ccathk(ik,i,k))**xint1(i,nmkpl,idec)
continue

c
irow = 0
do 140 ielec=ian,icath
   iother=iother(ielec)
do 140 i=1,nspec
   jcol = 0
   irow = irow + 1
do 140 jelec=ian,icath
do 140 jc=1,nspec

Correction factor to get accurate results for the
constant-current problem

corr = 1.0d0
if((jx.eq.2.and.idec.eq.1)) corr = 4.*pi/(9.*dsqrt(3.d0))
jcol = jcol + 1
b(irow,jcol) = 0.d0
if(irow.eq.jcol) then
   if((jx.eq.l)) b(irow,jcol) = -xint0(i,l,idec)
   *coeff*diodr(i)*corr
   if((jx.eq.1)) b(irow,jcol) = flck(zeta)*
   coeff*diodr(i)
else
   if((i.eq.jc.and.jx.ne.l)) b(irow,jcol) =
   xintl(i,1,idec)*coeff*diodr(i)
   if((i.eq.jc.and.jx.eq.l)) b(irow,jcol) =
   -flak(zeta)*coeff*diodr(i)
end if
if(ieleck.eq.ian) then
   flxak = fluxak(ik,i,jxfine)
   flxck = 0.d0
else
   flxak = 0.d0
   flxck = fluxck(ik,i,jxfine)
end if
if(ielec.eq.ian) then
   if((jx.eq.l)) then
      d(irow,1) = (+sum0(ielec,i) - cank(ik,i,jx-1)*
      xint0(i,1,idec)*corr - suml(iother,i) +
      ccathk(ik,i,jx-1)*xintl(i,1,idec))*coeff
      *diodr(i) + flxak + (-cank(ik,i,l)*flck(zeta)
      + ccathk(ik,i,l)*flak(zeta))*coeff*diodr(i)
   else
      d(irow,1) = flxak
   end if
else
   if((jx.ne.l)) then
NEWCHAN.FOR

\[
d(irow,l) = (+\sum_0(ielec,i) - ccathk(ik,i,jx-1)\times xint_0(i,l,idec)\times corr - \sum_0(iothr,i) + \\
cank(ik,i,jx-1)\times xintl(i,l,idec)\times coeff \\
\times diodr(i) + flxck + (-ccathk(ik,i,l)\times flck(zeta) \\
+ cank(ik,i,l)\times flak(zeta))\times coeff\times diodr(i)
\]

else
\[
d(irow,l) = flxck
\]
end if
end if

140 continue

145 format(4(lx,lpel1.4))
call matinv(n2,n2,1,b,d,determ)
c
if(determ.eq.0.d0) print *, ' zero det in matinv'
irow = 0
do 160 ielec=ian,icath
do 160 i=1,nspec
irow = irow + 1
if(d(irow,l).lt.-ciocr(i).and.d(irow,l).gt.-(ciocr(i) \\
+ 1.d-06)) d(irow,l) = -ciocr(i)
if(ielec.eq.ian) then
cank(ik,i,jx) = d(irow,l)
else
ccathk(ik,i,jx) = d(irow,l)
end if
160 continue

170 continue
if(idec.eq.ndec) go to 190
do 180 i=1,nspec
do 180 k=1,kend
jj = k*icut - icut + 1
cank(ik,i,k) = cank(ik,i,jj)
180 ccathk(ik,i,k) = ccathk(ik,i,jj)
jbegi = kend + 1
190 continue

200 continue

return
end

SUBROUTINE MATINV(L,N,M,b,d,DETERM)
c
This subroutine solves a matrix equation of the form 
BX = D. The answer X is put into the first column of D.
NEWCHAN.FOR

Input variables
n = dimension of square b matrix
m = number of columns in d matrix
b = matrix in equation bx=d
d : first column contains vector d in bx=d

Output variables
d : first column contains vector x in bx=d
detern = 0. if matrix has zero determinant

implicit double precision (a-h,o-z)
DIMENSION id(40),b(40,40),d(40,3)

DETERM=1.0
DO 1 I=1,L
1 ID(I)=0
DO 18 NN=1,N
BMAX=1.1
DO 6 I=1,N
IF(ID(I).NE.0) GOTO 6
BNEXT=0.0
BTRY=0.0
DO 5 J=1,L
IF(ID(J).NE.0) GOTO 5
IF(ABS(B(I,J)).LE.BNEXT) GOTO 5
BNEXT=ABS(B(I,J))
IF(BNEXT.LE.BTRY) GOTO 5
BNEXT=BTRY
BTRY=ABS(B(I,J))
JC=J
5 CONTINUE
IF(BNEXT.GE.BMAX*BTRY) GOTO 6
BMAX=BNEXT/BTRY
IROW=I
JCOL=JC
6 CONTINUE
IF(ID(JC).EQ.0) GOTO 8
DETERM=0.0
RETURN
8 ID(JCOL)=1
IF(JCOL.EQ.IROW) GOTO 12
DO 10 J=1,L
SAVE=B(IROW,J)
B(IROW,J)=B(JCOL,J)
10 B(JCOL,J)=SAVE
DO 11 K=1,M
SAVE=D(IROW,K)
D(IROW,K)=D(JCOL,K)
11 D(JCOL,K)=SAVE
12 F=1.0/B(JCOL,JCOL)
   DO 13 J=1,L
13 B(JCOL,J)=B(JCOL,J)*F
   DO 14 K=1,M
14 D(JCOL,K)=D(JCOL,K)*F
   DO 18 I=1,N
      IF(I.EQ.JCOL) GO TO 18
      F=B(I,JCOL)
   DO 16 J=1,L
16 B(I,J)=B(I,J)-F*B(JCOL,J)
   DO 17 K=1,M
17 D(I,K)=D(I,K)-F*D(JCOL,K)
18 CONTINUE
RETURN
END

SUBROUTINE SUMIK(XIK,J,IK,IAK,XIKA,XIJKC,L,M,N,IDEC)

Sums the partial-current-density functions

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

dimension xik(41,2,10l),xijka(1,m,n),xijkc(1,m,n)

common/geom/pi,hol,nj,ndec,icut
common/rxnpars cref(10),cre(10),porq(2,10,5),ip,iq,
    uref(5),nspec,nrxn,rh0
common/elecs/ian,icath

do 100 jx=1,nj
   jxfine = ifine(idec,jx)
   do 90 ielec=ian,icath
50 xik(ika,ielec,jx) = 0.
   xik(ika,ian,jx) = xik(ika,ian,jx) + xijka(ik,j,jxfine)
   xik(ika,icath,jx) = xik(ika,icath,jx) +
100 xijkc(ik,j,jxfine)
continue

return
end

FUNCTION XK(XM1)

Complete elliptic integral of the first kind
implicit double precision (a-h,o-z)

common/geom/pi,hol,nj,ndec,icut

a0 = 1.3862944d+00
a1 = 0.1119723d+00
a2 = 0.0725296d+00
b0 = 0.5d0
b1 = 0.1213478d+00
b2 = 0.0288729d+00

term = dlog(1./xml)
xk = a0 + a1*xml + a2*xml*xml + 
1 (b0 + b1*xml + b2*xml*xml)*term

return
end

SUBROUTINE SUMI(XI,XIJ)
Sums the partial current densities

implicit double precision (a-h,o-z)
dimension xi(2,101),xij(2,5,101)
common/geom/pi,hol,nj,ndec,icut
commom/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,
1 uref(5),nspec,nrxn,rhoO
commom/elecsjian,icath
do 100 ielec=ian,icath
do 100 jx=1,nj
   xi(ielec,jx) = 0.d0
do 100 j=1,nrxn
100 xi(ielec,jx) = xi(ielec,jx) + xij(ielec,j,jx)
return
end

FUNCTION IKP(IKIN,IELECIN,JIN)
Function for compressing three subscripts into one

implicit double precision (a-h,o-z)
SUBROUTINE INTCXI(NSPEC,XINT0,XINT1)

This subroutine calculates \( x_{\text{int}0}(i,n-k+1,\text{idec}) \), \( 1/dx \)
times the integral from \((k-1)dx\) to \(kdx\) of \( dC/dx_i\), where
C is the solution to the asymmetric Graetz problem in
channel flow and xi is the dimensionless normal variable, \( y/\eta \).

implicit double precision (a-h,o-z)
dimension xint0(10,101,5),xint1(10,101,5),array(5)

common/evprob/xlamda(3),ai(3),z,diodr(10)
common/geom/pi,hol,nj,ndec,icut

a = 1.35659745d0
b = 0.2d0
c = 0.060733452d0

aa = 0.9594d0
bb = 0.6069d0
cc = 0.4512d0
dd = 0.276d0

njml = nj - 1
n = njml

do 1000 idec=1,ndec
h = dfloat(icut)**(-ndec+idec)/dfloat(njml)
do 999 ix1=1,2
do 999 i=1,nspec
do 999 k=1,n
nmk = n-k
nmkpl = n-k+1
den = z/(h*diodr(i))
zeta = nmk/den
zetall = nmkpl/den
if(ixi .eq. 2) go to 500

xi = 0
if(zeta.lt.0.11d+00 .and. zetall.gt.0.11d+00) go to 600
if(zeta.ge.0.11d+00) go to 100

xintO(i,nmkpl,idec) = 1.5*den**(1./3.)*a*
1 (nmk**(2./3.) - (nmk+1)**(2./3.)) + b - 0.75*den**(-1./3.)*c*(nmk**(4./3.) -
1 (nmk+1)**(4./3.))
go to 999

sum - 0.0d0
do 110 j=1,3
110 sum = sum + dabs(ai(j))/x1amda(j) * (dexp(-x1amda(j)*zeta)
1 - dexp(-x1amda(j)*zetall))
xintO(i,nmkpl,idec) = -1. - 2.*den*sum
go to 999

xintO(i,nmkpl,idec) = 1.5*den**(1./3.)*a*(nmk**(2./3.) -
1 (0.11*den)**(2./3.)) + b*(0.11*den-nmk) -
1 0.75*den**(-1./3.)*c*
1 (nmk**(4./3.) - (0.11*den)**(4./3.))
sum = 0.0d+00
do 610 j=1,3
610 sum = sum + dabs(ai(j))/x1amda(j) * (dexp(-x1amda(j)*0.11)
1 - dexp(-x1amda(j)*zetall))
sum = 0.11*den - nmkpl - 2.*den*sum
xintO(i,nmkpl,idec) = xintO(i,nmkpl,idec) + sum
go to 999

xi = 1

if(zeta.le.0.18d+00 .and.zetall.ge.0.18d+00) go to 620
if(zeta.gt.0.18d+00) go to 510  
f0 = 0.d+00
if(n.eq.k) go to 505
f0 = -dexp(aa - bb/zeta - cc*dexp(-dd/zeta))
go to 999

505 xint1(i,nmkpl,idec) = (-dexp(aa - bb/zeta11 -
1 cc*dexp(-dd/zeta11)) + f0)/2.
continue

510 sum = 0.d0  
do 520 j=1,3
520 sum = sum + ai(j)/x1amda(j)*(dexp(-x1amda(j)*zeta) -
1 dexp(-x1amda(j)*zeta11))
xint1(i,nmkpl,idec) = -1. + 2.*den*sum 
go to 999

620 z11 = 0.18d0  
f0 = 0.d0
if(n.eq.k) go to 640
f0 = -dexp(aa - bb/zeta - cc*dexp(-dd/zeta))
go to 999

640 xint1(i,nmkpl,idec) = (-dexp(aa - bb/z11 -
1 cc*dexp(-dd/zll)) + f0)/2. * (-nmk + zll*den)
sum = 0.d0
do 660 j=1,3
660 sum = sum + ai(j)/xlamda(j)*(dexp(-xlamda(j)*0.18) -
1 dexp(-xlamda(j)*zetall))
sum = 2.*den*sum + zll*den - nmkpl  
xint1(i,nmkpl,idec) = xint1(i,nmkpl,idec) + sum
continue

999 h = h*dfloat(icut)
do 1000 continue

1000 continue

FUNCTION IFINE(IDECI,JXI)

Generates index for finely-spaced points

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

common/geom/pi,hol,nj,ndec,icut

ifine = 0
do 100 idec=1,ndec
do 100 jx=1,nj
   ifine = ifine + 1
100 if(idec.eq.ideci .and. jx.eq.jxi) return

print *, 'error in ifine, ideci=', ideci, 'jxi', jxi
stop
end

SUBROUTINE ERR(CORD,ITER,IVolts,VCell,VPDELTA,XJ,PJO,PJPD,
   JRXN,ERROR,ERRSAV,MATCALC,BSAVE,NITS,XIJKA,XIKJC,XIJ,
   XI,CSURF,SHISTR,XITOTJK,V,POMPS,POMPSK,PFINE,PTEMP,DELCO,
   COLD,CO,CANK,CCATHK,NCIK,NPTSTOT,IRSTRT,ERRTST,IERR,IREAD)

This program calculates the errors between the Butler-Volmer currents and the approximate currents at the collocation points.

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

dimension csurf(2,10,101),errsav(2,5,101),
   xij(2,5,101),cik(40,40),xijn(2,5,101),
   xi(2,101),xinew(2,101),limsp(2),xj(5),
   error(2,5,101),delc(2,10,101),cord(41),
   xitotjk(41,2,5),array(2,101),csave(2,10,101),
   bsave(40,40),xjdum(5),pfine(41,2,229),ptemp(2,229),
   pomps(2,101),xijka(41,5,404),xitotj(2,5),
   pompsk(41,2,101),v(2),
   xijkc(41,5,404),cank(41,10,101),ccathk(41,10,101),
   co(41,2,5),delco(41,2,5),b(40,40),
   d(40,3),dxijka(41,5,101),dxijkc(41,5,101)

c
    common/evprob/xlamda(3),ai(3),z,diodr(10),ciocr(10)
    common/co/coeff,xn
    common/geom/pi,hol,nj,ndec,icut
    common/rxnvars/cref(10),cre(10),porq(2,10,5),ip,iq,
    l uref(5),nspec,nrxn,rho0
    common/kinvars/sij(10,5),ratson(10,5),ason(2,5),ifor,iback
    common/cdata/xint0(10,101,5),xintl(10,101,5),n2,nd,isecc
    common/elecs/ian,icath
    common/gauss/xp(96),gw(96),ngt,xx(201),xielec(201),xiothr(20)
    njmore
    common/colloc/npts,icol(2,41),npextr,np(7)
    common/equalc/ikboth

c
    iitmax = 1
    if(matcalc.eq.1) iitmax = 20
    limelec = 2
NEWCHAN.FOR

V(icath) = 0.d0
V(ian) = Vcell

DO 100 j=1,nrxn
100 Xjdxm(j) = Xj(j)

Xjdxm(jrxn) = Pj0

Calculate initial concentration

IF(IRSTRT.EQ.0) THEN
   IEND = 1
ELSE
   READ(4,*) ,IEND,IALL
   PRINT *, 'IITSTART', IEND, 'IALL', IALL
ENDIF

IITF = IEND + IITMAX - 1
DO 440 IIT=IEND,IITF

PRINT *, 'ITERATION IIT', IIT
IF(MATCALC.EQ.0) GO TO 230
IF(ISEC.NE.0) THEN
   DO 110 IELEC=IAN,ICATH
      DO 110 J=1,NRXN
      DO 110 IK=1,NPTS
      110 CO(IK,IELEC,J) = COLD(IK,IELEC,J) + DELCO(IK,IELEC,J)
      PRINT *, 'COEFFICIENTS FROM DELCO AT 110'
      PRINT 120, CO(1,1,1)
      PRINT 120, ((CO(IK,IELEC,J), IELEC=1,2), IK=2,NPTS), J=1,NRXN
120 FORMAT(1X,1PE15.8)
   END IF

IF(IIT.EQ.1.AND.ITER.EQ.1) THEN

IF(IRSTRT.EQ.0) THEN
   J = 1
   PRINT *, 'CAN, CCATH CUR Fcn ON ANODE'
   PRINT *, 'POINTS', (ICOLL(IAN,IC), IC=2,NPTS)
   DO 140 IK=2,NPTS
      PRINT *, 'IK', IK
      CALL CFROMI(XIJK, XIJCK, IK, J, IELECK, IKP(IK,IELECK,J), CANK, CCATHK)
140 CONTINUE
   END IF

IF(IELECK.EQ.ICATH) PRINT 142,
((dfloat(jx-1)/dfloat(nj-1),cank(ikp(ik,1,1),1,jx),
c 1 ccathk(ikp(ik,1,1),1,jx)),jx=1,nj)
continue
format(3(lpell.4))
istop = 0
if(istop.ne.0) stop
print *,'potential fcns'
ikboth = 0
do 180 ik=2,npts
nextra = 0
do 203 jx=1,nj+28+npextr
nextra = nextra + 1
xol = xprt(jx,jreal)
call eqfive(ik,xol,phian,phicath,p2,p3,p5)
ptemp(ian,nextra) = phian
ptemp(icath,nextra) = phicath
print 181,xol,p2,p3,phicath,p5,
if(jreal.ge.1 .and. jreal.le.nj) pOmps(ian,jreal) = phian
if(jreal.ge.1 .and. jreal.le.nj) pOmps(icath,jreal) = phicath
181 format(0pf8.4,x,lp5el2.4)
continue
do 146 j=1,nrxn
do 146 ielec=ian,icath
  do 144 jx=1,nj
    pOmpsk(ikp(ik,ian,j),ielec,jx) = pOmps(ielec,jx)
do 143 nextra = 1,nj+28+npextr
143 pfine(ikp(ik,ian,j),ielec,nextra) = ptemp(ielec,nextra)
continue
do 147 j=1,nrxn
do 147 ielec=ian,icath
  do 152 jx=1,nj
    pOmpsk(ikp(ik,icath,j),ielec,jx) = pOmpsk(ikp(ik,ian,j),
      iother(ielec),jx)
  c if(ielec.eq.icath.and.ik.eq.2) print 148,
         dfloat(jx-1)/dfloat(nj-1),
c 1 pOmpsk(ikp(ik,ian,j),
c 1 ian,jx),pOmpsk(ikp(ik,ian,j),icath,jx),pOmpsk(ikp
148 (ik,icath,j),ian,jx),pOmpsk(ikp(ik,icath,j),icath,jx)
   format(5(lpell.4,1x))
148 continue
do 153 jx = 1,nj+28+npextr
153 pfine(ikp(ik,icath,j),ielec,jx) = -pfine(ikp(ik,ian,j),
1 do 180 ic=2,npts
   if(nrxn.eq.1) go to 180
   do 170 j=2,nrxn
   do 170 i=1,nspec
      cank(ikp(ik,ieleck,j),i,icoll(ian,ic)) =
      cank(ikp(ik,ieleck,l),i,icoll(ian,ic))
      ccathk(ikp(ik,ieleck,j),i,icoll(icath,ic)) =
      ccathk(ikp(ik,ieleck,l),i,icoll(icath,ic))
   170 continue
   continue
   end if
end if
istop = 0
if(istop.ne.0) stop
icount = 0
print *, 'calculate csave'
do 182 i=1,nspec
do 182 ic=2,npts
do 182 ielec=ian,icath
csave(ielec,i,icoll(ielec,ic)) = csurf(ielec,i,icoll(ielec,ic))
182 continue
continue
Update coefficients for the appropriate electrode(s)
factor=1.0d0
if (iit.gt.4) factor=0.7d0
if (iit.gt.4) factor=0.5d0
do 190 ielec=ian,icath
do 190 j=1,nrxn
do 190 ik=1,npts
c(ik,ielec,j) = cold(ik,ielec,j)
   + delco(ik,ielec,j)*factor
print *, 'cold(1,1,1)', cold(1,1,1), 'delco', delco(1,1,1)
print *, 'coefficients from delco'
print 120,co(1,1,1)
print 120,((c(ik,ielec,j),ielec=1,2),ik=2,npts),
   j=1,nrxn)
if(iit.eq.1) go to 230
do 210 i=1,nspec
do 210 ic=2,npts
   csurf(ian,i,icol(ian,ic)) = 0.d0
csurf(icath,i,icol(icath,ic)) = 0.d0
do 200 j=1,nrxn
do 200 ieleck=ian,icath
do 200 ik=2,npts
   csurf(ian,i,icol(ian,ic)) =
     co(ik,ieleck,j)*
     cank(ikp(ik,ieleck,j),i,icol(ian,ic)) +
     csurf(ian,i,icol(ian,ic))
csurf(icath,i,icol(icath,ic)) =
     co(ik,ieleck,j)*
     *ccathk(ikp(ik,ieleck,j),i,icol(icath,ic)) +
     csurf(icath,i,icol(icath,ic))
200       continue
do 210 ielec=ian,icath
   jx = icol(ielec,ic)
csurf(ielec,i,jx) = ciocr(i) + csurf(ielec,i,jx)
210       continue
230       continue
       print *, 'concentrations with new coeffs'
       print *, 'species 1'
       print 280,((csurf(ielec,1,icol(ielec,ic)),ielec=ian,icath),
   ic=2,npts)

Find the cathode coefficients that give zero concentration
on the cathode (if the cathode is the limiting electrode)

print *, 'limelec',limelec,'ian',ian,'icath',icath
if(limelec.ne.ian .and. limelec.ne.icath) go to 75
if(iit.eq.1) then
   j = 1
   do 90 irow = 2,npts
      do 90 ielecr = ian,icath
         jx = icol(ielecr,irow)
         ib = ikp(irow,ielecr,j) - 1
         if(limelec.eq.ielecr) then
            csurf(ielecr,1,jx) = 0.d0
         else
            csurf(ielecr,1,jx) = 2.d0
         end if
         d(ib,1) = 0.d0
      end if
   do 89 icol = 2,npts
      do 89 ielecc = ian,icath
         jb = ikp(icol,ielecc,j) - 1
b(ib,jb) = 0.d0
if(irow.ne.npts) then
  if(ielecr.eq.ian .and. ielecc.eq.ian) b(ib,jb) =
     cik(ikp(icol,ielecc,j),l,jx)
  if(ielecr.eq.icath .and. ielecc.eq.icath) b(ib,jb) =
     cik(ikp(icol,ielecc,j),l,jx)
end if
  cik(ib,jb) = b(ib,jb)
89 continue
if(irow.eq.npts) then
  if(ielecr.eq.ian) then
    b(ib,1) = 1.d0
    cik(ib,1) = 1.d0
  else
    b(ib,4) = 1.d0
    cik(ib,4) = 1.d0
  end if
end if
90 continue

print *,'cik matrix'
print 92,((b(i,j),j=1,2*(npts-1)),i=1,(2*npts-1))
print *,'d vector'
print 92,(d(i,l),i=1,2*(npts-1))
92 format(10(lr,lpell,4))
call matinv(2*(npts-1),2*(npts-1),1,b,d,determ)
print *,'after matinv d vector'
print 92,(d(i,l),i=1,2*(npts-1))
if(determ.eq.0.d0) then
  print *,'zero determinant in cik'
  stop
end if
j = 1
70 do icol=2,npts
   do ielecc=ian,icath
      co(icol,ielecc,j) = d(ikp(icol,ielecc,j)-1,1)
      print *,'coefficients from cik'
      print 120,co(1,l,1)
      print 120,((co(ik,ielec,j),ielec=1,2),ik=1,nrnx),
      j=1,nrnx)
   end do
75 continue
60 do ithru = 1,2
   do i = 1,nspec
      do ic = 2,npts
         csurf(ian,i,icoll(ian,ic)) = 0.d0
         csurf(icath,i,icoll(icath,ic)) = 0.d0
      end do
   end do
end do
do 50 j=1,nrxn
do 50 ieleck=ian,icath
do 50 ik=2,npts
   csurf(ian,i,icoll(ian,ic)) = co(ik,ieleck,j)*
   cank(ikp(ik,ieleck,j),i,icoll(ian,ic)) +
   csurf(ian,i,icoll(ian,ic))
   csurf(icath,i,icoll(icath,ic)) = co(ik,ieleck,j)*
   ccat(k(ikp(ik,ieleck,j),i,icoll(icath,ic)) +
   csurf(icath,i,icoll(icath,ic))
50 continue
do 60 ielec=ian,icath
   jx = icoll(ielec,ic)
   csurf(ielec,i,jx) = ciocr(i) + csurf(ielec,i,jx)
60 continue
print *,'concentrations with cik coeffs '
print *,'species 1'
print 280,((csurf(ielec,l,icoll(ielec,ic)),ielec=ian,icath),
   1 ic=2,npts)
if(ithru.gt.l) go to 76
   j = 1
   do 990 irow=2,npts
      do 990 ielecr=ian,icath
         jx = icoll(ielecr,irow)
         ib = ikp(irow,ielecr,j) - 1
         d(ib,1) = 0.00
         if(irow.ne.npts) then
            if(ielecr.eq.ian) d(ib,1) = 2. - csurf(ielecr,l,jx)
            if(ielecr.eq.icath) d(ib,1) = -csurf(ielecr,l,jx)
         end if
      990 continue
   call matinv(2*(npts-1),2*(npts-1),l,cik,d,deterrn)
   print *,'after matinv delta coeffs'
   print 92,(d(i,1),i=1,2*(npts-1))
970 continue
970 co(icol,ielecc,j) = d(ikp(icol,ielecc,j)-1,1) +
   co(icol,ielecc,j)
print *,'new coefficients from cik'
print 120,co(1,1,1)
print 120,(((co(ik,ielec,j),ielec=1,2),ik=2,npts),
   1 j=1,nrxn)
76 continue
end if
if(istop.ne.0) stop

    do 250 j=1,nrxn
    do 250 ic=2,npts
        do 240 ielec=ian,icath
            jx = icoll(ielec,ic)
        240 xij(ielec,j,jx) = 0.
    do 250 ik=2,npts
        jxa = icoll(ian,ic)
        jfinea = ifine(ndec,jxa)
        jxc = icoll(icath,ic)
        jfinec = ifine(ndec,jxc)
        xij(ian,j,jxa) = co(ik,ian,j)*xijka(ik,j,jfinea) +
        1 xij(ian,j,jxa)
        xij(icath,j,jxc) = co(ik,icath,j)*xijkc(ik,j,jfinec) +
        1 xij(icath,j,jxc)
    continue

    call sumi(xi,xij)

    do 270 ic=2,npts
    do 270 ielec=ian,icath
        jx = icoll(ielec,ic)
        p0mps(ielec,jx) = 0.
    do 260 j=1,nrxn
    do 260 ieleck=ian,icath
    do 260 ik=2,npts
        p0mps(ielec,jx) = co(ik,ieleck,j)*
        1 p0mpsk(ikp(ik,ieleck,j),ielec,jx) + p0mps(ielec,jx)
    continue

    continue

    print *, 'concentrations input to B-V'
    print *, 'species 1'
    print 280,((csurf(ielec,1,ico11(ielec,ic)),ie1ec=ian,icath),
    1 ic=2,npts)
    280 format(2(1x,lpe15.8))

    print *,
    istop = 0
    if(istop.ne.0) stop
    xnew = co(1,1,1)
    id1 = 1
    if(matcalc.eq.0) idl = 2
    do 310 id=1,idl
        if(id.gt.1) then
            if(ivolts.eq.1) then
                v(icath) = 0.
                v(ian) = vpdelta
            end if
            if(id.eq.1) then
                v(ian) = 0.
                v(icath) = -vpdelta
            end if
        end if
    310 continue
NEWCHAN.FOR

else
  xjadum(jrxn) = pjpd
end if
end if

c

iic = 2
max = 1

c
if(iit.eq.1 .and. matcalc.ne.0) max = 40
if(matcalc.ne.0 .and. iit.lt.iitcut) max = 50
do 295 inner=1,max
  print *, 'inner', 'inner', 'phistr', xnew
  call current(xij, xijka, xijkc, iall, ynew, xnew, ...
  xitotc, xitota, p0mps, csurf, cank, ccathk, xijnew, xinew, ...
  xitotjk, p0mpsk, dxijka, dxijkc, v, xjdmu)
  nav = (npts+2)/2
  if(limelec.ne.ian) then
    delphi = (xij(ian,1,icoll(ian,nav)) - xijnew(ian,1, ...
      icoll(ian,nav))) / dxijka(1,1,icoll(ian,nav))
  else
    delphi = (xij(icath,1,icoll(icath,nav)) - xijnew(icath,1, ...
      icoll(icath,nav))) / dxijkc(1,1,icoll(icath,nav))
  end if
  if(dabs(delphi).lt.1.d-02 .or. max.eq.l) go to 296
  print *, 'delphi', delphi, 'xij' , xij(ian,1,icoll(ian,nav)), ...
      'xijnew(ian,1,icoll(ian,nav)))
  if(dabs(delphi).gt.4.d0) delphi = 4.d0*dabs(delphi)/delphi
  xnew = xnew + delphi
  print *,'no convergence on initial guess for phi *'
  stop
295 continue
print *,'no convergence on initial guess for phi *'
stop
296 co(l,1,1) = xnew

c
297 phistr = xnew
print *,' error(elec) iBV(elec) iapp(cath)'
if(iall.eq.0) then
  jx1 = 2
  jx2 = npts
else
  jx1 = 1
  jx2 = nj
end if
do 300 j=1, nrxn
  do 300 ielec=ian, icath
    do 300 ic=jx1,jx2
      if(iall.eq.0) then
        jx = icoll(ielec,ic)
      else
        jx = ic
      end if
    end do
  end do
end do 300
300 continue

end if
if(id.eq.1) then
  error(ielec,j,jx) = xijnew(ielec,j,jx) - xij(ielec,j,jx)
  if(ielec.eq.ian) print 290,dfloat(jx-1)/dfloat(nj-1),
     error(ian,j,jx),xijnew(ian,j,jx),xij(ian,j,jx)
  if(ielec.eq.icath) print 290,dfloat(jx-1)/dfloat(nj-1),
     csurf(icath,1,jx),
     xijnew(icath,j,jx),xij(icath,j,jx)
else
  errsav(ielec,j,jx) = xijnew(ielec,j,jx) - xij(ielec,j,jx)
  end if
end if
290 format(1x,Opf8.4,lx,lpe20.13,2(lx,lpell.4))
300 continue
310 continue
istop = 0
if(istop.ne.0) stop
if(matcalc.eq.0) return
ierr = 0

do 320 j=1,nrxn
  do 320 ielec=ian,icath
    do 320 ic=2,npts
      if(abs(error(ielec,j,icoll(ielec,ic))*xijnew(ielec,j,
           icoll(ielec,ic))) .gt. errtst) then
        ierr=ierr+1
      end if
  end do
  print *, 'ielec', ielec,'jx' ,icoll(ielec,ic),'error',
     error(ielec,j,icoll(ielec,ic)),'xijnew',
     xijnew(ielec,j,icoll(ielec,ic)),'errtst',errtst
  end if
320 continue

do 340 j=1,nj
  do 340 i=1,nspec
    csurf(ian,i,jx) = 0.d0
    csurf(icath,i,jx) = 0.d0
  end do
  do 330 j=1,nrxn
    do 330 ieleck=ian,icath
      do 330 ik=2,npts
        csurf(ian,i,jx) = co(ik,ieleck,j)*
        cank(ikp(ik,ieleck,j),i,jx) + csurf(ian,i,jx)
      end do
    end do
  end do
  do 340 ielec=ian,icath
340  \text{csurf}(i, j x) = \text{ciocr}(i) + \text{csurf}(i, j x)
\text{print} *, 'species 1'
\text{print} 280, ((\text{csurf}(i, 1, \text{icoll}(i)), \text{ielec} = \text{ian}, \text{icath}),
\text{ic} = 2, \text{npts})
\text{do 350} j x = 1, nj
\text{jx} = \text{ifine}(\text{ndec}, j x)
\text{do 350} \text{ielec} = \text{ian}, \text{icath}
\text{ps0ps}(\text{ielec}, j x) = 0. d0
\text{do 350} j = 1, \text{nrxn}
\text{xij}(\text{ielec}, j, j x) = 0. d0
\text{do 350} \text{ik} = 2, \text{npts}
\text{ia} = \text{ikp}(\text{ik}, \text{ian}, j)
\text{ic} = \text{ikp}(\text{ik}, \text{icath}, j)
\text{if}(\text{ielec} .eq. \text{ian}) \text{xij}(\text{ian}, j, j x) = \text{co}(\text{ik}, \text{ian}, j) * 
\text{xijka}(\text{ik}, j, \text{jx} \text{fine}) + \text{xij}(\text{ian}, j, j x)
\text{if}(\text{ielec} .eq. \text{icath}) \text{xij}(\text{icath}, j, j x) = \text{co}(\text{ik}, \text{icath}, j) * 
\text{xijkc}(\text{ik}, j, \text{jx} \text{fine}) + \text{xij}(\text{icath}, j, j x)
\text{do 350} \text{ieleck} = \text{ian}, \text{icath}
\text{ps0ps}(\text{ielec}, j x) = \text{co}(\text{ik}, \text{ieleck}, j) * 
\text{ps0psk}(\text{ikp}(\text{ik}, \text{ieleck}, j), \text{ielec}, j x) + \text{ps0ps}(\text{ielec}, j x)
\text{do 351} \text{nextra} = 1, \text{nje} + 28 + \text{npext}
\text{do 351} \text{ielec} = \text{ian}, \text{icath}
\text{ptemp}(\text{ielec}, \text{nextra}) = 0. d0
\text{do 351} \text{ik} = 2, \text{npts}
\text{ia} = \text{ikp}(\text{ik}, \text{ian}, j)
\text{ic} = \text{ikp}(\text{ik}, \text{icath}, j)
\text{do 351} \text{ieleck} = \text{ian}, \text{icath}
\text{ptemp}(\text{ielec}, \text{nextra}) = \text{co}(\text{ik}, \text{ieleck}, j) * \text{pfine}(\text{ikp}(\text{ik}, \text{ieleck}, j),
\text{ieelec}, \text{nextra}) + \text{ptemp}(\text{ielec}, \text{nextra})
\text{call sumi(xi, xij)}
\text{end if}
\text{Set up matrix equation for calculating new current coefficients}
(\text{and concentration ordinates, if they are being used})
\text{do 370} \text{ik} = 2, \text{npts}
\text{do 370} \text{ielec} = \text{ian}, \text{icath}
\text{do 370} j = 1, \text{nrxn}
\text{b(1, ikp(\text{ik}, \text{ielec}, j))} = \text{xitotj}(\text{ik}, \text{ielec}, j)
\text{d(1, 1)} = 0. d0
\text{do 380} \text{ielec} = \text{ian}, \text{icath}
do 380 j=1,nrxn
do 380 ik=2,npts
380 d(1,1) = -co(ik,ielec,j)*xitotjk(ik,ielec,j) + d(1,1)

do 400 jc=1,nrxn
do 400 ic=2,npts
do 400 ielecc=ian,icath
if(ielecc.eq.ian) then
  b(ikp(ic,ielecc,jc), 1) = dxijka( 1, jc, icoll(ian,ic) )
  if(limelec.eq.ian) b(ikp(ic,ielecc,jc), 1) = 0.d0
else
  b(ikp(ic,ielecc,jc), 1) = dxijkc( 1, jc, icoll(icath,ic) )
  if(limelec.eq.icath) b(ikp(ic,ielecc,jc), 1) = 0.d0
end if
if(ic.eq.npts) b(ikp(ic,ielecc,jc),1) = 0.d0

do 390 jk=1,nrxn
do 390 ik=2,npts
do 390 ieleck=ian,icath
  ib = ikp(ic,ielecc,jc)
  jb = ikp(ic,ieleck,jk)
if(ielecc .eq. ian) then
  if(ieleck.eq.ian) then
    if(limelec .eq. ian) then
      if(ieleck.eq.ian) then
        b(ib,jb) = cank(jb,l,icoll(ian,ic))
      else
        b(ib,jb) = 0.d0
      end if
    else
      xijk = xijka(ik, jc, icoll(ian,ic))
      b(ib,jb) = dxijka( jb, jc, icoll(ian,ic))
      if(ielecc .eq. ieleck .and. jk.eq.jc) then
        b(ib,jb) = b(ib,jb) - xijk
      end if
    end if
  else
    if (limelec.eq.icath) then
      if(ieleck.eq.icath) then
        b(ib,jb) = ccathk(jb,l,icoll(icath,ic))
      else
        b(ib,jb) = 0.d0
      end if
    else
      xijk = xijkc(ik, jc, icoll(icath,ic))
      b(ib,jb) = dxijkc( jb, jc, icoll(icath,ic))
      if(ielecc .eq. ieleck .and. jk.eq.jc) then
        b(ib,jb) = b(ib,jb) - xijk
      end if
    end if
  end if
else
  if (limelec.eq.icath) then
    if(ieleck.eq.icath) then
      b(ib,jb) = ccathk(jb,l,icoll(icath,ic))
    else
      b(ib,jb) = 0.d0
    end if
  else
    xijk = xijkc(ik, jc, icoll(icath,ic))
    b(ib,jb) = dxijkc( jb, jc, icoll(icath,ic))
    if(ielecc .eq. ieleck .and. jk.eq.jc) then

\begin{verbatim}
         b(ib,jb) = b(ib,jb) - xijk
    end if
end if
end if
if(ic.eq.npts) then
    b(ib,jb) = 0.d0
    if(ik.eq.2.and.ieleck.eq.ian.and.ielecc.eq.ian)
        b(ib,jb) = 1.d0
    if(ik.eq.3.and.ieleck.eq.icath.and.ielecc.eq.icath)
        b(ib,jb) = 1.d0
end if
continue
if(ielecc.eq.limelec) then
    d(ib,l) = -csurf(ielecc,l,icoll(ielecc,ic))
    print *,'-csurf',d(ib,l)
else
    d(ib,l) = -error(ielecc,ic,icoll(ielecc,ic))
end if
if(ic.eq.npts) d(ib,l) = 0.d0
continue
d(ikp(npts,ian,l),l) = 0.d0
do 410 i-l,nptstot
    do 410 j-l,nptstot
        bsave(i,j) = b(i,j)
        print *, 'in err bsave matrix'
        print 420,((bsave(i,j),j=1,nptstot),i=1,nptstot)
        format(10(lx,lpell.4))
        print *, 'd vector'
        print 420,(d(i,l),l=1,nptstot)
call matinv(nptstot,nptstot,l,b,d,determ)
print *, 'after matinv, d vector'
print 420,(d(i,l),l=1,nptstot)
if(determ.eq.0.) then
    print *, 'zero det in matinv'
    print *, 'b matrix'
    print 420,((b(i,j),j=1,nptstot),i=1,nptstot)
    stop
end if
\end{verbatim}
Don’t let phi * change by more than 4

iuncon = 0
coldsv = cold(1,1,1)
do 430 j=1,nrxn
do 430 ik=1,npts
do 430 ielec=ian,icath
   delco(ik,ielec,j) = d(ikp(ik,ielec,j), 1)
   if (dabs(delco(ik,ielec,j)).gt.1.d-12) iuncon = 1
   delco(ik,ielec,j) = 0.d0
430 cold(ik,ielec,j) = co(ik,ielec,j)
    if(dabs(delco(1,1,1)).gt.4.d0)
        delco(1,1,1) = 4.d0*delco(1,1,1)/dabs(delco(1,1,1))
cold(2,2,1) = 1.d-06
print *,’delco(1,1,1’’,delco(1,1,1),’cold’,cold(1,1,1)
   nits = iit
   if(ierr.eq.0 .or. iuncon.eq.0) then
      ierr = 0
      return
   end if
440 continue

ierr = 0
    if(matcalc.eq.1) ierr = 1
return
end

SUBROUTINE NLIM(FLUXCK,IK)

Calculates the flux due to a step change on one electrode
and zero flux on the other electrode

implicit double precision (a-h,o-z)
dimension fluxak(41,10,404),fluxck(41,10,404),
cank(41,10,101),sum0(2,10),suml(2,10),cafine(41,10,404),
b(40,40),d(40,1),asum0(2,10),asuml(2,10)
common/cdata/xint0(10,101,5),xint1(10,101,5),n2,nd,isec
common/elecs/ian,icath
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,
uref(5),nspec,nrxn,rho0
common/kinpars/sij(10,5),ratson(10,5),ason(2,5),ifor,iback
common/geom/pi,hol,nj,ndec,icut
common/co/coeff,xn
common/evprob/xlambd(3),ai(3),z,diodr(10),ciocr(10)
c

cdamp = 1.0d-06
n2 = 2*nspec
jbegin = 2
kend = (nj-1)/icut + 1
c
i = 1
do 10 jx=1,nj
10 cank(ik,i,jx) = ciocr(i)
c
do 19 idec=1,ndec
h = dfloat(icut)**(idec-ndec)/dfloat(nj-1)
do 17 jx=jbegin,nj
zeta = (jx-1)*h*diodr(i)/z
jxfine = ifine(idec,jx)
c
do 11 ielec=ian,icath
sum0(ielec,i) = 0.d0
c
if(jx.eq.2) go to 13
n = 'x - 1
rml = n - 1
c
do 12 k=1,nml
kp1 = k + 1
nmkpl = n - k + 1

kpl = n - k + 1

term = xint0(i,nmkpl,idec) * (cank(ik,i,kpl)-cank(ik,i,k))
print *, 'jxfine',jx,' k',k,' term',term,'xint0(i,nmkpl,idec)',

print *, 'jx',jx,' k',k,' term',term,'xint0(i,nmkpl,idec)',

12 sum0(ian,i) = sum0(ian,i) + (cank(ik,i,kpl)
-cank(ik,i,k))*xint0(i,nmkpl,idec)
c
13 continue
c
bb = -xint0(i,1,idec)

1 dd = sum0(ian,i) - cank(ik,i,jx-1)*
 xint0(i,1,idec) - flak(zeta)

canc(ik,i,jx) = dd/bb
17 continue
do 20 jx=1,nj
20 cafine(ik,i,ifine(idec,jx)) = cank(ik,i,jx)
if(idec.eq.ndec) go to 19
do 18 k=1,kend
jj = k*icut - icut + 1
18  cank(ik,i,k) = cank(ik,i,jj)
jbegin = kend + 1
19  continue
istop = 0
if(istop.ne.0) stop

jbegin = 2

do 190 idec=1,ndec
h = dfloat(icut)**(idec-ndec)/dfloat(nj-1)
do 170 jx=jbegin,nj
    zeta = (jx-1)*h*diodr(i)/z
    jxfine = ifine(idec,jx)
do 110 ielec=ian,icath
    asum0(ielec,i) = 0.d0
    asuml(ielec,i) = 0.d0
110  continue
 if(jx.eq.1) go to 130
n = jx - 1

do 120 k=1,n
    kpl = k + 1
    nmkpl = n - k + 1
    asum0(ian,i) = asum0(ian,i) + (cafine(ik,i,
1      ifine(idec,kpl))
1      -cafine(ik,i,ifine(idec,k))*xint0(i,nmkpl,idec)
    asuml(ian,i) = asuml(ian,i) + (cafine(ik,i,
1      ifine(idec,kpl))
1      -cafine(ik,i,ifine(idec,k))*xintl(i,nmkpl,idec)
120  continue

130  continue

140  continue

do 140 ielec=ian,icath
iothr = iother(ielec)
if(ielec.eq.ian) then
    fluxak(ik,i,jxfine) = (+asum0(ielec,i)
1      - flak(zeta))*coeff*diodr(i)
else
    fluxck(ik,i,jxfine) = (flck(zeta)
1      - asuml(iothr,i))*coeff*diodr(i)
end if
140  continue
170  continue

fluxck(ik,i,ifine(idec,1)) = 1.d30*coeff*diodr(i)
fluxak(ik,i,ifine(idec,1)) = 0.d0
jbegin = kend + 1
190 continue
c fill in the other decades
c
if(ndec.eq.1) return
do 180 idec=2,ndec
do 180 k=1,kend
   jj = k*icut - icut + 1
180 fluxck(ik,i,ifine(idec,k)) = fluxck(ik,i,ifine(idec-1,jj))
c
do 200 idec=1,ndec
c   print *,'idec',idec
c   do 200 jx=1,nj
c      jxfine = ifine(idec,jx)
c      print *,'jx',jx,'fluxc',fluxck(ik,1,jxfine),'fluxa',
c      1 fluxak(ik,1,jxfine)
c 200 continue
istop=0
if(istop.ne.0) stop
return
c
FUNCTION P(N,X)
c implicit double precision (a-h,o-z)
c Calculation of Legendre polynomials
c
pl = 1.0
p2 = 2.0*x - 1.0
if (n-1) 1,2,3
1 p = pl
   return
2 p = p2
   return
3 nml = n - 1
   do 4 nu=1,nml
      p = ((x*dfloat(4*nu+2)-(2*nu+1))*p2 - dfloat(nu)*pl) / 
      1 dfloat(nu+1)
      pl = p2
4 p2 = p
   return
end
c
FUNCTION FLAK(ZETA)

Solution to Graetz problem:
flux due to a step change in concentration on the other electrode

implicit double precision (a-h,o-z)
common/evprob/xlamda(3),ai(3),z,diodr(10),ciocr(10)

aa = 0.9594d+00
bb = 0.6069d+00
c = 0.4512d+00
dd = 0.276d+00
flak = 0.

if(zeta.eq.0.) return

if(zeta.gt.0.18d0) then
    sum = 0.d0
    do 52 j=1,3
        52 sum = sum + ai(j)*dexp(-xlamda(j)*zeta)
    flak = 1. - 2.*sum
else
    flak = dexp(aa-bb/zeta - cc*dexp(-dd/zeta))
end if
return
end

FUNCTION FLCK(ZETA)

Solution to Graetz problem:
flux due to a step change in concentration on the same electrode

implicit double precision (a-h,o-z)
common/evprob/xlamda(3),ai(3),z,diodr(10),ciocr(10)

a = 1.35659745d0
b = 0.2d0
c = 0.060733452d0

if(zeta.eq.0.) then
    flck = 1.d30
    return
end if

if (zeta.lt.0.11) then
    flck = a*zeta**(-1./3.) - b - c*zeta**(1./3.)
else
    sum = 0.d0
do 100 j=1,3
100   sum = sum + dabs(ai(j))*dexp(-xlamda(j)*zeta)
   end if

return
end

SUBROUTINE EQFIVE(IK,XOL,PHIAN,PHICATH,P2,P3,P5)

This subroutine calculates the dimensionless form of the
((phi_0) - (phi_*)/i<avg> given by equation 5 in Newman
and Parrish, JES, 117, 43-48, 1970. The singularity in
the equation was eliminated by the method of Kantorovich
and Krylov

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

common/geom/pi,hol,nj,ndec,icut
   common/gauss/xp(96),gw(96),ngt,xx(201),xielec(201),xiothr(20
11),njmore
   common/elecs/ian,icath
   common/eualc/ikboth
   common/params/deltal,delta2,delta3

ielec=ian
iothr = icath
an = 0.5d0

if(xol.gt.0.d0 .and. xol.lt.1.d0) then
   xmin = xol/2.
   xmax = xmin + 0.5
   xie = curfcn(ik,ian,xol)
   if(ik.ne.ikboth) then
      xio = 0.d0
   else
      xio = curfcn(ik,icath,xol)
   end if
else
   xmin = 0.5
   xmax = xmin
   xie = 0.
   xio = 0.
end if

130  sumra = 0.d0
      sumrb = 0.d0
NEWCHAN.FOR

```fortran
sumrc = 0.d0
cf sumrf = 0.d0
sumrs = 0.d0
sumrt = 0.d0
c
do 180 iregn=1,4
  go to (101,102,103,104),iregn
101    x11 = 0.d0
    xul = xmin**an
    go to 150
102    if(xol.ge.1.d0 .or. xol.le.0.d0) go to 180
    if(xol.ge.0.75) then
      xul = (1.-xmin)**an
      x11 = (1.-xol)**an
    else
      x11 = xmin**an
      xul = xol**an
    end if
    go to 150
103    if(xol.le.0.d0 .or. xol.ge. 1.d0) go to 180
    if(xol.le.0.25) then
      xll = xol**an
      xul = xmax**an
    else
      xll = (1.-xol)**an
      x11 = (1.-xmax)**an
    end if
    go to 150
104    xul = (1.-xmax)**an
    x11 = 0.d0
150    suma = 0.d0
    sumb = 0.d0
    sumc = 0.d0
cf sumf = 0.d0
    sums = 0.d0
    sumt = 0.d0
c
do 170 jxp=1,ngt
    xprm = xp(jxp)*(xul-x11) + x11
    z = xprm**(1./an)
    if ((iregn.eq.3.and.xol.gt.0.25) .or. iregn.eq.4 .or.
      (iregn.eq.2.and.xol.gt.0.75)) z = 1. - z
    y = c*(xol-z)
    sinh2 = (dexp(y) - dexp(-y))**2/4.0
    if(dabs(y).lt.0.01) sinh2 = (y*(1. + y**2/6.*(1. +
      y**2/20.*(1. + y**2/42.))))**2
    cosh2 = 1. + sinh2
    if(y.ne.0) alnth2=dlog(sinh2/cosh2)
```

aa = curfcn(ik,ian,z)
if(ik.ne.ikboth) then
  bb = aa
else
  bb = aa + curfcn(ik,icath,z)
end if
Y2 = xprm**(l./an - 1.) / an
clnch2 = dlog(cosh2)*Y2
if(y.ne.0.) suma = suma + gw(jxp)*(aa*alnth2 - xie*dlog(y**2)) * Y2
  if(y.ne.0.) snum = snum + gw(jxp)*((bb-aa)*alnth2-xio*dlog(y**2)) * Y2
  sumc = sumc + gw(jxp)*bb*clnch2
  sumf = sumf + gw(jxp)*bb*(z-0.5)*Y2
  sums = sums + gw(jxp)*bb*Y2
  sumt = sumt + gw(jxp)*clnch2
continue
sumra = suma/2.*(xul-xll) + sumra
sumrb = snum/2.*(xul-xll) + sumrb
sumrc = sumc/2.*(xul-xll) + sumrc
sumrf = sumf/2.*(xul-xll) + sumrf
sumrs = sums/2.*(xul-xll) + sumrs
sumrt = sumt/2.*(xul-xll) + sumrt
continue
g2 = 0.d0
  if(xol.lt.1.d0 .and. xol.gt.0.d0) g2 = dlog(c**2) +
    2.*(1.-xol)*(dlog(1.-xol) - 1.) + 2.*xol*(dlog(xol) - 1.)
pl = sumrf/(2.*hol)
  if(ik.ne.ikboth) sumrs = 0.d0
  p2 = -sumra/(2.*pi)
p3 = -(g2*xie+sumra)/(2.*pi)
p5 = -(sumrc-sumrs*sumrt)/(2.*pi)
phicath = -(g2*xio + sumrb)/(2.0*pi)+p5
phian = -(g2*xie + sumra)/(2.0*pi)+p5
return
end
SUBROUTINE READIN(XN,XJ,UJTH,NSUBJ,NG,ROOT,
  NRE,M,IRSTRT,VCELL,VSTART,JRXN,XJJ,IVOLTS,DELTAP,
  ERRTST,MAXIT,IREAD)
This subroutine reads and prints the input data
implicit double precision (a-h,o-z)
dimension root(96),ujth(5),alpha(2,5),
1 nsubj(5),xj(5)
c
common/evprob/xlamda(3),ai(3),z,diodr(10),ciocr(10)
common/geom/pi,hol,nj,ndec,icut
common/rxnpars/cref(10),cre(10),porq(2,10,5),ip,iq,
1 uref(5),nspec,nrxn,rho0
common/kinpars/sij(10,5),ratson(10,5),asen(2,5),ifor,iback
common/cdata/xint0(10,101,5),xint1(10,101,5),n2,nd,iseec
common/gauss/xp(96),gw(96),ngt,xx(201),xielec(201),xiothr(20
11),njmore
common/colloc/npts,icoll(2,41),npextr,np(7)
common/params/deltal,delta2,delta3

read 120,ichar
read *,ng
if(ng.gt.48 .or. ng.lt.1) print
100 format(' ng=',i2,' is incorrect input')
c Read the Gaussian quadrature roots and weights for the
interval 0 to 1. Then put in the roots and weights for
the interval -1 to 0.
c
do 110 k=1,ng
  karg = ng + 1 - k
  kpng = k + ng
  read *,(root(karg),gw(karg))
  root(kpng) = root(karg)
  gw(kpng) = gw(karg)
110 root(karg) = -root(karg)
c
read *,(xlamda(i),i=1,3)
read *,(ai(i),i=1,3)
read 120,ichar
120 format(a72)
read *,nj,ndec,irstrt
print 130,irstrt
130 format(' restart=',i3)
print 130,irstrt
140 format(' nj=',i3,' is incorrect input'
if(nj.gt.101 .or. nj.lt.1) print 140,nj
140 format(' nj=',i3,' is incorrect input'
if(ndec.gt.5 .or. ndec.lt.1) print 150,ndec
150 format(' ndec=',i3,' is incorrect input'
c
read 120,ichar
read *,hol,icut,xn,pehol,rho0
read 120,ichar
read *,vcell,vstart,xjj,jrxn,ivolts,deltap,errstt,maxit,
1 * ir,m
  if(ivolts.eq.1) then
    print *, 'continuation on voltage'
  else
    print *, 'continuation on J(',jrxn,')'
  end if
  print 155,errtst,maxit
155 format('convergence criterion',lpell.4,
  'maximum iterations',i2)
  print 220,nj,ndec,hol,icut,xn,pehol,rho0,vcell
220 format('nj=',i3,' ndec=',i3,' hol=',lpe11.4,'
    icut=',i3,' x=',lpe11.4,
    rho=',lpe11.4,' v(cell)=',lpe11.4)
  print *, 'principal reactant',ir,'main rxn',m
  z=3./16. * pehol
  read 120,ichar
  read *,isec,nspec,nrxn,nre
  if(isec.ne.0) print 230
230 format('secondary current distribution')
  print 240,nspec,nrxn,nre
240 format(i3,' species,',i3,
    ' reactions, reference rxn is rxn',i3,/)  
  if(nspec.gt.10 .or. nspec.lt.1) print 250,nspec
250 format('nspec=',i3,' is incorrect input')
  if(nrxn.gt.5 .or. nrxn.lt.1) print 260,nrxn
260 format('nrxn=',i3,' is incorrect input')
  if(nre.gt.5 .or. nre.lt.1) print 270,nre
270 format('nre=',i3,' is incorrect input')
  read 120,ichar
  read *,ujth(nre)
  do 310 j=1,nrxn
    read 120,ichar
    read *,alpha(1,j),alpha(2,j),nsubj(j),xj(j),ujth(j)
    if(j.eq.nre .and. ujth(j).ne.ujth(nre)) print *,
1     'error in ujth(nre)'
    print 300,alpha(1,j),alpha(2,j),xj(j),ujth(j),nsubj(j)
300 format('alpha(a)=',lpe11.4,' alpha(c)=',lpe11.4,
1     ' xj=',lpe11.4,' ujth=',lpe11.4,/
    ' nsubj=',i2)
310 continue
  do 340 i=1,nspec
    read 120,ichar
read *,diodr(i),cref(i),cre(i),ciocr(i)
print 330,diodr(i),cref(i),cre(i),ciocr(i)
330 format(' diodr=',1pe11.4,' cref=',1pe11.4,' cre=',1pe11.4,' ciocr=',1pe11.4)
340 continue

do 370 i=1,nspec
   read 120,ichar
   read *,(sij(i,j),j=1,nrxn)
do 360 j=1,nrxn
      print 350,sij(i,j),i,j
350 format(' sij=',1pe11.4,' i=',i2,' j=',i2)
do 370 continue
370 continue

do 400 j=1,nrxn
   ason(1,j) = -alpha(1,j)*sij(ir,m)/nsubj(m)
ason(2,j) = -alpha(2,j)*sij(ir,m)/nsubj(m)
print *,'ason(1,j)=' ,ason(1,j),' ason(2,j)=' ,ason(2,j)
do 390 i=1,nspec
   ratson(i,j)=(sij(i,j)/nsubj(j))/(sij(ir,m)/nsubj(m))
   print 380,ratson(i,j),i,j
380 continue
400 continue

doi120,ichar
read *,npts
read 120,ichar
read *,(icol1(1,ic),ic=2,npts)
read 120,ichar
read *,(icol2(2,ic),ic=2,npts)
read 120,ichar
read *,iread
read 120,ichar
read *,delta1,delta2,delta3
read 120,ichar
read *,npxextr
np(1) = npxextr/2
np(2) = 1 + np(1)
np(3) = 8 + np(1)
np(4) = 13 + np(1)
np(5) = 16 + np(1)
np(6) = 21 + np(1)
np(7) = 29 + np(1)
return
end
function xprt(jx,jreal)
implicit double precision (a-h,o-z)
common/colloc/npts,icoll(2,41),npextr,np(7)
common/geom/pi,hol,nj,ndec,icut

xprt = dfloat(jx-np(2))/dfloat(nj-l)
    jreal = 0
if(jx.eq.nlp(3)) jreal = 1
if(jx.ge.nlp(5).and.jx.le.nl+jnp(4)) then
    jreal = jx - (np(4)+1)
    xprt = dfloat(jreal-1)/dfloat(nj-l)
end if
if(jx.eq.nl+jnp(6)) jreal = nj
if(jx.gt.nlp(1) .and. jx.lt.nlp(5)) xprt = dfloat(jx-1)/dfloat(nj-l)
    1
if(jx.gt.nl+jnp(4) .and. jx.lt.nl+jnp(7)) xprt =
    1 dfloat(jx-nil+jnp(6))/
    1 dfloat(nj-nil)/8. + 1.0
if(jx.ge.nl+jnp(7)) xprt = dfloat(jx-nil)/dfloat(nj-l)

return
end

SUBROUTINE GAUSSQ(ik,aviAN,aviCATH,dwnstr)
implicit double precision (a-h,o-z)
common/evprob/xlambda(3),ai(3),z,diodr(10),ciocr(10)
common/co/coeff,xn
common/geom/pi,hol,nj,ndec,icut
common/elecs/ian,icath
common/gauss/xp(96),gw(96),ngt,xx(201),xielec(201),xiothr(201),njmore
common/colloc/npts,icoll(2,41),npextr,np(7)
common/params/deltal,delta2,delta3

ielec=ian
iothr = icath
  c = pi/(2.*hol)
an = 0.5d0

  sumrf = 0.d0
  sumrs = 0.d0
  sumrt = 0.d0
do 170 iregn=1,2
  xll = 0.d0
  xu1 = 0.5**an

do 170 jxp=1,ngt
  xprm = xp(jxp)*(xu1-xll) + xll
  zp = xprm**(1./an)
  if (iregn.eq.2) zp = 1. - zp

  aa = curfcn(ik,ian,zp)
  bb = curfcn(ik,icath,zp)

  Y2 = xprm**(1./an - 1.) / an

  sumrf = sumrf + gw(jxp)*(aa+bb)*(zp-0.5)*Y2
  sumrs = sumrs + gw(jxp)*aa*Y2
  sumrt = sumrt + gw(jxp)*bb*Y2

170 continue

  sumrf = sumrf/2.*(xu1-xll)
  sumrs = sumrs/2.*(xu1-xll)
  sumrt = sumrt/2.*(xu1-xll)

  dwnstr = sumrf/(2.*hol)
  avian=sumrs
  avicath=sumrt

return
end
Appendix H. Tables of Experimental Parameters.
Table H-1. Solution concentrations for disk experiments (mol/liter).

<table>
<thead>
<tr>
<th>Solution</th>
<th>NaOH</th>
<th>KOH</th>
<th>K$_3$Fe(CN)$_6$</th>
<th>K$_4$Fe(CN)$_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a†</td>
<td>0.2457</td>
<td>0.01000</td>
<td>0.05030</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.2582</td>
<td>0.01003</td>
<td>0.04983</td>
<td></td>
</tr>
<tr>
<td>1b</td>
<td>0.2992</td>
<td>0.01001</td>
<td>0.05003</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2.1091</td>
<td>0.04993</td>
<td>0.05008</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.4015</td>
<td>0.01430</td>
<td>0.01479</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.7230</td>
<td>0.00472</td>
<td>0.05020</td>
<td></td>
</tr>
</tbody>
</table>

† Boldface refers to solutions that were analyzed for ferricyanide by iodometric titration with zinc and for ferrocyanide by potentiometric titration with ceric sulfate. All other concentrations were calculated by weighing.
Table H-2. Measured parameters for the 5.00-mm disk experiments.

<table>
<thead>
<tr>
<th>Run</th>
<th>T (°C)</th>
<th>Ω (rpm)</th>
<th>$I_{lim}$ (mA)</th>
<th>O. C. P. (mV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a-1</td>
<td>19.8</td>
<td>400</td>
<td>0.530</td>
<td>+1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.800</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.066</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>1.330</td>
<td>+2</td>
</tr>
<tr>
<td>1a-2</td>
<td>19.8</td>
<td>400</td>
<td>0.546</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>0.546</td>
<td>-1</td>
</tr>
<tr>
<td>1a-3</td>
<td>20.1</td>
<td>400</td>
<td>0.549</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.820</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.094</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>1.366</td>
<td></td>
</tr>
<tr>
<td>1a-4</td>
<td>19.8</td>
<td>2500</td>
<td>1.357</td>
<td>-1.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.102</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.830</td>
<td></td>
</tr>
<tr>
<td>1b-1</td>
<td>19.6</td>
<td>400</td>
<td>0.540</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.807</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.074</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>1.339</td>
<td></td>
</tr>
<tr>
<td>1b-2</td>
<td>19.35</td>
<td>400</td>
<td>0.545</td>
<td>-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.811</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.078</td>
<td></td>
</tr>
<tr>
<td>1b-3</td>
<td>19.6</td>
<td>400</td>
<td>0.540</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.813</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.080</td>
<td></td>
</tr>
<tr>
<td>1b-4</td>
<td></td>
<td>2500</td>
<td>1.350</td>
<td>-1.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.080</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.812</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>0.541</td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>T (°C)</td>
<td>Ω (rpm)</td>
<td>( I_{\text{lim}} ) (mA)</td>
<td>O. C. P. (mV)</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>---------</td>
<td>-----------------</td>
<td>--------------</td>
</tr>
<tr>
<td>2-1</td>
<td></td>
<td>400</td>
<td>2.02</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>3.02</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>4.02</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>5.07</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>5.07</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>5.08</td>
<td></td>
</tr>
<tr>
<td>2-2</td>
<td></td>
<td>2500</td>
<td>5.17</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>4.13</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>4.13</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>3.11</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>2.10</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>2.06</td>
<td></td>
</tr>
<tr>
<td>3-1</td>
<td></td>
<td>400</td>
<td>0.782</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>20.1</td>
<td>900</td>
<td>1.169</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.551</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>1.936</td>
<td></td>
</tr>
<tr>
<td>3-2</td>
<td></td>
<td>2500</td>
<td>1.916</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>19.4</td>
<td>1600</td>
<td>1.535</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>1.156</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>0.769</td>
<td></td>
</tr>
<tr>
<td>3-3</td>
<td></td>
<td>400</td>
<td>0.770</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>19.8</td>
<td>900</td>
<td>1.152</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>1.530</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>1.908</td>
<td></td>
</tr>
<tr>
<td>3-4</td>
<td></td>
<td>2500</td>
<td>1.877</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td>19.5</td>
<td>1600</td>
<td>1.503</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>1.129</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>0.754</td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>T (°C)</td>
<td>Ω (rpm)</td>
<td>$I_{\text{lim}}$ (mA)</td>
<td>O. C. P. (mV)</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>---------</td>
<td>-----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>4-1</td>
<td>20.1</td>
<td>400</td>
<td>0.263</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.394</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>0.522</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>0.650</td>
<td></td>
</tr>
<tr>
<td>4-2</td>
<td>19.9</td>
<td>2500</td>
<td>0.661</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>0.528</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.397</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>0.265</td>
<td></td>
</tr>
<tr>
<td>4-3</td>
<td>20.1</td>
<td>400</td>
<td>0.264</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.393</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>0.522</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2500</td>
<td>0.651</td>
<td></td>
</tr>
<tr>
<td>4-4</td>
<td>19.9</td>
<td>2500</td>
<td>0.657</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1600</td>
<td>0.525</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>900</td>
<td>0.394</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>400</td>
<td>0.264</td>
<td></td>
</tr>
</tbody>
</table>
Table H-3. Calculated parameters for the 5.00-mm disk experiments.*

<table>
<thead>
<tr>
<th>Run</th>
<th>$c_{Ferri}$ (mol/l)</th>
<th>$T$ (K)</th>
<th>$\mu$ (cp)</th>
<th>$\rho$ (g/cc)</th>
<th>$\nu$ (cm$^2$/s)</th>
<th>$D \times 10^6$ (cm$^2$/s)</th>
<th>$Sc$</th>
<th>$\mu D / T \times 10^{10}$ (dyne/K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a-1</td>
<td>0.01000</td>
<td>292.9</td>
<td>1.127512</td>
<td>1.021008</td>
<td>1.1043</td>
<td>6.001</td>
<td>1840</td>
<td>2.31</td>
</tr>
<tr>
<td>1a-2</td>
<td>292.7</td>
<td>1.127521</td>
<td>1.1043</td>
<td>6.197</td>
<td>1784</td>
<td>2.39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1a-3</td>
<td>293.2</td>
<td>1.124752</td>
<td>1.1016</td>
<td>6.250</td>
<td>1763</td>
<td>2.40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1a-4</td>
<td>293.0</td>
<td>1.127507</td>
<td>1.1043</td>
<td>6.322</td>
<td>1747</td>
<td>2.43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1b-1</td>
<td>0.01001</td>
<td>292.7</td>
<td>1.136076</td>
<td>1.023245</td>
<td>1.1103</td>
<td>6.061</td>
<td>1832</td>
<td>2.35</td>
</tr>
<tr>
<td>1b-2</td>
<td>292.7</td>
<td>1.136089</td>
<td>1.1103</td>
<td>6.109</td>
<td>1818</td>
<td>2.37</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1b-3</td>
<td>292.7</td>
<td>1.136076</td>
<td>1.1103</td>
<td>6.116</td>
<td>1815</td>
<td>2.37</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1b-4</td>
<td>293.1</td>
<td>1.136056</td>
<td>1.1102</td>
<td>6.136</td>
<td>1809</td>
<td>2.38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-1</td>
<td>0.049934</td>
<td>293.1</td>
<td>1.588597</td>
<td>1.0949</td>
<td>1.4326</td>
<td>4.209</td>
<td>3404</td>
<td>2.25</td>
</tr>
<tr>
<td>2-2</td>
<td>293.1</td>
<td>1.568597</td>
<td>1.4326</td>
<td>4.399</td>
<td>3257</td>
<td>2.35</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3-1</td>
<td>0.014302</td>
<td>293.2</td>
<td>1.104543</td>
<td>1.021</td>
<td>1.0818</td>
<td>6.036</td>
<td>1792</td>
<td>2.27</td>
</tr>
<tr>
<td>3-2</td>
<td>292.5</td>
<td>1.107251</td>
<td>1.0845</td>
<td>6.039</td>
<td>1796</td>
<td>2.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3-3</td>
<td>292.9</td>
<td>1.107251</td>
<td>1.0845</td>
<td>6.002</td>
<td>1807</td>
<td>2.27</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3-4</td>
<td>292.6</td>
<td>1.107251</td>
<td>1.0845</td>
<td>5.856</td>
<td>1852</td>
<td>2.21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-1</td>
<td>0.0047219</td>
<td>293.2</td>
<td>1.101</td>
<td>1.030</td>
<td>1.069</td>
<td>6.268</td>
<td>1705</td>
<td>2.35</td>
</tr>
<tr>
<td>4-2</td>
<td>293.0</td>
<td>1.103</td>
<td>1.071</td>
<td>6.431</td>
<td>1665</td>
<td>2.42</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-3</td>
<td>293.2</td>
<td>1.101</td>
<td>1.069</td>
<td>6.282</td>
<td>1702</td>
<td>2.36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-4</td>
<td>293.0</td>
<td>1.103</td>
<td>1.071</td>
<td>6.372</td>
<td>1681</td>
<td>2.40</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* See notes on the following page.
Notes for table H-3.

Viscosities were estimated by first using Boeffard's equation to obtain the viscosity at 25 °C:

$$\mu(cps) = 0.96714 + 0.09622c_{NaOH} - 0.20528c_{Ferri} + 0.090255c_{Ferro}$$

and then using modified equations from the CRC Handbook to account for the temperature dependence. The original equations (for pure water) are

$$\log_{10}[\mu_T(cp)] = \frac{1301}{998.333 + 8.1855(T-20) + 0.00585(T-20)^2 - 3.330233 \quad (0 - 20 °C)}$$

$$\log_{10}\frac{\mu_T}{\mu_{20}} = \frac{1.3272(20-T) - 0.00153(T-20)^2}{T + 105} \quad (20 - 100 °C).$$

These equations are based on $\mu(20 °C)=1.002$ cp. Equation H-2 was modified by substituting $\mu_{25}$ into equation H-3 to obtain $\mu_{20}$ for the ferricyanide solution and adjusting the final constant (3.330233) to give the proper $\mu_{20}$.

The densities were estimated from Boeffard's correlation:

$$\rho(g/cm^3) = 0.99702 + 0.04423c_{NaOH} + 0.17118c_{Ferri} + 0.23119c_{Ferro} - 0.00133c_{NaOH}^2 - 0.00787c_{NaOH}c_{Ferro}$$

where the concentrations are in mol/l.
Table H-4. Solution concentrations for channel experiments (mol/liter).

<table>
<thead>
<tr>
<th>Solution</th>
<th>NaOH</th>
<th>K₃Fe(CN)₆</th>
<th>K₄Fe(CN)₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>1c</td>
<td>0.3190</td>
<td>0.01001</td>
<td>0.05009</td>
</tr>
<tr>
<td>1d</td>
<td>0.3415</td>
<td>0.01001</td>
<td>0.05109</td>
</tr>
<tr>
<td>1e</td>
<td>0.2832</td>
<td>0.01012</td>
<td>0.05758</td>
</tr>
<tr>
<td></td>
<td>0.2996</td>
<td>0.01002</td>
<td>0.050093</td>
</tr>
<tr>
<td>1f</td>
<td>0.301145</td>
<td>0.01004</td>
<td>0.05042</td>
</tr>
<tr>
<td>2</td>
<td>2.1091</td>
<td>0.04993</td>
<td>0.05008</td>
</tr>
</tbody>
</table>

† Boldface refers to solutions that were analyzed for ferricyanide by iodometric titration with zinc and for ferrocyanide by potentiometric titration with eerie sulfate. All other concentrations were calculated by weighing.
H-5. Measured parameters for the channel experiments

\(L = 2.370 \text{ in}, \ W = 2.007 \text{ in}\).

<table>
<thead>
<tr>
<th>Run</th>
<th>h (in)</th>
<th>Q (ml/min)</th>
<th>O.C.P. (mV)</th>
<th>V-V\text{ref} (mV)</th>
<th>I (mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1c-1</td>
<td>0.127</td>
<td>4.97</td>
<td>+2.0</td>
<td>-800</td>
<td>-4.09</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-600</td>
<td>-4.10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-400</td>
<td>-4.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-4.09</td>
</tr>
<tr>
<td>1c-2</td>
<td></td>
<td>4.91</td>
<td>+1.7</td>
<td>-100</td>
<td>-4.16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-4.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-600</td>
<td>-4.25</td>
</tr>
<tr>
<td>1c-3</td>
<td></td>
<td>4.98</td>
<td>+1.0</td>
<td>-600</td>
<td>-4.26</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-4.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-100</td>
<td>-4.17</td>
</tr>
<tr>
<td>1c-4</td>
<td></td>
<td>4.98</td>
<td>+1.1</td>
<td>-50</td>
<td>-3.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-20</td>
<td>-2.21</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-10</td>
<td>-1.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>-0.3</td>
</tr>
<tr>
<td>1c-5</td>
<td>0.127</td>
<td>—</td>
<td>+1</td>
<td>-600</td>
<td>-4.74</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-4.71</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-4.70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-100</td>
<td>-4.59</td>
</tr>
<tr>
<td>1c-6</td>
<td></td>
<td>6.90</td>
<td>+0.3</td>
<td>-50.1</td>
<td>-3.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-20</td>
<td>-2.29</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-9.9</td>
<td>-1.32</td>
</tr>
<tr>
<td>1c-7</td>
<td></td>
<td>6.80</td>
<td>0.0</td>
<td>-600</td>
<td>-4.70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-800</td>
<td>-4.70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1000</td>
<td>-4.76</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1200</td>
<td>-4.91</td>
</tr>
<tr>
<td>1c-8</td>
<td></td>
<td>—</td>
<td>-0.4</td>
<td>-35</td>
<td>-3.36</td>
</tr>
<tr>
<td>Run</td>
<td>h (in)</td>
<td>Q (ml/min)</td>
<td>O.C.P. (mV)</td>
<td>V-V&lt;sub&gt;ref&lt;/sub&gt; (mV)</td>
<td>I (mA)</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>------------</td>
<td>-------------</td>
<td>----------------</td>
<td>--------</td>
</tr>
<tr>
<td>1d-1</td>
<td>0.127</td>
<td>3.3</td>
<td>+1.0</td>
<td>-600</td>
<td>-3.75</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-400</td>
<td>-3.76</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-3.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-100</td>
<td>-3.66</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-50</td>
<td>-3.07</td>
</tr>
<tr>
<td>1d-2</td>
<td></td>
<td>3.35</td>
<td>+0.8</td>
<td>-20</td>
<td>-1.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>-0.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-600</td>
<td>-3.73</td>
</tr>
<tr>
<td>1d-3</td>
<td></td>
<td>3.69</td>
<td>+0.0</td>
<td>-800</td>
<td>-3.68</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1000</td>
<td>-3.73</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1200</td>
<td>-3.87</td>
</tr>
<tr>
<td>1d-4</td>
<td></td>
<td>3.23</td>
<td>—</td>
<td>-600</td>
<td>-3.66</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-35</td>
<td>-2.62</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-9.9</td>
<td>-1.16</td>
</tr>
<tr>
<td>1d-5</td>
<td></td>
<td>3.21</td>
<td>+3.5</td>
<td>-10</td>
<td>-1.19</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-35</td>
<td>-2.59</td>
</tr>
<tr>
<td>1d-6</td>
<td></td>
<td>3.23</td>
<td>-3.0</td>
<td>-1200</td>
<td>-3.71</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-800</td>
<td>-3.83</td>
</tr>
<tr>
<td>1e-1</td>
<td>0.0178</td>
<td>16.71</td>
<td>+2.0</td>
<td>-400</td>
<td>-22.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-600</td>
<td>-22.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-800</td>
<td>-23.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1000</td>
<td>-23.1</td>
</tr>
<tr>
<td>1e-2</td>
<td></td>
<td>17.02</td>
<td>+1.0</td>
<td>-200</td>
<td>-23.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-100</td>
<td>-22.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-23.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-50</td>
<td>-19.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-35</td>
<td>-16.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-20</td>
<td>-11.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-10.1</td>
<td>-6.75</td>
</tr>
<tr>
<td>1e-3</td>
<td>0.0178</td>
<td>16.85</td>
<td>—</td>
<td>-400</td>
<td>-23.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-600</td>
<td>-23.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-800</td>
<td>-23.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1000</td>
<td>-23.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1200</td>
<td>-24.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1400</td>
<td>-48.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-400</td>
<td>-23.1</td>
</tr>
<tr>
<td>Run</td>
<td>h (in)</td>
<td>Q (ml/min)</td>
<td>O.C.P. (mV)</td>
<td>V-V&lt;sub&gt;ref&lt;/sub&gt; (mV)</td>
<td>I (mA)</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>------------</td>
<td>------------</td>
<td>------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>1e-4</td>
<td>16.77</td>
<td>-1.0</td>
<td>-400</td>
<td>-22.7</td>
<td>-22.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-600</td>
<td>-22.7</td>
<td>-22.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-400</td>
<td>-22.7</td>
<td>-22.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-800</td>
<td>-22.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1100</td>
<td>-22.5</td>
<td></td>
</tr>
<tr>
<td>1f-1</td>
<td>0.0178</td>
<td>0.21</td>
<td>+2.7</td>
<td>-20</td>
<td>-4.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-10</td>
<td>-3.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-5</td>
<td>-2.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-35</td>
<td>-6.28</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-50</td>
<td>-7.08</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-65</td>
<td>-7.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-79.9</td>
<td>-7.82</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-95</td>
<td>-7.96</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-110</td>
<td>-8.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-130</td>
<td>-8.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-180</td>
<td>-8.17</td>
</tr>
<tr>
<td>1f-2</td>
<td>0.4</td>
<td>+2.7</td>
<td>-180</td>
<td>-8.65</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-160</td>
<td>-8.70</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-120</td>
<td>-8.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-90</td>
<td>-8.40</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-70</td>
<td>-8.11</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-50</td>
<td>-7.45</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-35</td>
<td>-6.55</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-20</td>
<td>-5.05</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-10</td>
<td>-3.63</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-180</td>
<td>-8.75</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-230</td>
<td>-8.75</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-300</td>
<td>-8.76</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-400</td>
<td>-8.82</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-480</td>
<td>-8.85</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-600</td>
<td>-8.88</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-800</td>
<td>-8.85</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-900</td>
<td>-8.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1000</td>
<td>-9.00</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1200</td>
<td>-9.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1300</td>
<td>-10.65</td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td>h (in)</td>
<td>Q (ml/min)</td>
<td>O.C.P. (mV)</td>
<td>V-V\textsubscript{ref} (mV)</td>
<td>I (mA)</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>------------</td>
<td>-------------</td>
<td>-------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>2-1</td>
<td>0.127</td>
<td>4.94</td>
<td>+1</td>
<td>-600</td>
<td>-16.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-400</td>
<td>-16.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-200</td>
<td>-16.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-100</td>
<td>-15.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-50</td>
<td>-11.4</td>
</tr>
<tr>
<td>2-2</td>
<td>5.04</td>
<td>+1</td>
<td>-35</td>
<td>-10.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-20</td>
<td>-6.73</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-10</td>
<td>-4.06</td>
<td></td>
</tr>
<tr>
<td>2-3</td>
<td>4.94</td>
<td>-6</td>
<td>-600</td>
<td>-17.0</td>
<td></td>
</tr>
<tr>
<td>2-4</td>
<td>7.01</td>
<td>—</td>
<td>-600</td>
<td>-18.8</td>
<td></td>
</tr>
<tr>
<td>2-5</td>
<td>3.57</td>
<td>—</td>
<td>-600</td>
<td>-15.3</td>
<td></td>
</tr>
<tr>
<td>2-6</td>
<td>5.05</td>
<td>-2</td>
<td>-600</td>
<td>-17.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-800</td>
<td>-16.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1000</td>
<td>-16.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1200</td>
<td>-17.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1400</td>
<td>-17.5</td>
<td></td>
</tr>
<tr>
<td>2-7</td>
<td>7.0</td>
<td>—</td>
<td>-600</td>
<td>-18.8</td>
<td></td>
</tr>
</tbody>
</table>
### Table II-6. Calculated dimensional parameters for the channel experiments.*

<table>
<thead>
<tr>
<th>Run</th>
<th>(c_{\text{Fe}^2+} (\text{mol/L}))</th>
<th>(c_{\text{Fe}^3+} (\text{mM/L}))</th>
<th>(\mu (20^\circ\text{C})) (cp)</th>
<th>(\rho) (g/cc)</th>
<th>(\Psi) (cp)</th>
<th>(\phi^{-1} Y \cdot 1) cm (^{-1})</th>
<th>(&lt;v&gt;) (cm/s)</th>
<th>(I_{111}) (mA)</th>
<th>(D \times 10^6) (cm (^{2}) /s)</th>
<th>(h D / T \times 10^{10}) (dyne/K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1c</td>
<td>0.01001</td>
<td>0.05009</td>
<td>1.13935</td>
<td>1.024107</td>
<td>1.11253</td>
<td>0.1212</td>
<td>0.050270</td>
<td>4.2</td>
<td>5.93</td>
<td>2.31</td>
</tr>
<tr>
<td>Id</td>
<td>0.01001</td>
<td>0.05109</td>
<td>1.14340</td>
<td>1.025297</td>
<td>1.11519</td>
<td>0.1275</td>
<td>0.069426</td>
<td>4.7</td>
<td>5.93</td>
<td>2.31</td>
</tr>
<tr>
<td>1e</td>
<td>0.01012</td>
<td>0.05758</td>
<td>1.13532</td>
<td>1.024301</td>
<td>1.10838</td>
<td>0.1179</td>
<td>0.033345</td>
<td>3.7</td>
<td>5.94</td>
<td>2.32</td>
</tr>
<tr>
<td>1f</td>
<td>0.01004</td>
<td>0.05042</td>
<td>1.12647</td>
<td>1.023422</td>
<td>1.11046</td>
<td>0.1171</td>
<td>1.2177</td>
<td>23.2</td>
<td>5.91</td>
<td>2.29</td>
</tr>
<tr>
<td>2-1</td>
<td>0.04993</td>
<td>0.05008</td>
<td>1.568597</td>
<td>1.0949</td>
<td>1.43264</td>
<td>0.5855</td>
<td>0.015186</td>
<td>8.2</td>
<td>5.93</td>
<td>2.30</td>
</tr>
<tr>
<td>2-2</td>
<td>0.051081</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.028925</td>
<td>8.8</td>
<td>5.93</td>
<td>2.30</td>
</tr>
<tr>
<td>2-3</td>
<td></td>
<td>0.050068</td>
<td>17.0</td>
<td></td>
<td></td>
<td></td>
<td>15.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-4</td>
<td></td>
<td>0.071047</td>
<td>18.8</td>
<td></td>
<td></td>
<td></td>
<td>15.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-5</td>
<td></td>
<td>0.36182</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15.3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Boldface refers to solutions that were analyzed for ferricyanide by iodometric titration with zinc and for ferrocyanide by potentiometric titration with ceric sulfate. All other concentrations were calculated by weighing.
Table H-7. Calculated dimensionless parameters for the channel experiments.

<table>
<thead>
<tr>
<th>Run</th>
<th>( \frac{c_{\text{Ferro}}}{c_{\text{Ferr}}} )</th>
<th>J</th>
<th>N</th>
<th>( \delta )</th>
<th>Sc</th>
<th>Re</th>
<th>( \frac{h}{L} )</th>
<th>( \frac{d}{L} ) Pe</th>
<th>Nu_{lev}</th>
<th>Nu_{exp}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1c</td>
<td>5.00244</td>
<td>400.86</td>
<td>0.32883</td>
<td>0.26539</td>
<td>1876</td>
<td>2.915</td>
<td>0.053763</td>
<td>586.15</td>
<td>15.472</td>
<td>15.417</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.36619</td>
<td>0.29642</td>
<td>1876</td>
<td>4.026</td>
<td>1876</td>
<td>9.935</td>
<td>0.0075353</td>
<td>388.14</td>
</tr>
<tr>
<td>1d</td>
<td>5.10168</td>
<td></td>
<td>0.27291</td>
<td>0.21911</td>
<td>1877</td>
<td>1.929</td>
<td>0.21911</td>
<td>3.4738</td>
<td>12.093</td>
<td>11.846</td>
</tr>
<tr>
<td>1e</td>
<td>5.68972</td>
<td></td>
<td>1.8993</td>
<td>1.5230</td>
<td>1875</td>
<td>9.935</td>
<td>1875</td>
<td>9.935</td>
<td>0.0075353</td>
<td>279.87</td>
</tr>
<tr>
<td>1f</td>
<td>5.01899</td>
<td></td>
<td>0.44094</td>
<td>0.54131</td>
<td>1873</td>
<td>0.1237</td>
<td>0.54131</td>
<td>3.4738</td>
<td>2.8012</td>
<td>4.1907</td>
</tr>
<tr>
<td>2-1</td>
<td>1.00300</td>
<td>450.0</td>
<td>0.27552</td>
<td>—</td>
<td>2646</td>
<td>2.8254</td>
<td>0.053763</td>
<td>801.36</td>
<td>17.172</td>
<td>16.769</td>
</tr>
<tr>
<td>2-2</td>
<td></td>
<td></td>
<td>0.27636</td>
<td>2.8826</td>
<td>817.58</td>
<td>17.288</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>2-3</td>
<td></td>
<td></td>
<td>0.27452</td>
<td>2.8254</td>
<td>801.36</td>
<td>17.172</td>
<td>17.172</td>
<td>17.173</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>2-4</td>
<td></td>
<td></td>
<td>0.30849</td>
<td>4.0093</td>
<td>1137.1</td>
<td>19.297</td>
<td>18.991</td>
<td>18.991</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>2-5</td>
<td></td>
<td></td>
<td>0.24635</td>
<td>2.0418</td>
<td>579.12</td>
<td>15.410</td>
<td>15.455</td>
<td>15.455</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>
References


12. W. Kroenig and J. Grolig (to Farbenfabriken Bayer A.-G.), U.S. Pat. 3,451,905


43. Franz Wenisch, Heinz Nohe, Heinz Hannebaum, R. K. Horn, Manfred Stroezel,


67. J. R. Sellars, Myron Tribus, and J. S. Klein, "Heat Transfer to Laminar Flow
in a Round Tube or Flat Conduit - The Graetz Problem Extended," *Transactions of the ASME*, **78** (February, 1956), 441-448.


dissertation, University of California, Berkeley (June 1971), UCRL-20557.


98. Klaus J. Vetter, *Electrochemical Kinetics — Theoretical and Experimental*


110. Professor David Graves, personal communication (April, 1986)
111. Professor John Newman, personal communication (June, 1986).

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.