Lawrence Berkeley National Laboratory

Recent Work

Title

Cyclic Voltammetry at a Rotating Disk, Electroreduction of Nitrate in Acidic Nickel Solutions, and Frequency-Response Analysis of Porous Electrodes

Permalink

https://escholarship.org/uc/item/93p5b3k7

Author

Shain, P.M.

Publication Date

1990-06-01

eScholarship.org Powered by the California Digital Library University of California

UC 404

LBL-29118

Lawrence Berkeley Laboratory

UNIVERSITY OF CALIFORNIA

Materials & Chemical Sciences Division

Cyclic Voltammetry at a Rotating Disk, Electroreduction of Nitrate in Acidic Nickel Solutions, and Frequency-Response Analysis of Porous Electrodes

P.M. Shain

(Ph.D. Thesis)

June 1990

For Reference

Not to be taken from this room

LBL-29118
Copy 1
Bldg. 50 Library.

Prepared for the U.S. Department of Energy under Contract Number 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.

LBL-29118

Cyclic Voltammetry at a Rotating Disk, Electroreduction of Nitrate in Acidic Nickel Solutions, and Frequency-Response Analysis of Porous Electrodes

by

Paul Mark Shain

June 1990

This work was supported by the Assistant Secretary for Conservation and Renewable Energy, Office of Energy Storage and Distribution, Energy Storage Division, of the U.S. Department of Energy (DOE) under Contract No. DE-AC03-76SF00098.

Cyclic Voltammetry at a Rotating Disk,
Electroreduction of Nitrate in Acidic Nickel Solutions,
and
Frequency-Response Analysis of Porous Electrodes

by

Paul Shain

Abstract

Models of electrochemical systems were developed to interpret experimental results. Linear sweep voltammetry and frequency-response analysis (electrochemical impedance spectroscopy) were the experimental techniques considered; the reduction of nitrate in acidic nickel solutions and the frequency response of redox couples in porous electrodes, the electrochemical systems.

Linear sweep voltammetry has been used to study the kinetics of nitrate reduction in acidic nickel solutions. This is a step in the manufacture of nickel battery electrodes and would also be important in the corrosion of steel by nitric acid. The reaction is interesting because negative ions react at the negative electrode.

Bernardi studied this reaction with potential step and linear sweep voltammetry experiments. The voltammetry model was developed to explain the results of the latter. It was based on earlier work by Matlosz and Newman but allows for the participation of adsorbed species in the reactions. The assumption of a stagnant-diffusion layer was relaxed. The model accounts properly for the beginning of the sweep before the boundary layer develops.

The analysis showed that the transfer coefficient for desorption must be greater than that for reduction to insure the presence of a peak in the voltammogram. Two possible explanations for the presence of the peak: catalysis by underpotential-deposited nickel and double-layer effects are ruled out.

Frequency-response analysis was applied to a porous electrode system related to the NASA iron/chromium redox battery. Porous electrodes (including porous flow-through electrodes and redox batteries) and the frequency-response analysis of electrochemical systems are reviewed briefly.

The results of frequency response experiments are often analyzed in terms of analogous electric circuits. A model of the frequency response of porous electrodes based on a first-principles approach is presented. The model concerns porous electrodes with and without flow of reactant through the electrode. The frequency-response model is based on the steady-state models of Newman and coworkers but has been modified to account for small sinusoidal time variations of the variables.

Results of the model are compared to experiment, and the effect of various dimensionless parameters on the frequency response is explored.

John Newman

Et je me disais, avec ravissement, Voilà une chose que je pourrai étudier toute ma vie sans jamais la comprendre.

— Samuel Beckett, *Molloy*

To Wendy, for her encouragement and sacrifices

Acknowledgements

I have learned much from working with Professor Newman. I am grateful for opportunities that most graduate students don't have and for John's encouragement and patience.

I thank Professors Scott Lynn and Marcin Majda for the time they spent reviewing this manuscript and for their helpful suggestions.

Some of the folks who have helped in this work or enhanced my life while I've been working on this include my buddies Dale and Deborah; members of the Newman Group: Alan H., Alan W., Chrys, Dawn (who suggested the nitrate reduction problem), Gregg (who helped build the flow cell), John S., Masao, Mike, Phil, Sue G., Sue T., Suresh, Tom, Vicki, and Vince; and others: Anne-Marie, Brian, Cathy, Claudia, Gina, Jacques, Jennifer, John D., Karrie, Kathy, Ken, Lucy, Mariette, Mark I., Mark V., Rajiv, Sholeh, Stan, Steve, and Sue.

Family is very important. Mine has provided much encouragement, support, and diversion—and many meals. In the last few years it has grown to include Wendy's and David.

This work was supported by the Assistant Secretary for Conservation and Renewable Energy, Office of Energy Storage and Distribution, Energy Storage Division, of the U. S. Department of Energy (DOE) under Contract No. DE-AC03-76SF00098.

Contents

CHAPTER 1

Overview and List of Symbols

This dissertation deals with two subjects: studying the reduction of nitrate in acidic nickel solutions with linear sweep voltammetry (Chapters 2 through 4) and the frequency-response analysis of flow-through porous electrodes (the following chapters). Mathematical models are developed to provide tools for studying these two distinct experimental problems.

In Chapter 2 experiments on the reduction of nitrate ion in acidic nitrate solutions with and without nickel are discussed. These experiments give some clues about what the reaction mechanism is and eliminate possible mechanisms. In Chapter 3 a model is developed to simulate the linear sweep voltammetry experiments. Following this chapter are several appendices containing the solutions to problems needed for developing the programs used and listings of the programs. This model is used in Chapter 4 to simulate the experiments described in Chapter 2.

The rest of the thesis is concerned with a different problem: the frequency-response or electrochemical impedance analysis of flow-through porous electrodes. Chapter 5 contains a discussion of this type of electrochemical reactor, its practical uses, redox batteries in general, and the iron-chromium redox battery in particular. The next chapter provides some background on electrochemical impedance measurements and our philosophy of modeling such systems.

In Chapter 7 experiments on a rotating-disk electrode to determine diffusivities of the electroactive species in an iron chloride and hydrochloric acid solution are presented. Chapter 8 presents the development of a model for the simulation of impedance experiments on a flow-through porous electrode. Measurements of the impedance of the ferrous/ferric chloride redox reaction in a flow-through porous electrode are discussed and compared to the above-mentioned model in Chapter 9. The effect of the magnitude of several dimensionless groups on the model's predictions is explored also.

Literature references are found at the end of each chapter or appendix.

The definitions of symbols used in this work follow. Some symbols have been used in more than one context.

List of Symbols

a	integral used as a coefficient in calculating superposition integrals; hydrodynamic constant 0.51023; bed specific area, cm^{-1}
a'	dimensionless form of a
A	area, cm^2; integral used as a coefficient in calculating superposition integrals
B_n	nth constant
c	concentration, mole/cm^3
C	capacity or double-layer capacity, F; transmission line capacitance, F/cm
C_{dl}	dimensionless double-layer capacity
$c_{o,i}$	initial concentration of species i, mole/cm^3
D	diffusion coefficient, cm^2/s
D_a	axial dispersion coefficient, cm^2/s
erf(x)	error function of x
erfc(x)	error function complement of x
F	Faraday's constant, 96,487 C/equiv
G	transmission line leakage, S/cm
i	current or current density, A or A/cm^2
I	current or current density, A or A/cm^2
i_j	partial current density of reaction j, A/cm^2
i_o	exchange current density, A/cm^2
I^*	dimensionless current density
j	$\sqrt{-1}$; flux, mol/cm^{2}s
k_m	mass transfer coefficient, cm/s

L	bed length, cm; transmission line inductance, H/cm
n	number of electrons taking part in a reaction
n_j'	charge transferred between phases in reaction j
N	flux, mol/cm^{2}s
P_n	dimensionless group
Pe	Peclet number
q	charge, C
r_o	radius of disk electrode, cm
R	transmission resistance, Ω/cm
s	Laplace transform variable; stoichiometric coefficient
Sh	Sherwood number
t	time, s
U^θ	open circuit potential for secondary reference state at infinite dilution, V
v	potential, V; superficial velocity, cm/s
V	potential, V
v_y	axial velocity near the surface of a rotating disk, cm/s
V_{surf}	applied potential, V
V_Ω	ohmic potential drop, V
x	distance, cm
y	distance, cm; dimensionless distance
z_i	charge of species i
Z	impedance, Ω or Ωcm^2
Z_n	nth eigenfunction
α	transfer coefficient; penetration depth, cm
β	combination of transfer coefficients
Γ_{max}	total surface concentration of sites, mole/cm^2

$\Gamma(x)$	gamma function of x
δ_i	Nernst diffusion layer thickness for species i, cm
Δ	a combination of terms
ΔG_f^o	gibbs energy of formation
ΔU	difference in open-circuit potentials between the side and main reactions, V
ε	bed porosity
ζ	dimensionless distance
η	distance variable; overpotential, V
θ_i	dimensionless time, dimensionless concentration
Θ_i	surface concentration of species i, mole/cm^2
κ	solution conductivity, S/cm
λ_n	nth eigenvalue
μ^θ	electrochemical potential of species i for secondary reference state at infinite dilution (gibbs energy of formation), J/mole
ν	kinematic viscosity, cm^2/s
π	3.14159...
σ	bed conductivity, S/cm
τ	time, s; dimensionless time
ϕ	phase shift, rad
Φ	potential, V
ω	rotation rate or frequency, s^{-1}
Ω	rotation rate, s^{-1}; dimensionless frequency
subscripts	
a	anodic
b	bulk value
c	cathodic
f	feed stream

i	species index
j	reaction index
ℓ	limiting current
n	normal direction
P	product
R	reactant, main reaction
S	side reaction
w	wall
1	matrix phase
2	solution phase

superscripts

old	value for the previous time step
ref	reference or at reference concentration
ss	steady state
t	transient
′	first derivative, dimensionless form
″	second derivative

over marks

—	time-average or steady-state value, Laplace transform, or response to a unit change in concentration at the boundary
~	time-varying value (a complex function of position)

CHAPTER 2

Electrochemical Reduction of Nitrate in Acidic Nickel Solutions

To understand the mechanism of the electroprecipitation of nickel hydroxide from acidic, nickel solutions, Bernardi conducted an experimental investigation of nickel nitrate reduction with a rotating-disk electrode.[1,2] Nickel hydroxide precipitation has practical importance: nickel hydroxide is the active material in nickel battery electrodes. Knowledge of the reduction of nitrate, in addition to being used to produce nickel battery electrodes,[3,4] can be used to synthesize hydroxylamine and to protect metals from corrosion by nitric acid.[4-8]

Nickel battery electrodes are produced by impregnating a porous electrode with nickel nitrate and then reducing the nitrate to increase the pH of the solution near the electrode. If the pH increases sufficiently, nickel hydroxide precipitates.

The reduction of nitrate can be written as producing hydroxide ions or as consuming protons:

$$NO_3^- + H_2O + 2e^- \rightarrow NO_2^- + 2OH^- \tag{1}$$

or

$$NO_3^- + 3H^+ + 2e^- \rightarrow HNO_2 + H_2O\ . \tag{2}$$

The increase in pH will cause nickel hydroxide to precipitate when its solubility product (($K_{sp} = c_{Ni^{2+}} c_{OH^-}^2$) is exceeded:

$$Ni^{2+} + 2OH^- \rightarrow Ni(OH)_2\ . \tag{3}$$

Bernardi conducted both potential step and linear sweep voltammetry experiments under conditions at which no precipitation occurs. She proposed a three-step mechanism: adsorption of nitrate, reduction of nitrate to an adsorbed intermediate, and reduction of the intermediate to a soluble product. It was not assumed that any of these three steps was elementary.

After Bernardi's and my experimental results are discussed in this chapter, a mathematical model of the experimental system is developed in Chapter 3 which is used to examine the mechanism of the reduction of nitrate in Chapter 4.

2.1. Bernardi's experimental procedure

In her voltammetry experiments, Bernardi used a Stonehart BC1200 potentiostat, a Princeton Applied Research 175 universal programmer, a Hewlett-Packard 7047A x-y plotter, and a Nicolet 206 digital oscilloscope. Her working electrode was a 0.764 cm diameter glassy carbon disk in a 2 cm diameter insulating shaft. She ramped the potential from -0.1 V to -0.7 V *versus* a saturated calomel reference electrode (SCE) at a constant rate while the disk rotated at a constant speed. The electrode was polished with successively finer grades of diamond paste—9 μm, 3 μm, 1 μm. The polished electrode was dipped for about ten seconds in each of three solutions: 70% nitric acid, carbon tetrachloride, and 70% nitric acid. Each dip was followed by a rinse with purified water. The polished, dipped, and rinsed electrode was cycled at 1 V/s in 1 M nitric acid, rinsed, and placed in the experimental cell. To obtain reproducible results, it was necessary to dip the electrode after each experiment and to polish it after every one or two experiments. The counterelectrode was a platinum screen. The nickel nitrate and nitric acid solution in the experimental cell was sparged with nitrogen for two hours before the experiment to remove oxygen. The water was purified by filtration and reverse osmosis.[9]

2.2. Bernardi's experimental results

The results of Bernardi's linear sweep voltammetry experiments are given in Chapter 7 of her dissertation.[10] The experiments were repeated under different conditions; however the base case was a solution composition of 2.56 M nickel nitrate and 0.006 M nitric acid, a rotation speed of 800 rpm, and a sweep rate of 1 mV/s. As in any well-controlled experiments, only one parameter was changed at a time while the others were maintained at the base conditions. The other experiments included a 5 mV/s sweep rate; 0.06, $5 \cdot 10^{-4}$, and 10^{-5} M nitric acid concentrations; 1.5 and 0.5 M nickel nitrate concentrations; and a

400 rpm rotation speed.

The results of these experiments, shown in Figures 7-5 through 7-10 in Bernardi's dissertation,[11] are reproduced here as Figures 2-1 through 2-6.

To explain the experimental results, Bernardi proposed that nitrate was adsorbed on the electrode and reduced to nitrite that was then further reduced. This explained the peak with a shoulder. At more cathodic potentials hydrogen production would explain the increase in current. The model discussed in the next chapter was developed to account for all phenomena relevant to such a mechanism.

Information gleaned from examining the experimental results of Figures 2-1 to 2-6 should help in the modeling of the experiments.

Figure 2-1 shows that the peak disappears at high sweep rates. This suggests that at the high sweep rate something necessary to producing the peak is not being allowed time to happen. The steady-state results shown in Figure 2-6 show that steady-state phenomena produce a peak. During the modeling of the experiments, we may try to find parameters by fitting the steady-state results.

Figures 2-2 and 2-3 show that the peak heights decrease with decreasing electrolyte concentration. The one result that seems anomalous is that for 10^{-5} M nitric acid. It is not clear from the figure whether the peak has disappeared and the hydrogen reaction has shifted to the left or there is a huge peak. Simulations suggest that the latter is the case. This curve is reproducible. There seems to be a slight shoulder that may be what is left of the peak.

Figure 2-4 shows that the reaction rate increases with rotation speed. However the increase is less than a factor of $\sqrt{2}$ for a doubling of the rotation speed, so the supply of the soluble reactant is not mass transfer limited.

Figure 2-5 shows that a solution that has been used gives a larger current than a freshly made solution. This shows that there is an intermediate reactant produced during the first experiment that augments the current during the next experiment.

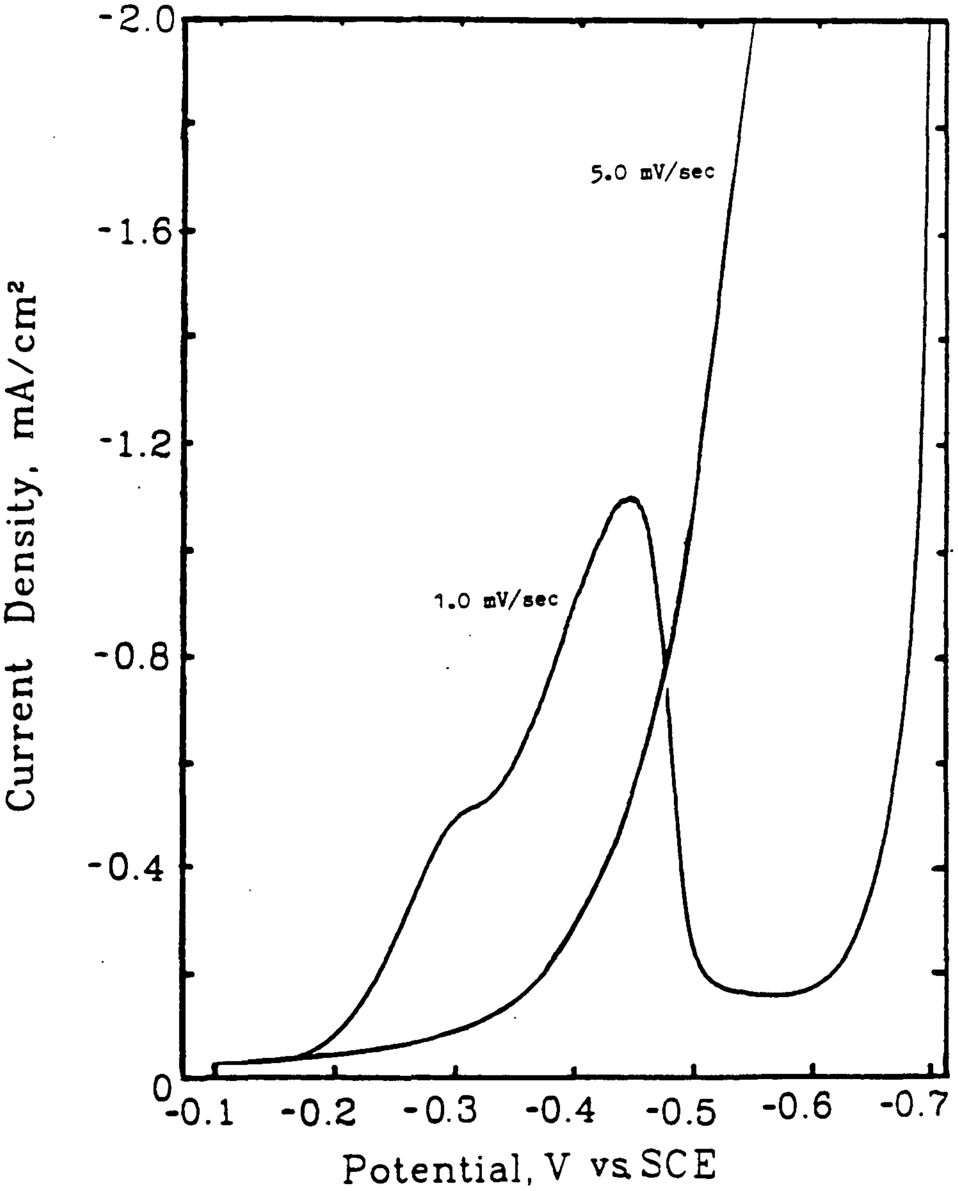


Figure 2-1. Current density on a glassy carbon rotating-disk electrode as a function of applied potential at two different sweep rates. The solution was 2.56 M $Ni(NO_3)_2$, 0.006 M HNO_3; the rotation speed, 800 rpm. The sweep rate for the other figures is 1 mV/s unless otherwise noted.[11]

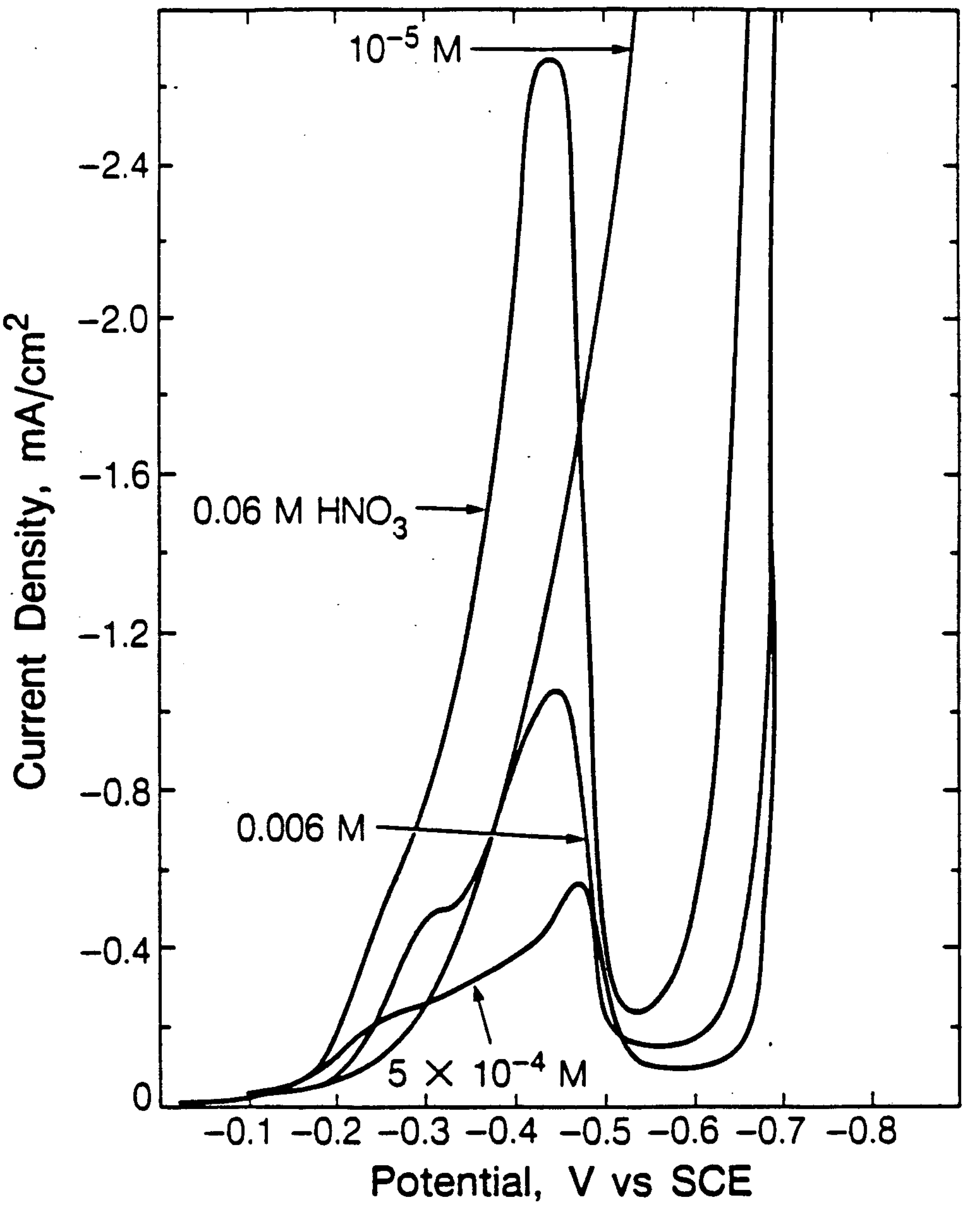
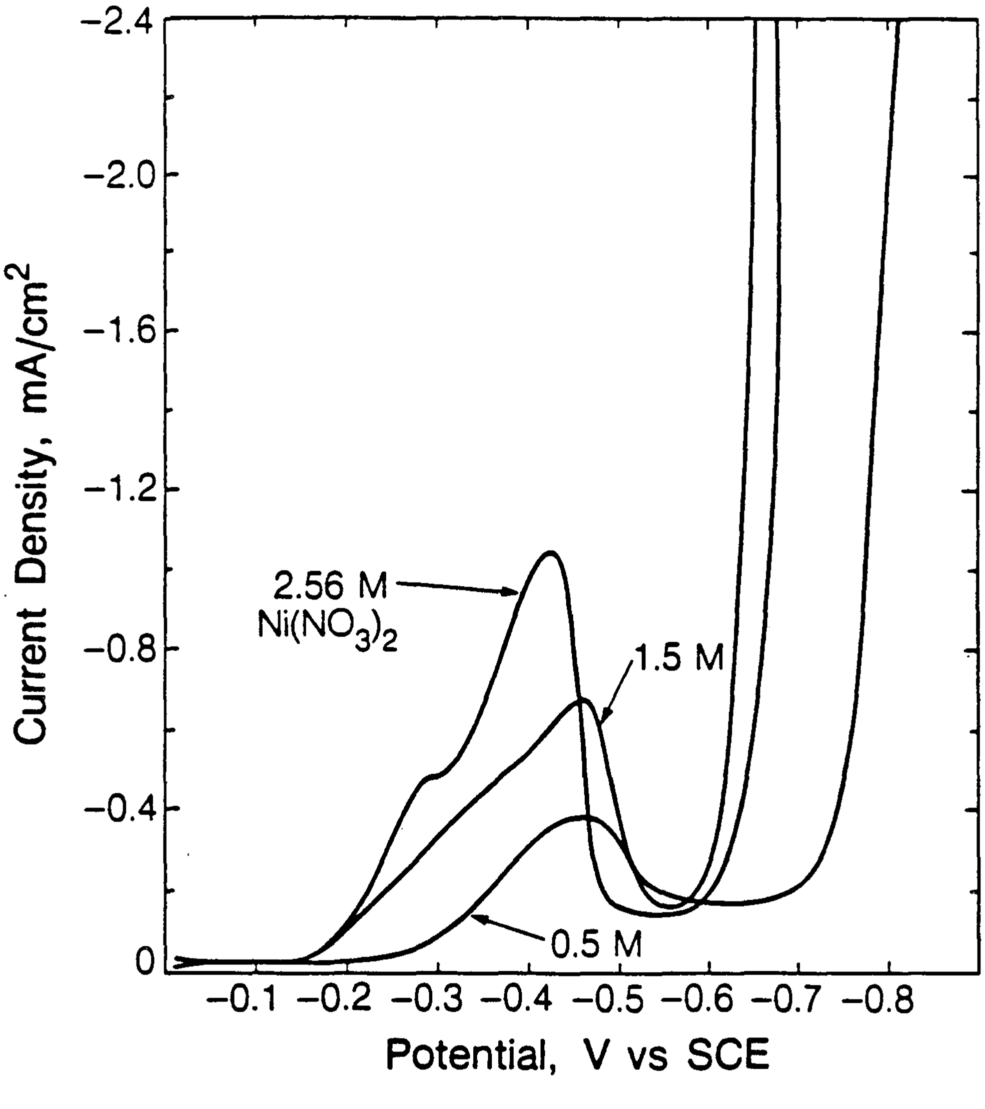


Figure 2-2. Current density on a glassy carbon rotating-disk electrode as a function of applied potential with different HNO_3 concentrations. Other conditions as in Figure 2-1.[11]



XBL 861-9411

Figure 2-3. Current density on a glassy carbon rotating-disk electrode as a function of applied potential with different $Ni(NO_3)_2$ concentrations. Other conditions as in Figure 2-1.[11]

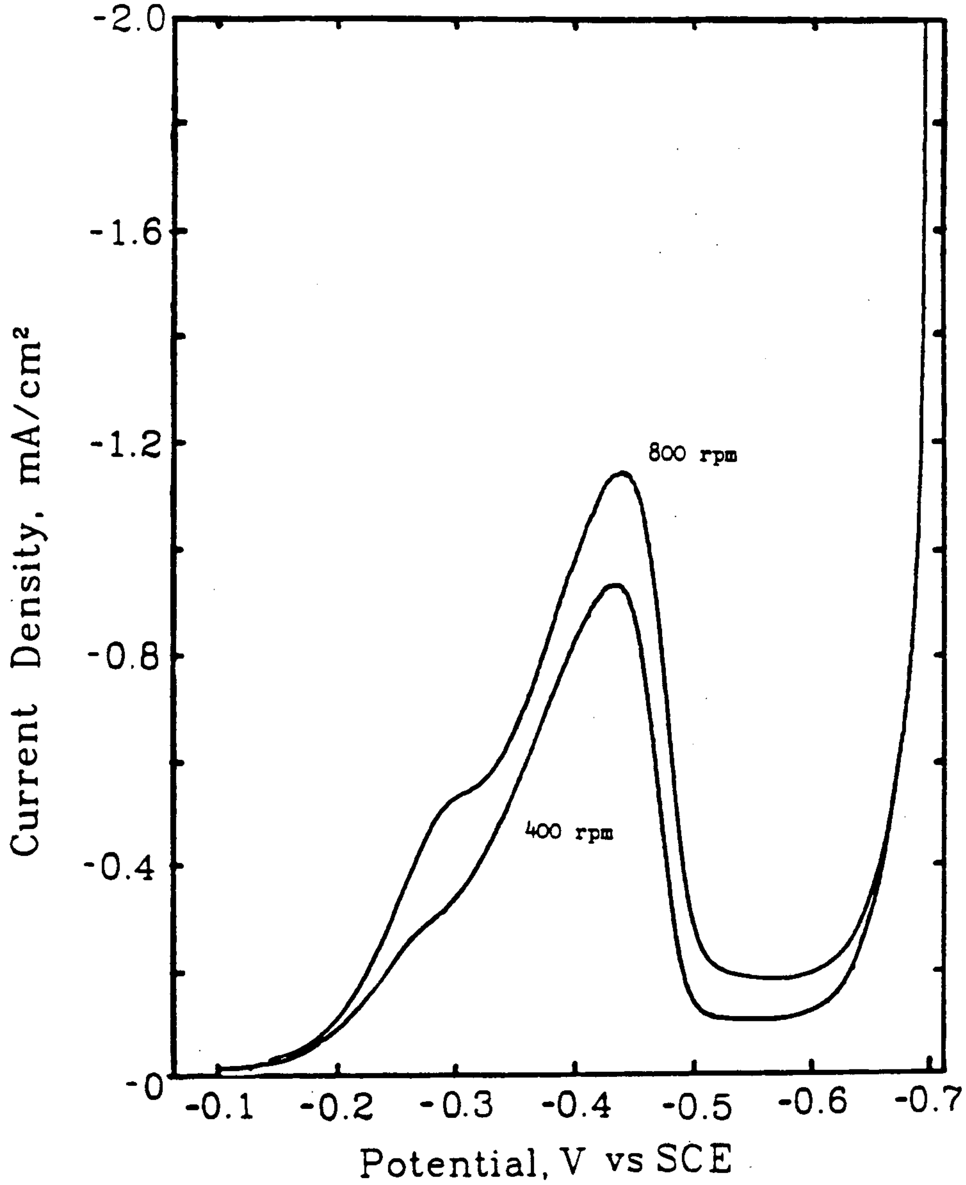


Figure 2-4. Current density on a glassy carbon rotating-disk electrode as a function of applied potential at two different rotation speeds. Other conditions as in Figure 2-1.[11]

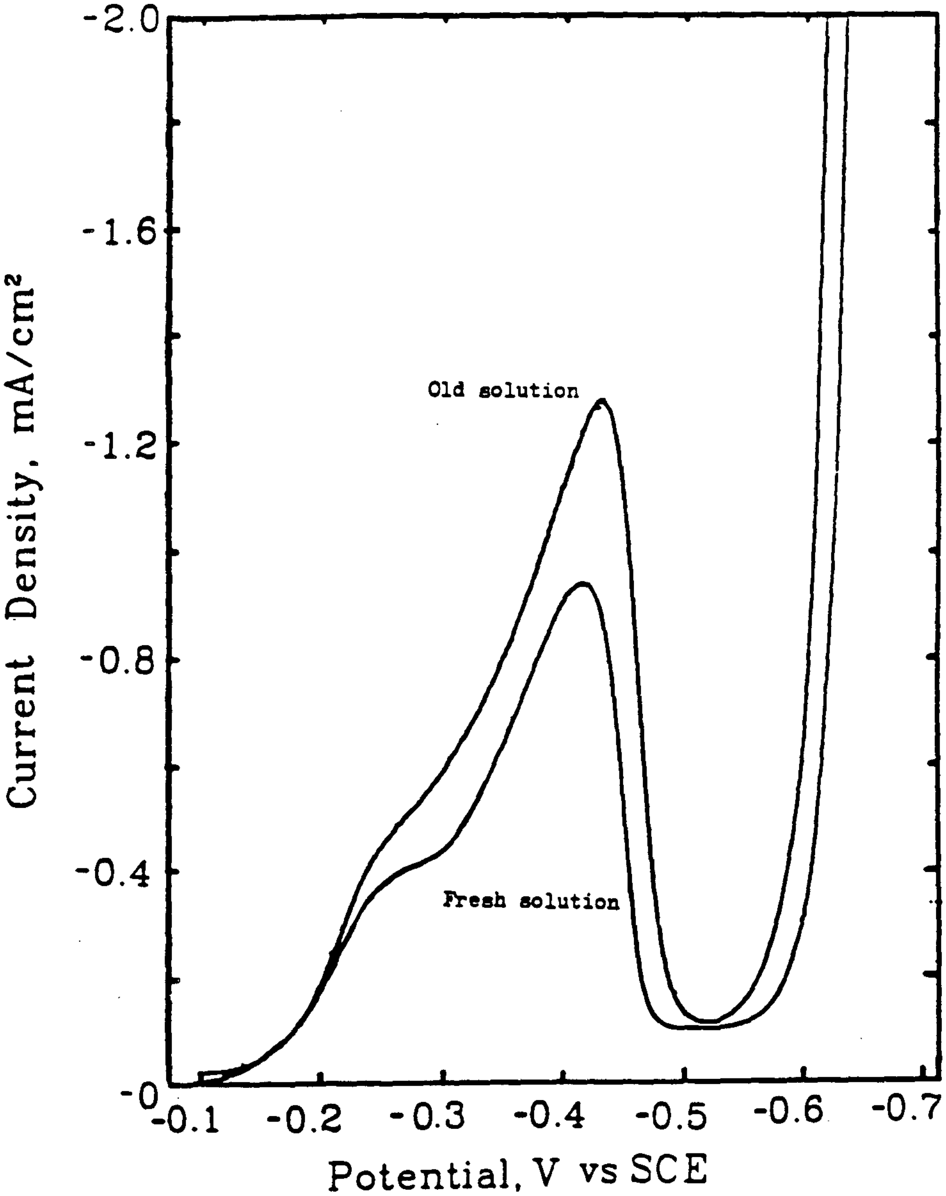


Figure 2-5. Current density on a glassy carbon rotating-disk electrode as a function of applied potential for fresh and previously used electrolyte solutions. The rotation speed was 600 rpm and the other conditions were as in Figure 2-1.[11]

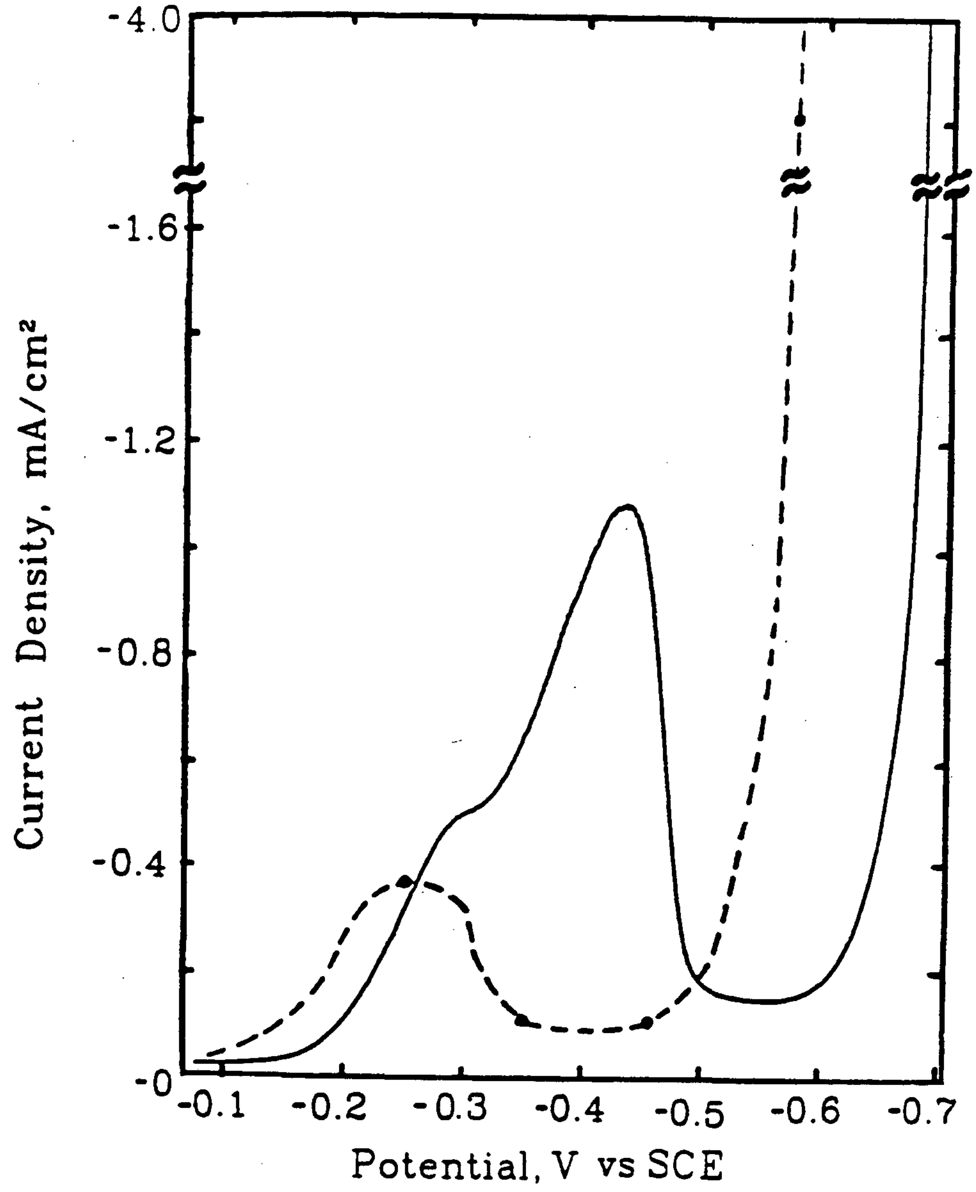


Figure 2-6. Current density on a glassy carbon rotating-disk electrode as a function of applied potential for a potential sweep experiment (solid curve) and for potential step experiments (dots connected by dashed curve). Other conditions as in Figure 2-1.[11]

The area under the peak is too large to be accounted for by the reduction of a monolayer of adsorbed nitrate. Therefore nitrate must adsorb while it is being reduced.

To ensure that nitrate desorbs sufficiently fast at the higher cathodic potentials to stop the reaction, the transfer coefficient for the reduction must be less than that for adsorption of nitrate.

The explanation that we have accepted for the presence of a peak in the current is that nitrate is adsorbed and available for reduction until the potential reaches a sufficiently high value that it can no longer be adsorbed. On the other hand, the peak resembles that for passivation of a corroding surface and could be caused by the formation of a film (perhaps of nickel hydroxide) on the surface. Since Bernardi did not observe the formation of a film on the electrode and claims that the conditions of the experiment do not lead to formation of a film, we discount this possibility.

A possible explanation of the cause of the peak is the structure of the double layer. This is an effect discussed by Frumkin in the 1950's.[12-14] It is possible that the peak caused by the reduction of an anion represents a limiting current that then decreases at more cathodic potentials because the anions are repelled from the electrode. If the concentration of electrolyte is increased, the current reduction becomes less pronounced because the anions are shielded from the electrode. We see no evidence of this concentration effect in Figures 2-1 and 2-2. In addition, if the observed peak is a limiting current it would have to represent the reduction of hydroxide to hydride—a reaction that seems unlikely.

2.3. Experiments without nickel

Another possible explanation for the experimental results is that nickel deposited on the electrode catalyzes the reduction reaction. The experiments described in this section were intended to test this hypothesis. If the nickel deposits as islands, the perimeter of these islands may have catalytic activity that vanishes after a monolayer is formed. From the experimental conditions and the Pourbaix diagram for nickel[15] shown schematically in Figure 2-7 we would not expect nickel ions to be reduced to nickel [Ni^{2+}(aqueous) + $2e^-$ → Ni(solid)]. For potentials where water is stable in acidic solutions, nickel dissolves. However we do not yet rule out underpotential deposition. To determine if the observed behavior

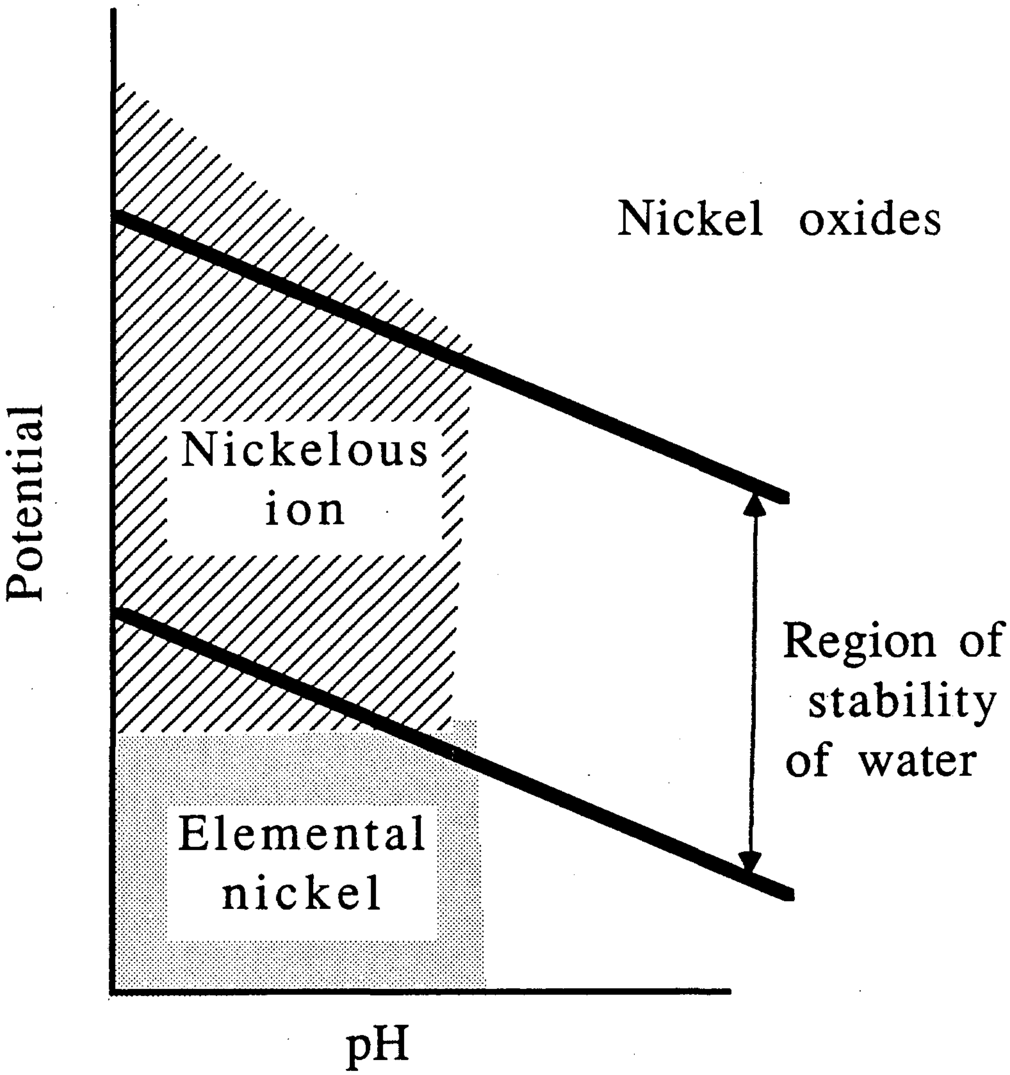


Figure 2-7. Pourbaix diagram showing regions of stability of water, nickel, nickelous ions, and nickel oxides (after Reference 15).

is due to the presence of nickel and not merely to the reduction of adsorbed nitrate, we repeated the experiment with other nitrate compounds: sodium nitrate and nitric acid. First we repeated the experiment with nickel nitrate to show experimental competence, to show the reproducibility of the results, and to control properly the experiment.[16]

2.3.1. Experimental procedure

For the experiments with sodium nitrate, we followed Bernardi's procedure as closely as possible. However, there were changes made. The working electrode was a 0.5-centimeter diameter, glassy carbon rotating disk. The water was purified in the same apparatus to a resistivity of at least 16.0 MΩ·cm. The electrode was polished between runs only with 1 μm diamond paste. Acetone was used for dipping the polished electrode instead of carbon tetrachloride. The digital oscilloscope recorded the results at the rate of 500 ms per point in most cases.

The effect of hydrolysis on the pH of the solution was accounted for. Nickel, unlike sodium, is the cation of a weak base so nickel solutions are more acidic than sodium solutions. Instead of using a solution of 2.56 M nickel nitrate and 0.006 M nitric acid, we used a solution of 5.126 M nitrate and the same pH as the former solution—about 0.75[17]—or 0.178 M nitric acid and 4.948 M sodium nitrate.

2.3.2. Expected results

There are two mutually exclusive possibilities for the role of nickel in the reaction:

(1) nickel plays no role in the reaction and there will be no difference in the results of the two experiments, and

(2) nickel acts as a catalyst and without it there is no reaction or possibly less reaction.

2.4. Results and conclusions

The results of the experiments conducted to reproduce Bernardi's results with solutions of 2.56 M nickel nitrate and 0.006 M nitric acid were themselves reproducible, but the peaks were smaller than those obtained by Bernardi and had no shoulder. One such voltammogram is reproduced here in Figure 2-8. The results are plotted as points, not curves like Bernardi's, because they were recorded by a digital oscilloscope, not a chart recorder.

As described in Section 2-3, we performed experiments without nickel in the solution to find out if we would still observe a peak. Figure 2-9 is the result of an experiment with a 0.2 M nitric acid solution. The sweep rate was 20 mV/s; the rotation speed, 800 rpm. It is obvious that the presence of nickel is not necessary for the appearance of a peak. Figure 2-10 is the result of an experiment with a solution of 4.948 M $NaNO_3$ and 0.178 M HNO_3. Only wishful thinking would convince us that there is a peak near 0.5 V *vs.* SCE. However, after several potential sweeps, a pronounced peak is observed. This is seen in Figure 2-11.

We conclude from these experiments—especially from Figure 2-9—that the presence of nickel is not necessary for the formation of a peak and that the reduction in current (following the peak) is not caused by nickel covering the surface of the electrode.

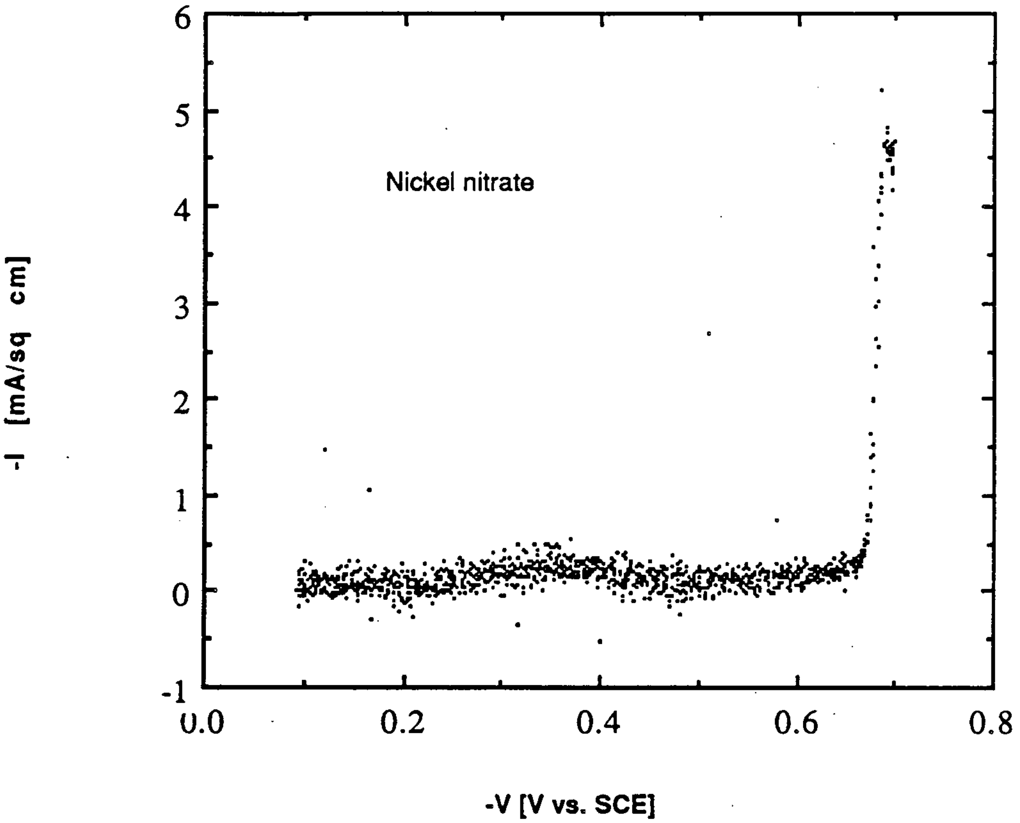


Figure 2-8. Voltammogram produced from a solution of 2.56 M $Ni(NO_3)_2$ and 0.006 M HNO_3 with a glassy carbon rotating-disk working electrode and a SCE reference electrode. The sweep rate was 1 mV/s; the rotation speed, 800 rpm (the same conditions as Figure 2-1).

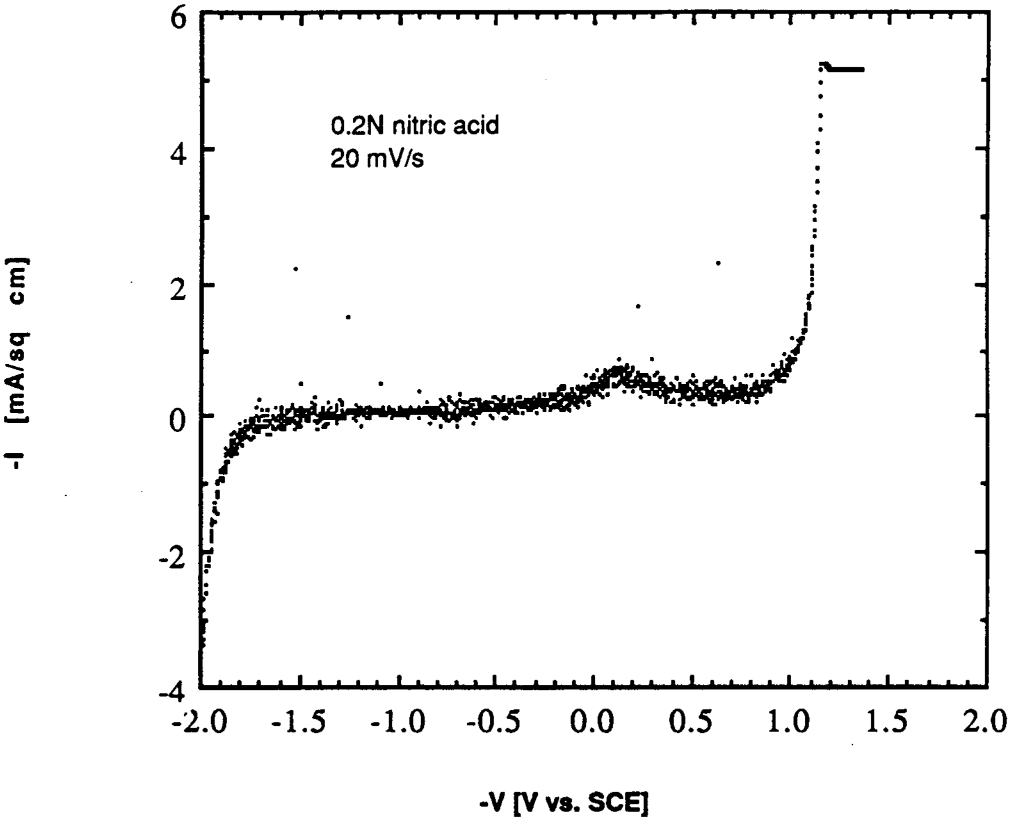


Figure 2-9. Voltammogram produced from a solution of 0.2 M HNO_3 and a SCE reference electrode. The sweep rate was 20 mV/s; the rotation speed, 800 rpm; the working electrode glassy carbon.

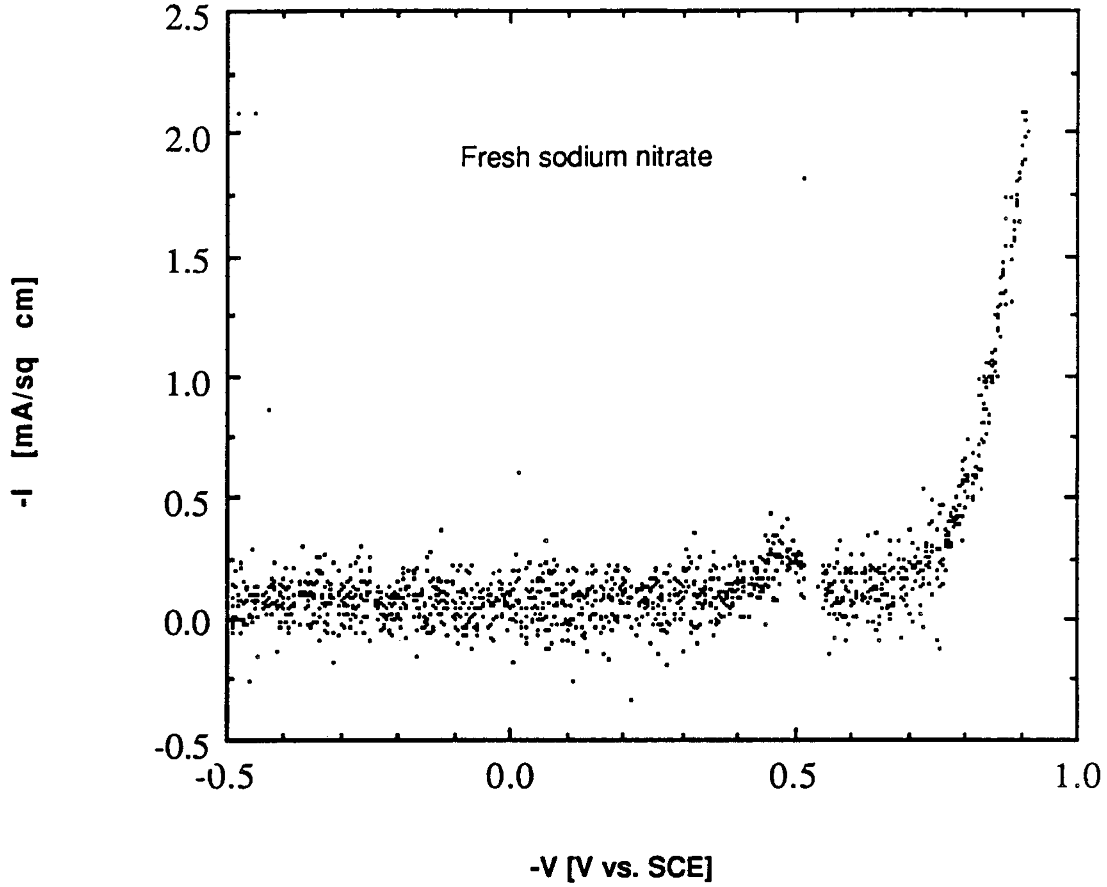


Figure 2-10.. Voltammogram produced from a solution of 4.948 M $NaNO_3$ and 0.178 M HNO_3 and a SCE reference electrode. The sweep rate was 1 mV/s; the rotation speed, 800 rpm; the working electrode glassy carbon.

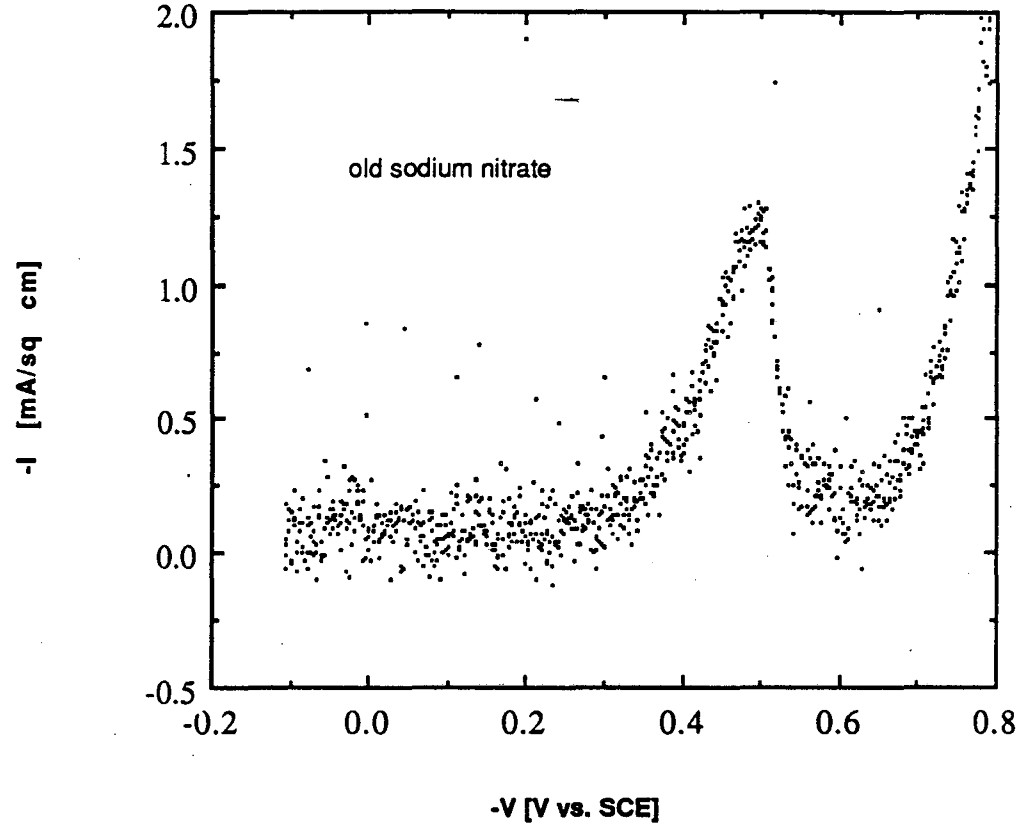


Figure 2-11. Voltammogram produced from the same solution and conditions as Figure 2-10, but after several potential cycles (from -0.1 to 0.8 V).

References

1. Dawn Bernardi, *Mathematical Modeling of Lithium(alloy), Iron Sulfide Cells and The Electrochemical Precipitation of Nickel Hydroxide*, Ph.D. Dissertation, University of California, Berkeley (1986) (LBL-20858).

2. Dawn Bernardi and John Newman, "Modeling the Electrochemical Precipitation of $Ni(OH)_2$ on a Rotating Disk Electrode," in progress.

3. V. V. Zhdanov, K. I. Tikhonov, and A. L. Roitnyan, "Issledovanie katodnogo protsessa na nikelevom elektrode v rastvorakh azotnokislogo nikelya," *Zh. Prikl. Khim.*, **53**, 581 (1980) [English translation: "Cathodic process on a nickel electrode in nickel nitrate solutions," *J. Appl. Chem. USSR*, **53**, 581 (1980).]

4. V. V. Zhdanov, N. A. Chvyrina, K. I. Tikhonov, I. A. Shoshina, and A. L. Roitnyan, "Elektrovosstanovlenie nitrat-iona na nikelevom elektrode," *Zh. Prikl. Khim.*, **53**, 833 (1980) [English translation: "Electroreduction of the nitrate ion at a nickel electrode," *J. Appl. Chem. USSR*, **53**, 833 (1980).]

5. V. V. Zhdanov and K. I. Tikhonov "Issledovanie adsorbtsii nitrita na gladkom nikelevom elektrode," *Zh. Prikl. Khim.*, **53**, 1053 (1980) [English translation: "Adsorption of nitrite on a smooth nickel electrode," *J. Appl. Chem. USSR*, **53**, 1053 (1980).]

6. E. N. Mirolyubov, M. M. Kurtepov, and N. D. Tomashov, "Izuchenie Korrozionnogo i Elektrokhimicheskogo Povedeniya Nerzhaveyushch Stalei Pri Katodnoi Polyarizatsii v Rastvorakh Azotnoi Kisloty: 1. Osobennosti samorastvoreniya stalei," ["Study of Corrosive and Electrochemical Behaviour of Stainless Steels by Cathodic Polarization in Nitric Acid Solutions: 1. Peculiarities Relating to Self-Dissolution of Steels,"] *Izv. Akad. Nauk SSSR, Otd. Khim. Nauk*, no. 6, 1015 (1960).

7. E. N. Mirolyubov, M. M. Kurtepov, and N. D. Tomashov, "Izuchenie Korrozionnogo i Elektrokhimicheskogo Povedeniya Nerzhaveyushch Stalei Pri Katodnoi Polyarizatsii v Rastvorakh Azotnoi Kisloty: 2. Katodnye Protsessy na Nerzhaveyuikh Stalyakh v Rastvorakh Azotnoi Kisloty," ["Study of Corrosive and Electrochemical Behaviour of Stainless Steels by Cathodic Polarization in Nitric Acid Solutions: 2. Cathodic Processes Developing on the Surfaces of Stainless Steels in Nitric Acid Solutions,"] *Izv. Akad. Nauk SSSR, Otd. Khim. Nauk*, no. 7, 1178 (1960).

8. E. N. Mirolyubov, M. M. Kurtepov, and N. D. Tomashov, "Izuchenie Korrozionnogo i Elektrokhimicheskogo Povedeniya Nerzhaveyushch Stalei Pri Katodnoi Polyarizatsii v Rastvorakh Azotnoi Kisloty: 3. Vliyanie Sostava Stalei," ["Study of Corrosive and Electrochemical Behaviour of Stainless Steels by Cathodic Polarization in Nitric Acid Solutions: 3. Effects of Steels' Composition,"] *Izv. Akad. Nauk SSSR, Otd. Khim. Nauk*, no. 7, 1185 (1960).

9. Bernardi, *op. cit.*, p. 205-8.

10. *Ibid.*, Chapter 7.

11. *Ibid.*, pp. 212-221.

12. A. Frumkin, "Adsorptionserscheinungen und elektrochemische Kinetik," ["Adsorption Phenomena and Electrochemical Kinetics,"] *Z. Elektrochem.*, **57**, 807 (1955)

13. A. N. Frumkin and N. V. Nikolajeva, "On the Electroreduction of Anions," *J. Chem. Phys.*, **26**, 1552 (1957).

14. Roger Parsons, "The Structure of the Electrical Double Layer and Its Influence on the Rates of Electrode Reactions," *Adv. Electrochem. Electrochem. Eng.*, 1, 1 (1961).

15. Marcel Pourbaix, *Atlas of Electrochemical Equilibria in Solutions*, p. 333, National Association of Corrosion Engineers, Houston, Texas (1974).

16. Richard P. Feynman, *"Surely You're Joking, Mr. Feynman!"*, p. 314, Bantam Books (1985).

17. D. Bernardi, *op. cit.*, Figure 7-3, p. 209.

CHAPTER 3

Voltammetry Modeling

Voltammograms (graphs of the current produced by a system as a function of the applied potential) can be simulated using a mathematical model of the experiment. Such a model is developed here to understand the experimental results presented in Chapter 2. The model is essentially a material balance of the reacting species at the electrode-solution interface. The model can handle an arbitrary number of species and reactions, the reaction of uncharged species, adsorption and desorption, and rotating and stationary electrodes.

The concentration derivatives in the material balance equations can be evaluated using superposition integrals[1] (Duhamel's principle). These relate a concentration derivative at the electrode surface (which changes in response to the applied potential) to the history of the concentration profile. The arbitrary applied potential is approximated by step functions. The response to each of these steps is weighted by its distance in time and summed. This is done using the method of Acrivos and Chambré[2] that is related to Wagner's method for solving Volterra integral equations.[3] We adopted the algorithm and computer programs of Matlosz and coworkers,[4-7] and we adapted them to the problem at hand. The governing equations, superposition integrals, and Matlosz's programs will be discussed in the next three sections. Modifications necessary to the programs will be discussed in Section 3.4.

3.1. Governing equations

Matlosz's model accounts for electron transfer reactions at an electrode. The hydrodynamic situations that were included were diffusion through a stagnant film (to a rotating disk) and semi-infinite diffusion (to a stationary electrode). The present model accounts for the participation of adsorbed species in heterogeneous reactions which may not involve electron transfer. Charge may also be transferred between the solution and the surface by adsorption or desorption. We account for this by substituting $n_j' = -\sum_i s_{ij}z_i$ for n_j, the number of electrons transferred. In addition to the concentration of each solute and adsorbate, the ohmic drop is treated as an unknown. Its value is a function of the current being passed and could be eliminated by using this function of the other unknowns, but this leads to computational problems.

The equations to be solved are material balances for each species at the electrode surface, that is to say, on the surface (for adsorbates) or in the solution immediately outside the diffuse part of the double layer (for solutes). For solutes, the material balance is:

$$D_i \frac{\partial c_i}{\partial y} = \sum_j \frac{s_{ij} i_j}{n_j' F} \ ; \tag{3-1}$$

for adsorbates:

$$\Gamma_{max} \frac{d\Theta_i}{dt} = \sum_j \frac{s_{ij} i_j}{n_j' F} \tag{3-2}$$

or

$$2\Gamma_{max} \frac{\Theta_i - \Theta_i^{old}}{\Delta t} = \sum_j \frac{s_{ij}\left(i_j + i_j^{old}\right)}{n_j' F} \ . \tag{3-3}$$

The surface concentration of unoccupied sites is treated as a species and is determined not from a material balance, but from the condition:

$$\sum_i \Theta_i = 1 \ . \tag{3-4}$$

The spatial derivatives in Equation 3-1 are calculated using superposition integrals. The equation defining the ohmic drop is

$$V_\Omega = V_{Surf}(t) - \frac{FA}{4\kappa r_o} \sum_i z_i D_i \left[\frac{\partial c_i}{\partial x}\right]_{x=0} \ . \tag{3-5}$$

3.2. Superposition integrals

A linear partial differential equation (*e.g.*, the equation of heat conduction) valid in a one-dimensional domain and subject to given initial conditions and transient boundary conditions can be solved if the solution to the equation is known for a unit-step change in the boundary condition. The solution of the original problem is the superposition (linear combination) of solutions to the step-change problem denoted $\bar{c}(x,t)$. In other words, if the arbitrary, possibly nonlinear, transient boundary condition is expressed as the sum of step changes of different magnitudes made at different times, then the response of the system to this linear combination of step changes on the boundary is the linear combination of the responses to these step changes.

For example, consider Fick's second law:

$$\frac{\partial c}{\partial t} = D\,\frac{\partial^2 c}{\partial x^2}\,. \tag{3-6}$$

If we want to know how the concentration distribution c(x,t) (or how the flux to the electrode) changes in response to changes in concentration at the electrode surface (which may result from an applied time-varying potential), we can divide the transient surface boundary conditions into step changes in concentration at the surface made at different times and add up the responses to these changes. This summation can be represented in the limit by Duhamel's integral:[1]

$$c(x,t) = \int_0^t \frac{\partial c}{\partial t}\bigg|_{0,\tau} \cdot \bar{c}(x,t-\tau)\,d\tau\,. \tag{3-7}$$

The first term in the integrand (the time derivative at the surface) comes from the imposed transient boundary condition; the second term, the solution to the unit-step change problem ($\bar{c}(x,t-\tau)$ is the concentration distribution at time *t* in response to a unit-step change at the boundary at time τ). Wagner[3] and Acrivos and Chambré[2] suggest dividing the integral into the sum of integrals over small increments of τ. They approximate the time derivative of the surface concentration as constant over the small time intervals and rewrite Duhamel's integral for the derivative of the surface concentration as

$$\frac{\partial c}{\partial x}\bigg|_{0,t} = \sum_{k=0}^{n-1} \frac{c_{k+1}-c_k}{\Delta t} \cdot \int_{k\Delta t}^{(k+1)\Delta t} \frac{\partial \bar{c}}{\partial x}\bigg|_{0,t-\tau} d\tau\,. \tag{3-8}$$

Thus the derivative can be expressed in terms of the unknown concentration, the known concentrations at all previous times, and known functions of $\bar{c}$.

At a step n, the values of the surface concentrations for all the previous steps have been calculated, and it is possible to evaluate all these terms before any iteration. This can be seen from:

$$\left.\frac{\partial c}{\partial x}\right|_{0,t} = \sum_{k=0}^{n-2} \frac{c_{k+1}-c_k}{\Delta t} \cdot \int_{k\Delta t}^{(k+1)\Delta t} \left.\frac{\partial \bar{c}}{\partial x}\right|_{(0,n\Delta t-\tau)} d\tau$$
$$+ \frac{c_n-c_{n-1}}{\Delta t} \cdot \int_{(n-1)\Delta t}^{n\Delta t} \left.\frac{\partial \bar{c}}{\partial x}\right|_{(0,n\Delta t-\tau)} d\tau \,. \tag{3-9}$$

The integrals in the above equation depend only on the solution to the unit-step problem and the index n-k, so they can be calculated only once and stored. Matlosz calls these integrals A_{n-k} and writes

$$A_{n-k} = \int_{k\Delta t}^{(k+1)\Delta t} \left.\frac{\partial \bar{c}}{\partial x}\right|_{(0,n\Delta t-\tau)} d\tau \,. \tag{3-10}$$

For convenience the A_{n-k}'s are expressed as the difference of two simpler integrals

$$A_{n-k} = a[(n-k)\Delta t] - a[(n-k-1)\Delta t] \,, \tag{3-11}$$

where

$$a(t) = \int_0^t \left.\frac{\partial \bar{c}}{\partial y}\right|_{0,\tau} d\tau \,. \tag{3-12}$$

3.3. Matlosz's *CycVolt* program

Matlosz wrote a computer program to simulate cyclic voltammetry.[5] He and his coworkers used the program to model the iodide/tri-iodide/iodine system in propylene carbonate solution[4-6] and the deposition of mercury from chloride solution[5,7]

This program *CycVolt* contains the governing equations; the procedure *SuperPose* calculates the concentration derivatives using superposition integrals. Matlosz[5] incorrectly derived the integral a_i for the Nernst-layer case. His Equation 3-21 should be written

$$a_i(t) = -\frac{t}{\delta_i} - \sum_{m=1}^{\infty} \frac{2\delta_i}{D_i m^2 \pi^2} \left[1 - \exp\left[-\frac{m^2\pi^2 D_i}{\delta_i^2} t\right]\right] . \tag{3-13}$$

The corresponding corrections should also be made in his Equations 3-22 and 3-23. Implications of the result of this error are discussed in appendix A.[†]

3.4. Changes in *CycVolt* and *SuperPose*

The programs *CycVolt* and *SuperPose* written by Matlosz were changed to model the nitrate-reduction reaction. To allow adsorbed species to participate in the reactions, a more general reaction mechanism was incorporated. The details of this change are discussed in Section 3.4.1. The solution for convective diffusion to a rotating-disk electrode was added to *SuperPose*. The derivations for this change are discussed in Section 3.4.2. Listings of the modified versions of the programs are found in appendices D through I. (*CV* supercedes *CycVolt*; *Pose2Mod*, *PoseMod*; *SuperPose2*, *SuperPose*.)

Another change to *SuperPose* was to account for the time between the immersion of the electrode in the solution with the potential set at a given value and the time when the potential ramp is begun (t = 0). To do this we added a term to the superposition integral to account for the change in concentrations at a time far in the past (t = -∞). This term represents the solution to the steady-state mass transfer problem:

$$\frac{\partial c}{\partial x} = \left[c(t{=}0) - c_{bulk}\right] \frac{\partial \bar{c}(t{=}-\infty)}{\partial x} \quad . \tag{3-14}$$

This allows us to determine the values of the concentrations and coverages at time zero, the starting point of the experiment.

The Newton-Raphson routine that solves for the unknowns was also changed. The numerical derivatives are calculated by incrementing the unknown by an amount proportional to the value of the unknown. For the unknown ohmic potential drop it is reasonable to increment the potential by a specified amount, *i.e.* one millivolt. The unknowns include surface coverages and the ohmic potential as well as concentrations adjacent to the surface. Since the potential can be negative and the coverages can not exceed unity, changes were made in the convergence routine and in the section of the program where new guesses are kept within the appropriate range. Another change is that values of U^{θ} are calculated from

[†] The appendices are at the end of this chapter.

values of μ^θ. This is done because values of U^θ are not tabulated for reactions involving adsorbed species. The values of μ^θ are not tabulated for adsorbed species either, but using these as parameters reduces the number of unknown fitting parameters, allows us to calculate the U^θ's as needed, and ensures that these values are thermodynamically consistent. (The values of μ^θ or ΔG_f° are tabulated for solute species.[8,9])

3.4.1. Mechanism of the chemical reactions

Matlosz's program *CycVolt* deals only with solute species. We want to allow adsorbed species to participate in the reactions (either chemical or electrochemical) as well. Each chemical species can exist in solution or adsorbed on the surface. The adsorbed species were treated as additional, separate species with their own governing equations.

The rates of all the reactions, whether they involve adsorbed species or not, are given in Butler-Volmer form. The parameters in this equation are i_o, U^θ, α_c, and α_a. We might ordinarily think of adsorption or other reactions without charge transfer as having as parameters forward and backward rate constants. The relation between the two sets of parameters is discussed by Newman.[10]

3.4.2. Transient convective diffusion to a rotating disk

Matlosz's superposition program (*SuperPose*) simulates two hydrodynamic situations: semi-infinite diffusion and diffusion through a stagnant layer. We wanted to use the exact solution for a rotating-disk electrode instead of the latter to obtain more accurate simulations. Matlosz[5] suggested doing this using the analysis by Nişancioǧlu and Newman.[11]

The difference between the two situations enters the problem in the solution to the unit-step response problem (in procedure *SetCoeffs*). If the rotation speed is zero, *SuperPose* calculates coefficients appropriate to the semi-infinite diffusion case, otherwise it calculates those appropriate to the rotating-disk case.[†]

[†] The stagnant diffusion-layer approximation can be used instead of the Nişancioǧlu and Newman solution by specifying a negative rotation speed.

The difference in currents predicted by the stagnant diffusion-layer approximation and the rotating-disk solution may not be large.

Because the series solutions (for transport to the disk) require many terms to converge for short times, it is convenient to use the semi-infinite diffusion solution at times for which the problem behaves as semi-infinite diffusion, *i.e.* before the mass transfer boundary layer has grown to its steady-state thickness. A question arises about Matlosz's choice of when to use the short-time (semi-infinite diffusion) solution and when to use the solution for longer times (fully-developed boundary layer). He gives no justification for his choice of a time: $\frac{4\delta^2}{\pi^2 D}$.

We use the short-time solution for times less than $\frac{\delta^2}{30D}$. This choice is justified in appendix B.

3.4.3. Unit-step problem

For the rotating disk-electrode geometry, in the absence of migration, the unit-step problem to be solved is

$$\frac{\partial \bar{c}_i}{\partial t} + v_y \frac{\partial \bar{c}_i}{\partial y} = D_i \frac{\partial^2 \bar{c}_i}{\partial y^2} \tag{3-15}$$

subject to the initial and boundary conditions

$$\begin{aligned} &\bar{c}_i = 0 \text{ at } t = 0, \\ &\bar{c}_i = 1 \text{ at } y = 0, t > 0, \text{ and} \\ &\bar{c}_i = 0 \text{ as } y \rightarrow \infty . \end{aligned} \tag{3-16}$$

The derivation presented in this section follows that of Nişancioǧlu and Newman.[11] They solved this problem by introducing the dimensionless time

$$\theta = \Omega \left[\frac{D}{v} \right]^{1/3} \left[\frac{a}{3} \right]^{2/3} t \tag{3-17}$$

and distance

$$\zeta = y\left[\frac{av}{3D}\right]^{1/3} \sqrt{\Omega/\nu}\,. \qquad \textbf{(3-18)}$$

If we substitute for the axial velocity the first term of its series expansion in powers of y,

$$v_y = -a\,\Omega\,\sqrt{\Omega/\nu}\,y^2 \qquad (3\text{-}19)$$

and introduce dimensionless quantities, Equation 3-15 becomes

$$\frac{\partial \bar{c}_i}{\partial \theta} = 3\zeta^2 \cdot \frac{\partial \bar{c}_i}{\partial \zeta} + \frac{\partial^2 \bar{c}_i}{\partial \zeta^2}\,. \qquad (3\text{-}20)$$

The above equation is subject to the conditions

$$\begin{aligned} &\bar{c}_i = 0 \text{ at } \theta = 0, \\ &\bar{c}_i = 1 \text{ at } \zeta = 0, \theta > 0\,, \text{ and} \\ &\bar{c}_i = 0 \text{ as } \zeta \to \infty\,. \end{aligned} \qquad (3\text{-}21)$$

They then express $\bar{c}_i$ in terms of steady-state and transient parts:

$$\bar{c}_i = \bar{c}_i^{ss} - \bar{c}_i^{t}\,, \qquad (3\text{-}22)$$

each of which satisfies Equation 3-20 separately. The boundary conditions for $\bar{c}_i^{ss}$ are

$$\begin{aligned} &\bar{c}_i^{ss} = 0 \text{ as } \zeta \to \infty \text{ and} \\ &\bar{c}_i^{ss} = 1 \text{ at } \zeta = 0. \end{aligned} \qquad (3\text{-}23)$$

This yields the solution

$$\bar{c}_i^{ss} = \frac{1}{\Gamma\left(\frac{4}{3}\right)} \cdot \int_{\zeta}^{\infty} e^{-y^3} dy\,\,. \qquad (3\text{-}24)$$

This integral is tabulated as a function of ζ.[12] Abramowitz[13] gives formulas for the calculation of this integral. If we define

$$I(y) = \int_0^y e^{-u^3} du\,, \qquad (3\text{-}25)$$

we can differentiate with respect to y to obtain

$$I'(y) = e^{-y^3}\,, \qquad (3\text{-}26)$$

which for y = 0 gives

$$I'(0) = 1\,. \qquad (3\text{-}27)$$

The concentration profile's transient part $\bar{c}_i^{t}$ satisfies

$$\begin{aligned}&\bar{c}_i^t = 0 \text{ as } \zeta \to \infty\,,\\&\bar{c}_i^t = \bar{c}_i^{ss} \text{ at } \theta = 0\,, \text{ and}\\&\bar{c}_i^t = 0 \text{ at } \zeta = 0,\ \theta > 0\,.\end{aligned} \tag{3-28}$$

The solution of $\bar{c}_i^t$ is derived in terms of a boundary-value problem. First $\bar{c}_i^t$ is expressed in the form

$$\bar{c}_i^t = \sum_{n=0}^{\infty} B_n Z_n(\zeta) e^{-\lambda_n \theta} \tag{3-29}$$

where Z_n and λ_n are an eigenfunction and its eigenvalue. Z_n is the solution to

$$Z_n'' + 3\zeta^2 Z_n' + \lambda_n Z_n = 0 \tag{3-30}$$

subject to

$$Z_n(\infty) = 0,\ Z_n(0) = 0,\ Z_n'(0) = 1\,. \tag{3-31a,b,c}$$

The values of λ_n and B_n are tabulated in Reference 11. The concentration profile is given by substituting its steady and transient parts, Equations 3-24 and 3-29, into Equation 3-22 to obtain

$$\bar{c}_i = \frac{1}{\Gamma\left[\frac{4}{3}\right]} \cdot \int_{\zeta}^{\infty} e^{-y^3} dy - \sum_{n=0}^{\infty} B_n Z_n(\zeta) e^{-\lambda_n \theta}\,. \tag{3-32}$$

3.4.4. Coefficients required by the algorithm

The computer program needs coefficients a_i which are calculated from the derivative of the concentration at the surface:

$$a_i(t) = \int_0^t \left.\frac{\partial \bar{c}_i}{\partial y}\right|_{0,\tau} d\tau\,. \tag{3-12}$$

The surface derivative of the concentration profile is found from

$$\left.\frac{\partial \bar{c}_i}{\partial y}\right|_{0,\theta} = \frac{\partial \bar{c}_i}{\partial \zeta}\frac{d\zeta}{dy}\,. \tag{3-33}$$

The zeta derivative is found by differentiating the definition of zeta (Equation 3-18)

$$\frac{d\zeta}{dy} = \left[\frac{a\nu}{3D_i}\right]^{1/3} \sqrt{\Omega/\nu}\,. \tag{3-34}$$

The concentration derivative is found by differentiating Equation 3-22

$$\frac{\partial \bar{c}_i}{\partial \zeta} = \frac{\partial \bar{c}_i^{ss}}{\partial \zeta} - \frac{\partial \bar{c}_i^{t}}{\partial \zeta} . \tag{3-35}$$

The first term on the right side comes from differentiating Equation 3-24 and substituting the value of $I'(0)$ from Equation 3-27

$$\left.\frac{\partial \bar{c}_i^{ss}}{\partial \zeta}\right|_{0,t} = \frac{\partial}{\partial \zeta}\left[1 - \frac{I(\zeta)}{\Gamma\left[\frac{4}{3}\right]}\right]_{0,t} = -\frac{1}{\Gamma\left[\frac{4}{3}\right]} . \tag{3-36}$$

The second term on the right side of Equation 3-35 is found by differentiating Equation 3-29

$$\frac{\partial \bar{c}_i^{t}}{\partial \zeta} = \frac{\partial}{\partial \zeta}\sum_{n=0}^{\infty} B_n Z_n(\zeta) e^{-\lambda_n \theta} . \tag{3-37}$$

Performing the differentiation and using the boundary condition $Z_n'(0) = 1$ from Equation 3-31c, we find

$$\left.\frac{\partial \bar{c}_i^{t}}{\partial \zeta}\right|_{0,\theta} = \sum_{n=0}^{\infty} B_n e^{-\lambda_n \theta} , \tag{3-38}$$

into which we can substitute for θ its definition from Equation 3-17. This can be combined with Equations 3-35 and 3-36 to yield

$$\left.\frac{\partial \bar{c}_i}{\partial \zeta}\right|_{0,\theta} = -\frac{1}{\Gamma\left[\frac{4}{3}\right]} - \sum_{n=0}^{\infty} B_n e^{-\lambda_n \theta} . \tag{3-39}$$

We multiply this by $\frac{\partial \zeta}{\partial y}$ from Equation 3-34 and integrate to find a_i as in Equation 3-32

$$a_i(t) = \left[\frac{av}{3D_i}\right]^{1/3}\left[\frac{\Omega}{v}\right]^{1/2}$$

$$\times\left[-\frac{t}{\Gamma\left[\frac{4}{3}\right]} - \sum_{n=0}^{\infty}\frac{B_n\left[1-\exp\left[-\lambda_n\Omega\left[\frac{D_i}{v}\right]^{1/3}\left[\frac{a}{3}\right]^{2/3} t\right]\right]}{\lambda_n\Omega\left[\frac{D_i}{v}\right]^{1/3}\left[\frac{a}{3}\right]^{2/3}}\right] . \tag{3-40}$$

3.4.5. Effect on the limiting current of including the velocity profile

Matlosz simulated the linear sweep voltammetry of mercury deposition from chloride solution.[14] The stagnant diffusion-layer approximation predicts different limiting currents than the experiments produced. To see if this is due to the stagnant-diffusion-layer approximation, we ran the model for this case using the same data used by Matlosz for convective diffusion and for the stagnant diffusion layer approximation. The results of the calculations for several rotation speeds are displayed in Figure 3-1. There is not much difference between the curves in this case. At the first time step (10mV, 18.9s) and 400 rpm the current calculated using the stagnant diffusion layer approximation is 0.035% higher than that calculated using the convective-diffusion equation. The difference decreases with increasing rotation speed.

We expect the difference to be greater for high sweep rates and kinematic viscosities, low rotation speeds and diffusivities, and short times. We would like to quantify this difference so we compare the first ten terms of the summations which represent the concentration derivatives at the surface for the Nernst-layer and convective-diffusion cases for a concentration of zero at the surface and unity in the bulk. These derivatives are

$$\left.\frac{\partial \bar{c}_N}{\partial \zeta}\right|_{\zeta=0} = -\frac{1}{\Gamma\left(\frac{4}{3}\right)} - \sum_{m=1}^{10} \frac{2}{\Gamma\left(\frac{4}{3}\right)} e^{-m^2\pi^2\theta/\left[\Gamma\left(\frac{4}{3}\right)\right]^2}, \tag{3-41}$$

for the Nernst-layer approximation and

$$\left.\frac{\partial \bar{c}_{CD}}{\partial \zeta}\right|_{\zeta=0} = -\frac{1}{\Gamma\left(\frac{4}{3}\right)} - \sum_{n=0}^{9} B_n e^{-\lambda_n\theta}, \tag{3-42}$$

for the convective-diffusion equation. To compare the relative differences between these summations, we divide their absolute difference by their long-time asymptote and plot this value, $\Gamma\left(\frac{4}{3}\right)\left|\left.\frac{\partial \bar{c}_N}{\partial \zeta}\right|_{\zeta=0} - \left.\frac{\partial \bar{c}_{CD}}{\partial \zeta}\right|_{\zeta=0}\right|$, *versus* dimensionless time, θ, in Figure 3-2. The difference is large at short times and decreases exponentially with increasing time.

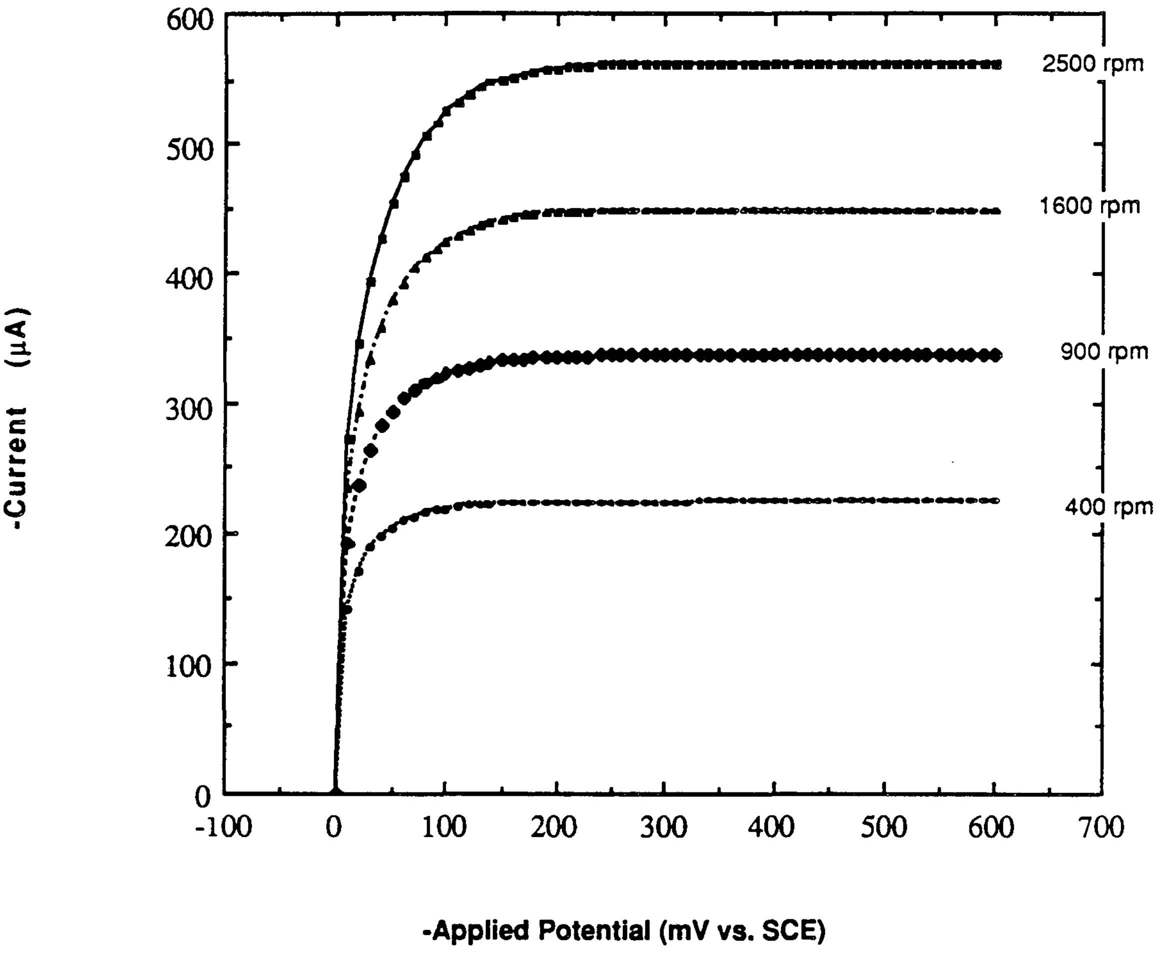


Figure 3-1. Simulated voltammograms for the deposition of mercury from brine on a rotating-disk electrode for several rotation speeds. The symbols indicate the convective-diffusion result; the lines, the stagnant-diffusion-layer result.

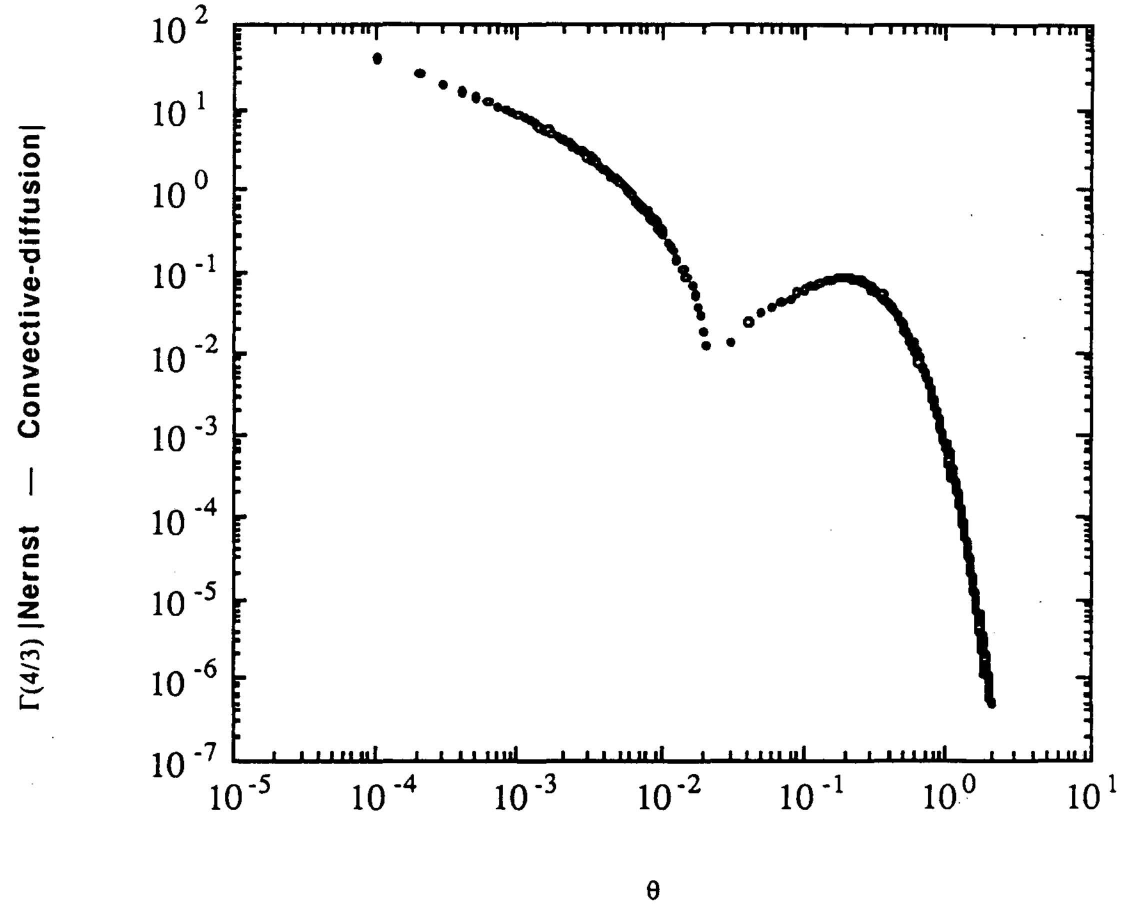


Figure 3-2. Relative difference between currents calculated from the Nernst-layer approximation and the convective diffusion equation.

3.5. First-order homogeneous reactions

For completeness we include the solution to the transient convective diffusion equation with first order homogeneous reaction kinetics. If there are homogeneous (bulk) chemical reactions that have first order or pseudo-first order kinetics, the problem is linear, and the superposition method remains applicable. If the rate of disappearance of species i is kc_i, the governing equation becomes

$$\frac{\partial c_i}{\partial \theta} = \frac{\partial^2 c_i}{\partial \zeta^2} + 3\zeta^2 \cdot \frac{\partial c_i}{\partial \zeta} - \kappa c_i \, , \tag{3-43}$$

subject to

$$\begin{aligned} &c_i = 0 \text{ as } \zeta \to \infty \, , \\ &c_i = 0 \text{ at } t = 0 \, , \text{ and} \\ &c_i = f(t) \text{ at } \zeta = 0 \, . \end{aligned} \tag{3-44}$$

The definition of κ is

$$\kappa = \frac{k}{\Omega \left[\frac{D_i}{\nu} \right]^{1/3} \left[\frac{a}{3} \right]^{2/3}} \, . \tag{3-45}$$

Assume a solution of the form[†]

$$c = e^{-\kappa\theta} g(\zeta, \theta) \, . \tag{3-46}$$

This allows us to reduce the problem to that of convective diffusion to a rotating-disk electrode:

$$\frac{\partial g}{\partial \theta} = \frac{\partial^2 g}{\partial \zeta^2} + 3\zeta^2 \cdot \frac{\partial g}{\partial \zeta} \, . \tag{3-47}$$

The boundary conditions are:

$$\begin{aligned} &g = 0 \text{ as } \zeta \to \infty \, , \\ &g = 0 \text{ at } \theta = 0 \, , \text{ and} \\ &g = f(t) e^{\kappa\theta} \text{ at } \zeta = 0 \, . \end{aligned} \tag{3-48}$$

The solution for g is the same as that used for the rotating-disk. Instead of using the boundary condition on c, $c(0,\theta) = f(\theta)$, one uses the boundary condition $g(0,\theta) = f(\theta)e^{\kappa\theta}$. If the *SuperPose* algorithm is used, the "currents" calculated are based on the value of $\left. \frac{\partial g}{\partial \zeta} \right|_{\zeta=0}$. To determine the true current, these values

[†] Mr. Alan West suggested this solution.

must be multiplied by $e^{-\kappa\theta}$.

3.6. Nonlinear problems

As stated earlier, superposition integrals can only be used to solve linear equations. If we want to include non-first-order bulk chemical reactions or migration in a model, a different technique would be used. For this type of problem, a finite-difference method could be used with time-stepping. Matlosz gave an example of how his *BandAid* program[15] could be used with time stepping to simulate linear sweep voltammetry on a rotating-disk electrode including the migration flux.[5]

References

1. Francis B. Hildebrand, *Advanced Calculus for Applications*, 2nd ed., Prentice-Hall, Englewood Cliffs, NJ (1976).

2. Andreas Acrivos and Paul L. Chambré, "Laminar Boundary Layer Flows with Surface Reactions," *Ind. Eng. Chem.*, 49, 1025 (1957).

3. Carl Wagner, "On the Numerical Solution of Volterra Integral Equations," *J. Math. Phys.*, 32, 289 (1954).

4. Karrie Hanson, Michael Matlosz, John Newman, and Charles W. Tobias, "Theory of Cyclic Voltammetry for Reactions Having Complex Stoichiometry," *Extended Abstracts*, Abstract III2-12, p. 950, International Society of Electrochemistry, 35th Meeting—Berkeley, CA, August, 1984.

5. Michael John Matlosz, *Experimental Methods and Software Tools for the Analysis of Electrochemical Systems*, Ph.D. Dissertation, University of California, Berkeley (1985) (LBL-19375).

6. Karrie J. Hanson, Michael J. Matlosz, Charles W. Tobias, and John Newman, "Electrochemistry of Iodide in Propylene Carbonate: II. Theoretical Model," *J. Electrochem. Soc.*, 134, 2210 (1987).

7. Michael Matlosz and John Newman, for the Removal of Mercury from Contaminated Brine," *J. Electrochem. Soc.*, 133, 1850 (1986).

8. Donald D. Wagman, William H. Evans, Vivian B. Parker, Richard H. Schumm, Iva Halow, Sylvia M. Bailey, Kenneth L. Churney, and Ralph L. Nuttall, "The NBS tables of chemical thermodynamic properties: Selected values for inorganic and C_1 and C_2 organic substances in SI units," *J. Phys. Chem. Ref. Data*, 11, Supplement 2 (1982).

9. Allen J. Bard, Roger Parsons, and Joseph Jordan, *Standard Potentials in Aqueous Solution*, Marcel Dekker, New York (1985).

10. John Newman, *Electrochemical Systems*, Chapter 8, Prentice-Hall, Inc., Englewood Cliffs, NJ (1973).

11. Kemal Nişancioğlu and John Newman, "Transient Convective Diffusion to a Disk Electrode," *J. Electroanal. Chem. Interfacial Electrochem.*, 50, 23 (1974).

12. Milton Abramowitz and Irene A. Stegun, eds., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, p. 320, Dover, New York (1964).

13. Milton Abramowitz, "Table of the Integral, $\int_0^x e^{-u^2}\,du$," *J. Math. Phys.*, 30, 162 (1951).

14. Matlosz, *op. cit.*, p. 168.

15. M. Matlosz and J. Newman, "Solving 1-D Boundary-Value Problems with *BandAid*: A Functional Programming Style and a Complementary Software Tool," *Comput. Chem. Eng.*, 11, 45 (1987).

APPENDIX A

The Incorrect Transient Term in $a_i(t)$

For use in an algorithm for the simulation of cyclic voltammetry using superposition integrals, Matlosz derived an equation for the integral $a_i(t)$. For the Nernst stagnant diffusion-layer case, Equation 3-21 in Reference 5 (in Chapter 3) is

$$a_i(t) = -\frac{t}{\delta_i} - \sum_{m=1}^{\infty} \frac{2\delta_i}{D_i m^2 \pi^2} \cdot \exp\left[-\frac{m^2\pi^2 D_i}{\delta_i^2} t\right] . \tag{A-1}$$

The correct equation is Equation 2-13

$$a_i(t) = -\frac{t}{\delta_i} - \sum_{m=1}^{\infty} \frac{2\delta_i}{D_i m^2 \pi^2} \cdot \left[1 - \exp\left[-\frac{m^2\pi^2 D_i}{\delta_i^2} t\right]\right] . \tag{2-13}$$

Since we are concerned with difference between values of a_i at different times, the constant term in Equation 2-13

$$\frac{2\delta_i}{D_i\pi^2} \sum_{m=1}^{\infty} \frac{1}{m^2} \tag{A-2}$$

may not bother us.† If we do not include this term, we are left with an expression similar to Matlosz's, except that we will add the summation term to $-t/\delta_i$ instead of subtracting it. At long times, the transient (summation) term is negligible, so its sign is irrelevant. At short times, this term is more important, but a different expression (for semi-infinite diffusion) is used instead. The transient term influences the results of the simulation only at intermediate times.

† The summation of m^{-2} from one to infinity converges to 1.6449340668482264363 7.[1]

Reference

1. William H. Beyer (Editor), *CRC Standard Math Tables*, 25th ed., CRC Press, Boca Raton, FL (1974).

APPENDIX B

Short-Time Solution to the Convective-Diffusion Equation

B.1. Problem

To determine the coefficients used when solving the convective-diffusion equation using superposition integrals, we need to know the value of certain coefficients. These coefficients are

$$a(t) = \int_0^t \left.\frac{\partial c}{\partial x}\right|_{x=0} dt' . \qquad \text{(B-1)}$$

B.2. Solution

In dimensionless terms, we can calculate

$$a'(\tau) = -\frac{a(t)D}{\delta} \qquad \text{(B-2)}$$

where the dimensionless time is

$$\tau = \frac{Dt}{\delta^2} . \qquad \text{(B-3)}$$

For the case of diffusion through a stagnant film

$$a' = \tau + \frac{2}{\pi^2}\sum_{m=1}^{\infty}\frac{1}{m^2} - \frac{2}{\pi^2}\sum_{m=1}^{\infty}\frac{e^{-m^2\pi^2\tau}}{m^2} . \qquad \text{(B-4)}$$

For short times, many terms are needed to make the series converge, so we would like to use a separate solution for short times. Once we have the solution, we have to decide at what value of τ to switch between the two solutions. Matlosz[1] used $\tau = 4/\pi^2$, but gave no reason for this choice.

To decide the range of τ for which we will use the short-time solution, we will consider the problem of diffusion through a stagnant layer, briefly postponing consideration of the problem of convective-diffusion to a rotating disk. We will derive the short-time solution and compare the series solutions for different numbers of terms to decide when to switch between the solutions in a region where they overlap.

B.3. Short-time solution

For a similar problem, Newman[2] derives the concentration and its gradient in a stagnant diffusion layer using Laplace transforms. For the present boundary conditions, these become

$$\bar{c}(x,s) = -\frac{1}{s}\cdot\frac{\sinh\left[\sqrt{\frac{s}{D}}(x-\delta)\right]}{\sinh\left[\sqrt{\frac{s}{D}}\delta\right]} \tag{B-5}$$

and

$$\left.\frac{\partial\bar{c}}{\partial x}\right|_{x=0} = -\frac{1}{\sqrt{sD}}\cdot\frac{\cosh\left[\sqrt{\frac{s}{D}}\delta\right]}{\sinh\left[\sqrt{\frac{s}{D}}\delta\right]} \tag{B-6}$$

where $\bar{c}(x,s)$ is the Laplace transform of c(x,t). At short times (or large values of s),

$$\begin{aligned}\left.\frac{\partial\bar{c}}{\partial x}\right|_{x=0} &= -\frac{1}{\sqrt{sD}}\cdot\frac{1+\exp\left[-2\sqrt{\frac{s}{D}}\delta\right]}{1-\exp\left[-2\sqrt{\frac{s}{D}}\delta\right]} \\ &= -\frac{1}{\sqrt{sD}}\left[1+2e^{-2\sqrt{\frac{s}{D}}\delta}+2e^{-4\sqrt{\frac{s}{D}}\delta}+2e^{-6\sqrt{\frac{s}{D}}\delta}+\cdots\right]\end{aligned} \tag{B-7}$$

which is inverted to give

$$\left.\frac{\partial c}{\partial x}\right|_{x=0} = -\frac{1}{\sqrt{\pi Dt}}\left[1+2e^{-\delta^2/Dt}+2e^{-4\delta^2/Dt}+2e^{-9\delta^2/Dt}+\cdots\right]. \tag{B-8}$$

To find the coefficients a′(t), we integrate this equation. It should be integrable by parts, but it is simpler to integrate its Laplace transform by dividing by s:

$$\frac{1}{s}\left.\frac{\partial \bar{c}}{\partial x}\right|_{x=0} = \frac{1}{s^{3/2}\sqrt{D}} \cdot \left[1 + 2\sum_{n=1}^{\infty} e^{-2n\sqrt{\frac{s}{D}}\delta}\right]. \qquad \textbf{(B-9)}$$

This can be inverted to give, in dimensionless terms

$$a'(t) = -\sqrt{\frac{4\tau}{\pi}}\left\{1 + 2\sum_{n=1}^{\infty}\left[e^{-n^2/\tau} - n\sqrt{\frac{\pi}{\tau}}\,\mathrm{erfc}\left[\frac{n}{\sqrt{\tau}}\right]\right]\right\}. \qquad \text{(B-10)}$$

The value of $a'(\tau)$ for semi-infinite stagnant diffusion is $\sqrt{4\frac{\tau}{\pi}}$. It is the limiting value for diffusion through a finite film for the shortest times. The terms in the series in Equation B-10 represent corrections to this limiting, "zero-term," case. The method used for calculating the error function complement is discussed in appendix C.

The values of $a'(\tau)$ for the zero-, one-, two-, and three-term short-time series and the ten- and hundred-term series (which overlap for long times) are plotted in Figure B-1. The zero-term solution, proportional to $\sqrt{\tau}$ is a straight line. As more terms are added to this solution, the value of a′ follows the long-time solution until larger times before diverging. The long-time solution intersects the ordinate axis at positive values which decrease as the number of terms used increases.

The relative errors for the different series are plotted in Figure B-2. The error is

$$\left|\frac{a'}{\text{true } a'} - 1\right|.$$

The true value of a′ is taken to be the hundred-term long-time solution for $\tau > 0.1$ and the three-term short-time solution for $\tau < 0.1$; from this figure we decide to use the ten-term long-time solution for $\log \tau > -1.5$ (where the relative error is less than $6 \cdot 10^{-6}\%$) and the zero-term short-time solution for $\log \tau < -1.5$ (where the relative error is less than $4 \cdot 10^{-5}\%$).

B.4. Complete convective-diffusion problem

The coefficient a(t) for convective diffusion to a rotating-disk was derived in Section 3.2.4. In dimensionless terms, we have

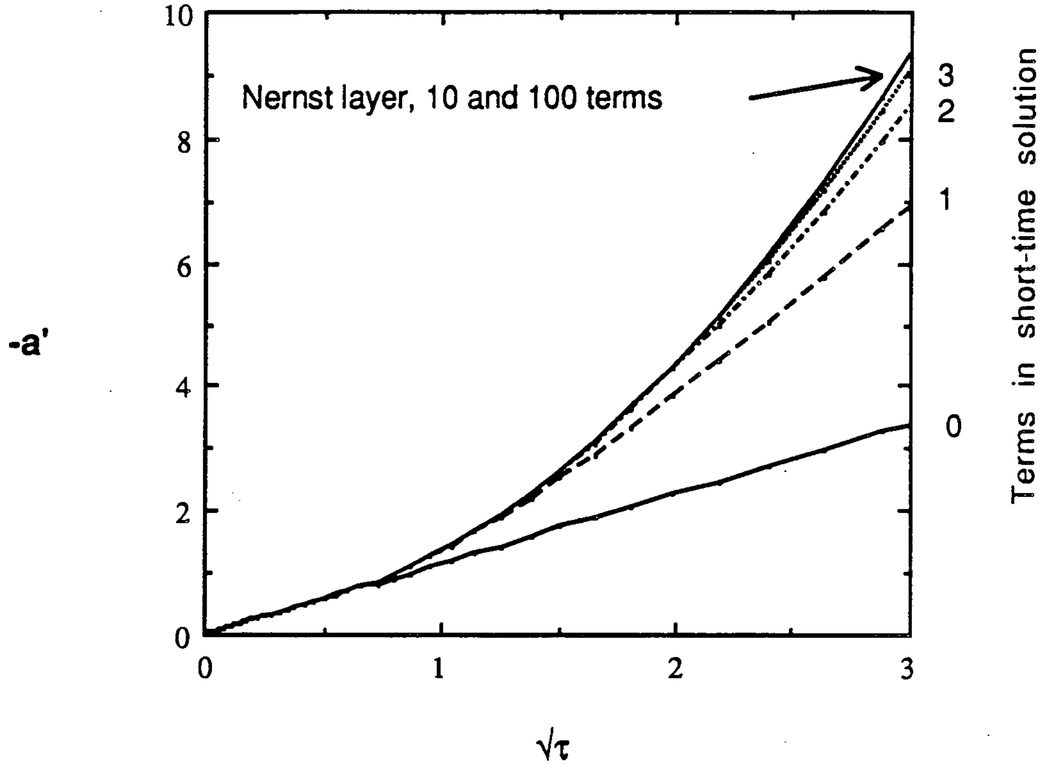


Figure B-1. Coefficients for the superposition integral program. Curves 0, 1, 2, 3: short-time solution for a stagnant diffusion layer with 1, 2, and 3 correction terms added to the semi-infinite solution. Curve L: stagnant diffusion layer solution with 10 and 100 terms (the two curves overlap).

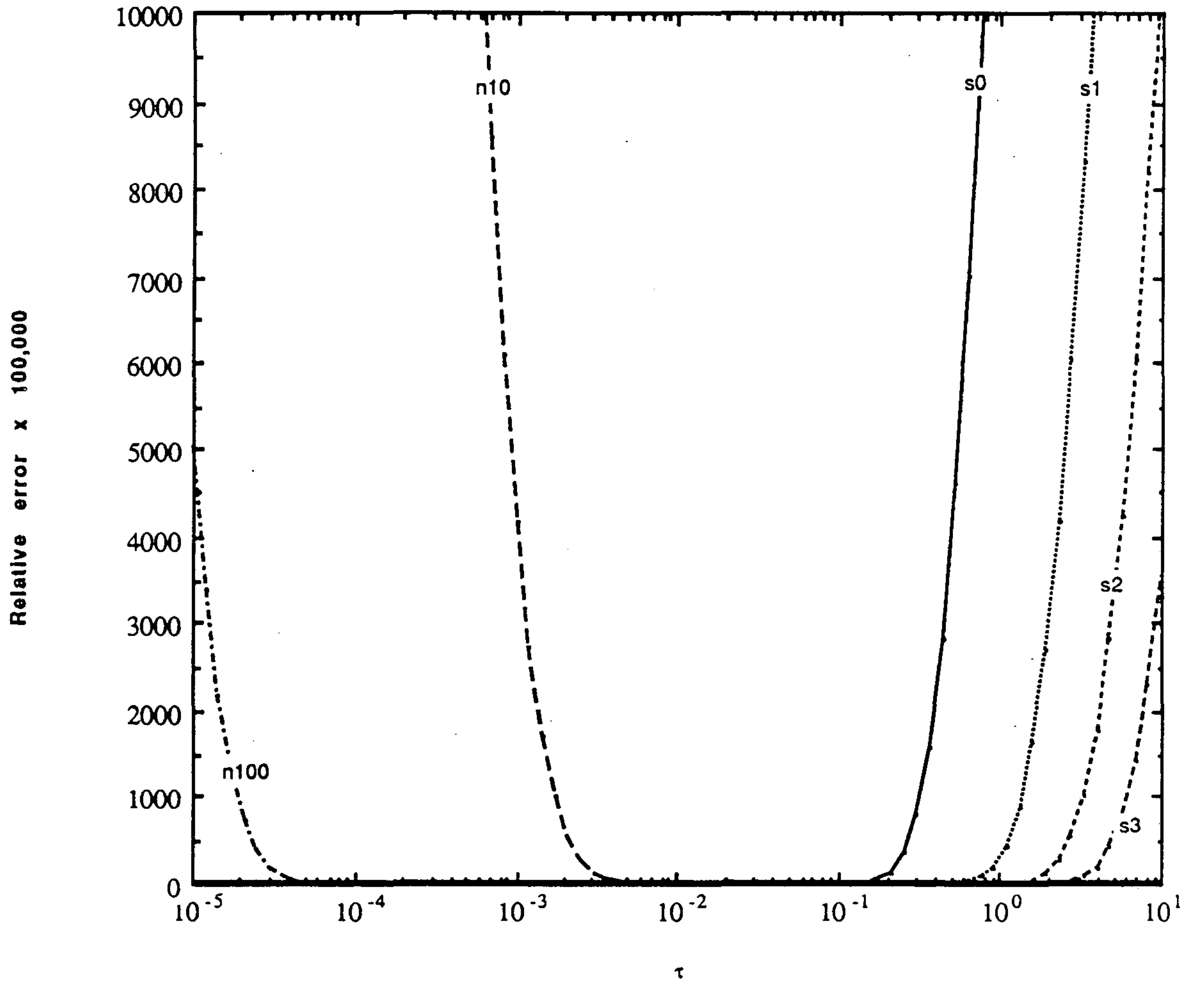


Figure B-2. Coefficients for superposition integral. "n" and "s" refer to the Nernst-layer approximation and to the short-time series, respectively. The numbers refer to the number of terms in the series.

$$a'(\tau) = \tau + \sum_{m=0}^{\infty} \frac{B_m \left(1 - e^{-\lambda_m \left[\Gamma\left(\frac{4}{3}\right)\right]^3 \tau}\right)}{\Gamma\left(\frac{4}{3}\right)\lambda_m}, \tag{B-11}$$

which is similar to the result for diffusion through a stagnant film (Equation B-4). The hundred-term stagnant-diffusion a′ and the ten-term convective-diffusion a′ are plotted in Figure B-3. The curves match well, especially at larger values of τ. This can be seen also in a plot of the relative difference between the two terms in Figure B-4. This difference is expressed as $\left|1 - \frac{a'_{\text{convective}} \text{ diffusion}}{a'_{\text{Nernst}}}\right|$ and decreases exponentially at large times.

We would expect our answer to be wrong by a constant: the summation $\sum \frac{1}{m^2}$ does not converge for few terms (see Table B-1). Since we know the value of $\sum_{m=1}^{\infty} \frac{1}{m^2}$, we used it in the stagnant-diffusion problem for the sum of the reciprocals of the eigenvalues rather than ten or one hundred terms as in the other series term. In the convective-diffusion problem we cannot do this: Nişancioǧlu and Newman give only the first ten eigenvalues and constants. To find the difference between $\sum_{m=0}^{\infty} \frac{B_m}{\Gamma\left(\frac{4}{3}\right)\lambda_m}$ and $\sum_{m=0}^{9} \frac{B_m}{\Gamma\left(\frac{4}{3}\right)\lambda_m}$, we looked at the value of a′ for the ten-term convective-diffusion series and the short-time

Table B-1. The summation of m^{-2} for various numbers of terms.

N	$\sum_{m=1}^{N} \frac{1}{m^2}$
10	1.54977
100	1.63498
200	1.63877
∞	1.64493

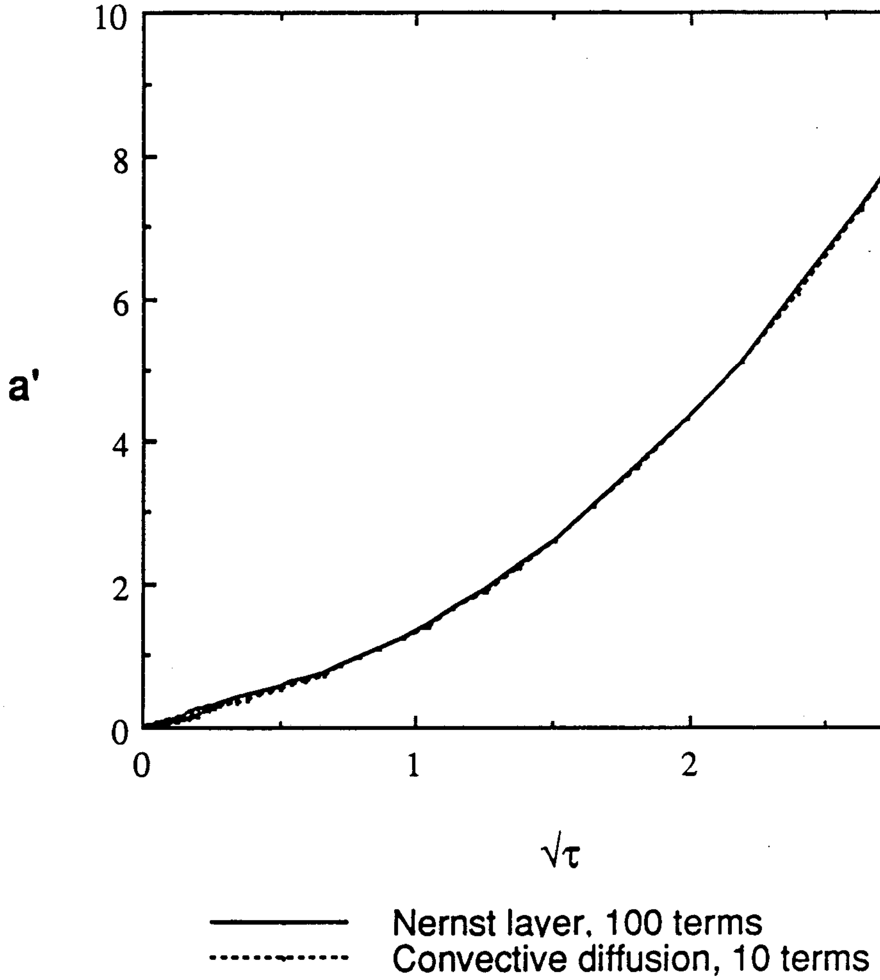


Figure B-3. Coefficients for superposition integral. Dashed curve: 100-term Nernst diffusion layer solution. Solid curve: 10-term convective diffusion solution.

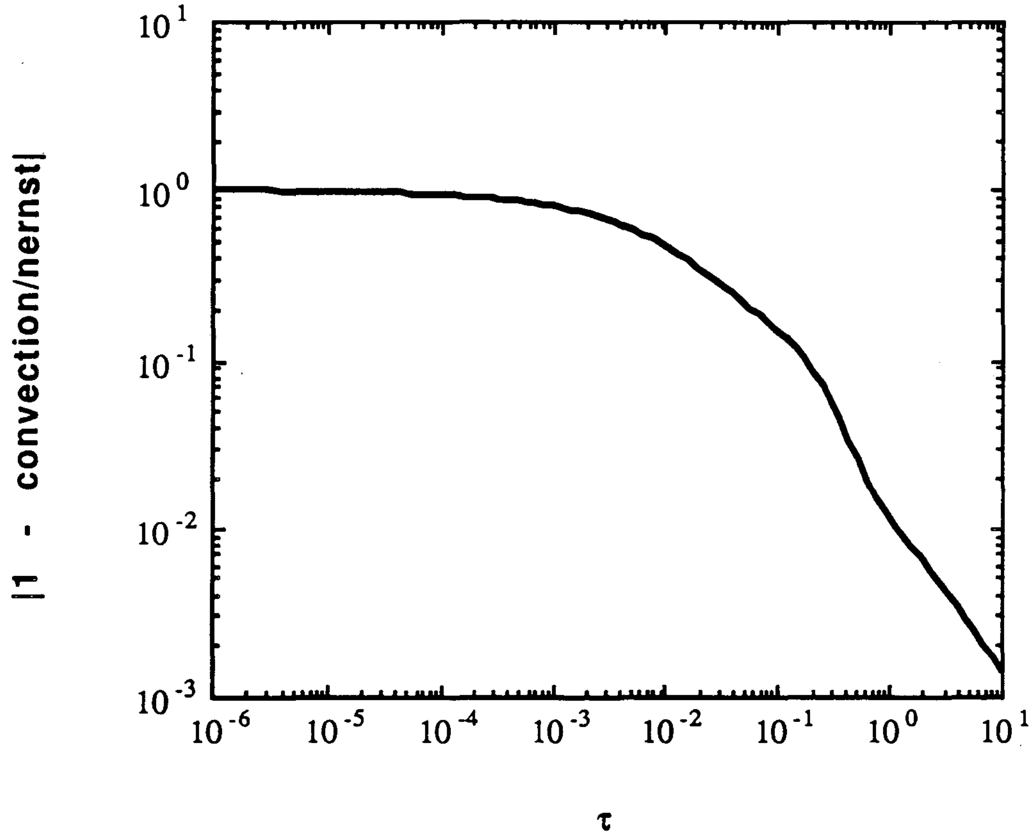


Figure B-4. Coefficients for the superposition integral.

solution where they should overlap ($\log \tau \sim -1.5$). The difference between the two is about 0.055. Therefore we should subtract this value from the ten-term convective-diffusion a′. Figure B-5 contains the same data as Figure B-4 but with the correction term 0.055 included. Since the superposition program only uses differences in a′, the value of the constant need not be precisely known for this purpose.

B.5. Selman's work

In appendix B of his dissertation,[3] Selman considers the flux at a rotating disk after a concentration step at the surface in the limit of high Schmidt numbers.

Selman writes the equation of convective diffusion and notes that the convective term is negligible for small times and derives the concentration gradient at the surface for small times. He then recognizes the long-time (steady-state) solution to the convective-diffusion equation and derives the surface concentration gradient.

Then, he rewrites the convective-diffusion equation in terms involving its short-time solution. He writes a perturbation expansion about the short-time solution and derives the concentration gradient at the surface. He solves the convective-diffusion equation numerically using two sets of variables: one for short times and one for long times, and explores the relationship between the flux and time for different numbers of terms.

His Equation B.22 gives his short-time series for the flux at the surface

$$\left.\frac{\partial c}{\partial \eta}\right|_o = \frac{1}{\sqrt{\pi\tau}} + \frac{3\tau}{4} + \frac{3\tau^{5/2}}{20\sqrt{\pi}} + O(\tau^4)\,, \tag{B-12}$$

where his notation differs slightly from ours. While the form of this equation differs from our Equation B-8, he shows that the exact solution differs from the steady-state solution at nearly the same time that we found.

In his Figure B.1,[4] Selman plots $(\partial c/\partial y)_o$ against τ_S. The exact solution diverges from the steady-state solution ($1/\sqrt{\pi\tau_S}$) at $\tau_S \approx 0.035$, where

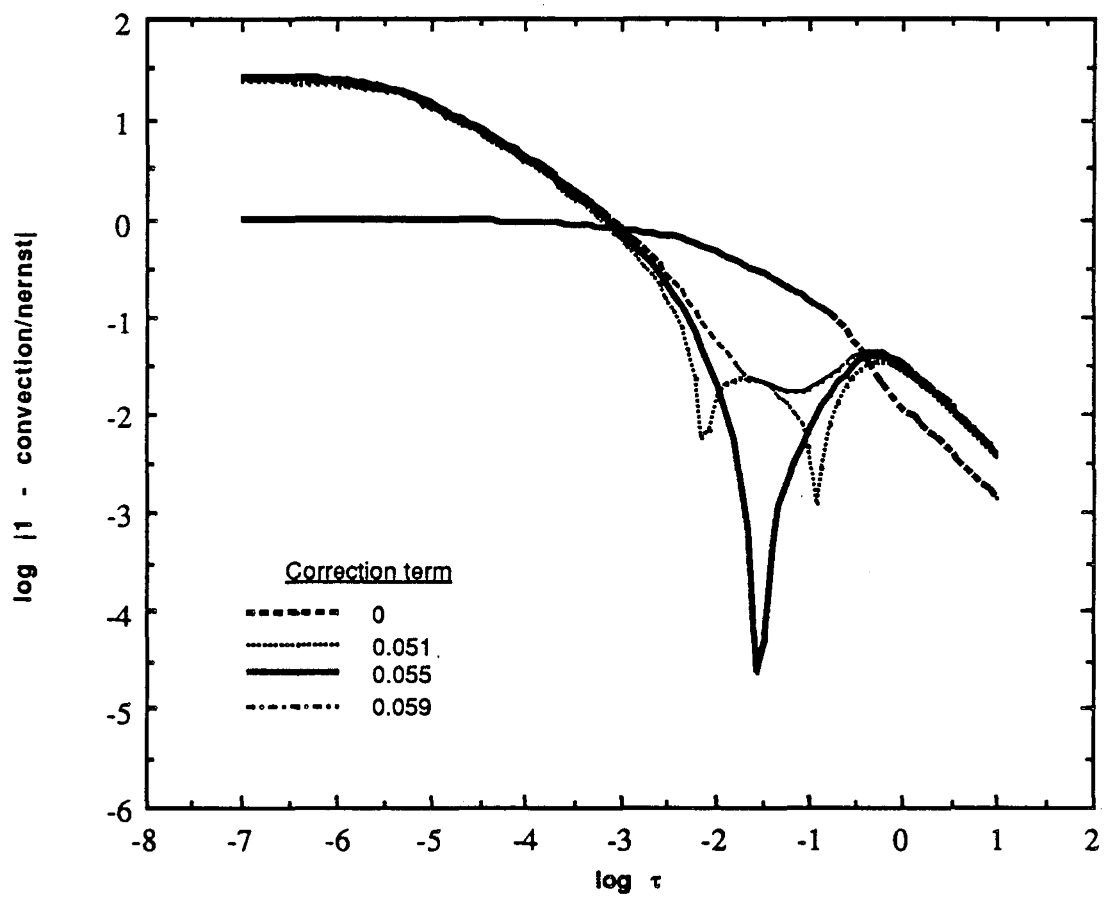


Figure B-5. Coefficients for superposition integral, the correction term 0.055 was added to the convection term. The ordinate is the same as Figure B-4.

$$\tau_S \equiv t\omega \left[\frac{a}{3}\right]^{2/3} Sc^{-1/3} . \tag{B-13}$$

Since

$$\delta = \Gamma\left[\frac{4}{3}\right]\left[\frac{3D}{a\nu}\right]^{1/3}\left[\frac{\nu}{\omega}\right]^{½} , \tag{B-14}$$

and since substituting the value of τ_S into Equation B-13 gives

$$t\omega\left[\frac{a}{3}\right]^{2/3} Sc^{-1/3} \approx 0.035 , \tag{B-15}$$

we determine a threshhold value for τ:

$$\log(\tau) = \log\left[\frac{Dt}{\delta^2}\right] \approx -1.36 . \tag{B-16}$$

This value is close to our choice of $\log(\tau) = -1.5$.

B.6. An alternative approach

The solution to the problem of diffusion into a semi-infinite medium initially at zero concentration from a boundary at unit concentration is

$$c(x,t) = 1 - \text{erf}\left[\frac{x}{\sqrt{4Dt}}\right] . \tag{B-17}$$

We know the fully developed boundary layer thickness δ and ask, "At what time does the semi-infinite diffusion boundary layer grow to this thickness?" To answer the question, we pick a value of the concentration sufficiently close to zero where we can say that the boundary layer ends. Then we use a table of error functions to determine the time at which the boundary layer has this thickness. This represents the longest time at which the semi-infinite diffusion would be appropriate for the rotating-disk problem. Matlosz's choice leads to a value of $c(\delta, \pi/4) = 1 - \text{erf}(0.78) = 0.27$. Clearly, this time is too large because the boundary layer has extended so far that the concentration is about equal to a fourth where it should be nearly zero. If we say that $c(\delta,t)$ should be 0.01, 0.05, 0.10, or 0.50, we find the corresponding times listed in Table B-2. From this we conclude that the rotating-disk solution must be used for times greater than $\delta^2/10D$. Our more rigorous analysis showed that we should switch solutions at $\log \tau = -1.5$,

Table B-2. Time before the boundary layer develops.

$c(\delta,t)$	$\frac{x}{\sqrt{4Dt}}$	t
0.01	1.82	$\frac{\delta^2}{13.25D}$
0.05	1.38	$\frac{\delta^2}{7.62D}$
0.10	1.16	$\frac{\delta^2}{5.38D}$
0.50	0.48	$\frac{\delta^2}{0.92D}$

or a time of $\delta^2/30D$, a time one-third as large.

References

1. Reference 5 in Chapter 3.

2. John Newman, *ChE 230 Course Reader: Theoretical Methods of Chemical Engineering*, University of California, Berkeley (Fall, 1987).

3. Jan Robert Selman, *Measurement and Interpretation of Limiting Currents*, Ph.D. Dissertation, pp. 216 , University of California, Berkeley (1971).

4. *Loc. cit.* p. 225.

APPENDIX C

Calculation of Error Function Complements

The error function complement is needed in the short-time solution to the problem of diffusion through a stagnant film.

Abramowitz and Stegun[1] give the formulas:

$$\sqrt{\pi} z e^{z^2} \operatorname{erfc}(z) \sim 1 + \sum_{m=1}^{\infty} (-1)^m \cdot \frac{1 \cdot 3 \cdots (2m-1)}{\left(2z^2\right)^m} , \tag{C-1}$$

which is valid for $z \rightarrow \infty$ and $|\arg z| < \frac{3\pi}{4}$, and

$$\operatorname{erfc}(z) = e^{-z^2} \sum_{i=0}^{5} a_i t^i , \tag{C-2}$$

where $t = \frac{1}{1 + pz}$ and the a_i's and p are specified constants.

Although Abramowitz and Stegun use infinity as the upper limit in the summation in Equation C-1, the series diverges for finite z. How many terms should we use? We can add terms until the absolute value of a term is larger than that of the previous term. If

$$\frac{|\,\text{mth term}\,|}{|\,(m-1)\text{th term}\,|} = \frac{2m-1}{2z^2} > 1 , \tag{C-3}$$

the mth term should not be used. The upper limit on the summation in Equation C-1 should be the largest integer less than or equal to $z^2 + ½$.

Table C-1 shows how the values calculated by the two formulas compare to each other and to the exact value given by Abramowitz and Stegun.[2] The function $z e^{z^2} \operatorname{erfc}(z)$ is convenient for comparisons because it approaches $1 + \frac{1}{z^2}$ for large z.

Table C-1. A comparison of the the value of erfc(z) calculated different ways.

z	z^2	ze^{z^2}erfc(z)		
		eqn C-1	eqn C-2	exact
2.94	0.1150	0.5362552	0.5364946	0.5361729
2.88	0.1200	0.5352233	0.5353977	0.5351147
2.82	0.1250	0.5342090	0.5343167	0.5340672
2.77	0.1300	0.5332133	0.5332505	0.5330302
2.72	0.1350	0.5322385	0.5321984	0.5320035
2.67	0.1400	0.5312844	0.5311595	0.5309867
2.62	0.1450	0.5303536	0.5301332	0.5299798
2.58	0.1500	0.5294481	0.5291188	0.5289825

Figure C-1 shows how the values of the error function complement calculated both ways compare. Their difference is smallest when the value of the argument is 2.747192. For arguments larger than 2.747192, we use the large-argument asymptote (Equation C-1); for smaller arguments, Equation C-2.

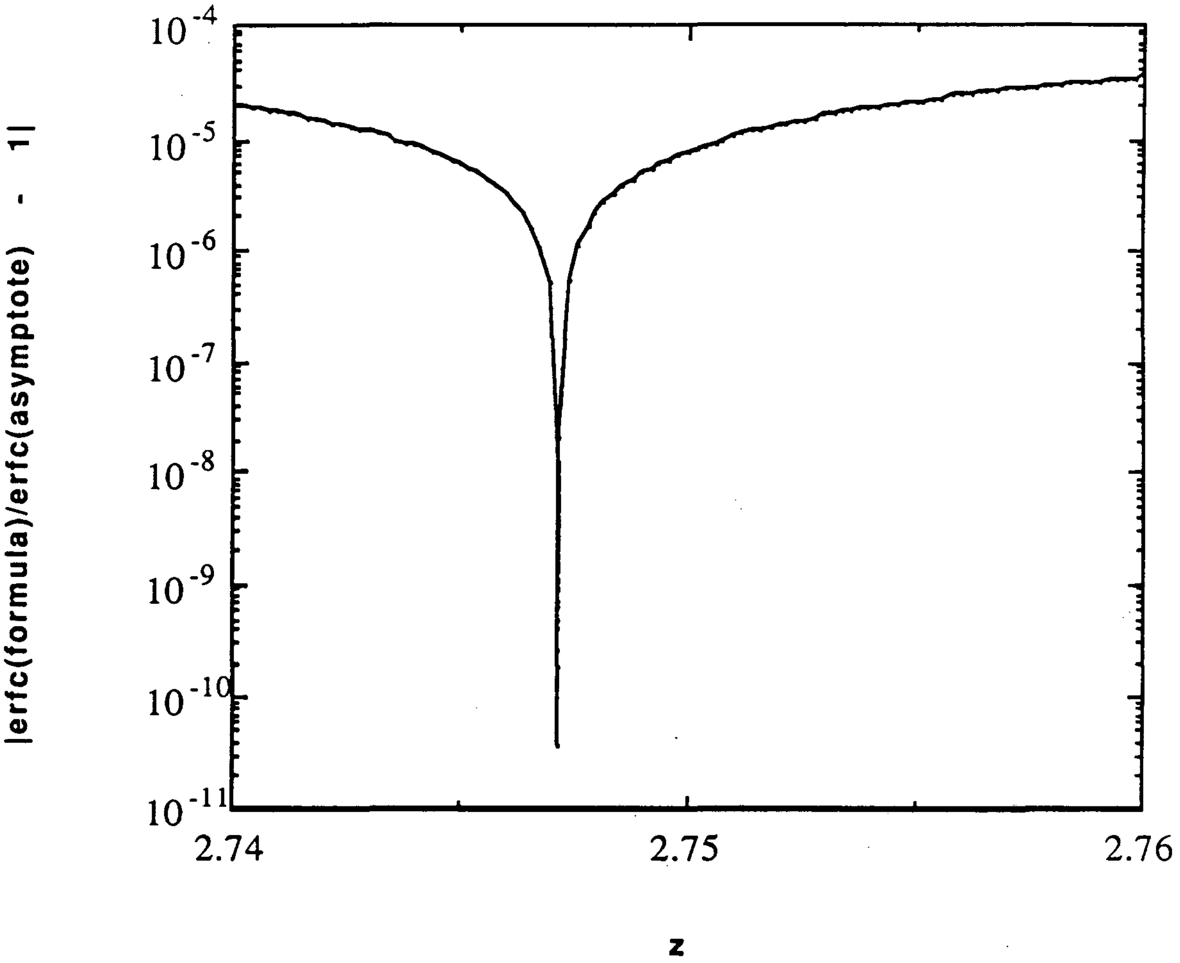


Figure C-1. Comparison of the values of the error function complement calculated by the formula (Equation C-2) and by the large-argument asymptote (Equation C-1).

APPENDIX D

Listing of Program *Pose2Mod*

```
[GFLOATING,ENVIRONMENT('Pose2Mod.pen')]

module Pose2Mod( input, output );

const   StepMax    = 601;
          NMax       = 8;
          NSpcsMax    = 8;
          SystemZero    = 1.0d-100;

type    RealNumber    = double;
          Vector    = array [1..NMax] of RealNumber;
          IVector    = array [1..NMax] of integer;
          Matrix    = array [1..NMax] of Vector;
          IMatrix    = array [1..NMax] of IVector;

          IterationStore    = array [0..StepMax] of integer;
          ErrorStore        = array [0..StepMax] of boolean;
          ResultStore        = array [0..StepMax] of RealNumber;

          SurfaceValueArray    = array [1..NSpcsMax] of ResultStore;

          string    = record    chars        : array [1..100] of char;
                                 length    : integer    end;

var       dtOriginal : RealNumber;


%include '[shain.super]IOPkg.pas'
%include '[shain.super]NewtRaph.pas'
%include '[shain.super]SuperPose2.pas'

end.  { Pose2Mod module }
```

APPENDIX E

Listing of Program *CV*

```
[GFLOATING,INHERIT('[shain.super]Pose2Mod.pen', '[shain.super]PrintCV.pen')]

{ Date last modified: November 2, 1989 }

program cv( input, output );

label    1;

const    R = 8.314;
          F = 96487;
          pi = 3.141592654;
          MuH2O = -237.129e+3;

var       C, dCdx : SurfaceValueArray;

          Co, Cref, ioRef, MuTheta, Utheta, alphaA, alphaC, n, nPrime,
                                                          sH2O : Vector;
          s, p, q : Matrix;
          nSpcs, nRxns, MaximumIterations, TotalSweeps : integer;
          Solutes, SolutesPlusOne : integer;

          Ure, Tolerance, GammaMax : RealNumber;
          t, I, I2, V : ResultStore;

          Iterations : IterationStore;
          Err1, Err2 : ErrorStore;

          tStart, tReversal, tRange, dt : RealNumber;
          StartStep, ReversalStep, CPUTime : integer;
          Sweep : integer;
          Vstart, Vreversal, dV : RealNumber;

          Uref : Vector;
          D, del, z : Vector;

          kappa, radius, b, A, RotationSpeed, omega, rhoZero, Temp, nu
                                                          : RealNumber;

          k : integer;
          kA1, kA2, kD1, kC2, Determ, Sum : RealNumber;


procedure ReadAndPrintParameters;

var       j : integer;

begin
   RI(nSpcs);
   RI(Solutes);
   SolutesPlusOne := Solutes + 1;

   RI(nRxns);
   RI(MaximumIterations);
   RR(Tolerance);
   RR(GammaMax);

   RV(D);
   RV(z);
   RR(Ure);
   RR(Temp);
RR(rhoZero);
RR(nu);

RV(MuTheta);
RV(alphaA);
RV(alphaC);
RV(n);
RV(ioRef);
RM(s);
RV(sH2O);

RV(Cref);
for k := SolutesPlusOne to (nSpcs-1) do
   Cref[k] := 1.0;

RV(Co);

RR(Vstart);
RR(Vreversal);
RR(dV);
RR(b);

RR(radius);
radius := sqrt(1.0/pi);

RR(kappa);
RR(RotationSpeed);
RI(TotalSweeps);

LF(2); TB(20);
write('Parameters:'); LF(3);
TB(20); write('——— Input Data ———');                    LF(2);   TB(10);

write('Number of unknowns (nSpcs) = '); WI(nSpcs,1);    LF(1);   TB(10);
write('Number of solutes = '); WI(Solutes,1);         LF(1);   TB(10);
write('Solutes plus one = '); WI(SolutesPlusOne,1);   LF(1);   TB(10);
write('Number of reactions (nRxns) = '); WI(nRxns,1);   LF(2);   TB(10);
write('MaximumIterations = '); WI(MaximumIterations,1);
                                                         LF(1);   TB(10);
write('Tolerance = '); WR(Tolerance,10,5);             LF(2);   TB(10);
write('GammaMax (mol/cm2) = '); WR(GammaMax,10,5);         LF(2);   TB(10);
write('D[k] (cm2/s) = '); WV(D,10,5,Solutes);          LF(2);   TB(10);
write('z[k] = '); WV(z,4,2,(nSpcs-1));                  LF(2);   TB(10);
write('Ure (V) = '); WR(Ure,10,5);                      LF(2);   TB(10);
write('T (K) = '); WR(Temp,10,5);                       LF(1);   TB(10);
write('rhoZero (kg/cm3) = '); WR(rhoZero,10,5);         LF(1);   TB(10);
write('nu (cm2/s) = '); WR(nu,10,5);                   LF(2);   TB(10);
write('MuTheta[k] (V) = '); WV(MuTheta,10,5,(nSpcs-1));
                                                         LF(1);   TB(10);
write('alphaA[j] = '); WV(alphaA,4,2,nRxns);            LF(1);   TB(10);
write('alphaC[j] = '); WV(alphaC,4,2,nRxns);            LF(1);   TB(10);
write('number of electrons (n) = '); WV(n,10,5,nRxns);LF(1);   TB(10);
write('ioRef[j] (A/cm2) = '); WV(ioRef,10,5,nRxns);   LF(2);   TB(10);
write('s[k,j] = '); WM(s,10,5,(nSpcs-1),nRxns);         LF(2);   TB(10);
write('sH2O[j] = '); WV(sH2O,10,5,nRxns);                LF(3);   TB(10);
write('Cref[k] mol/cm3 = '); WV(Cref,10,5,(nSpcs-1));   LF(3);   TB(10);
write('Co[k] mol/cm3 = '); WV(Co,10,5,Solutes);          LF(2);   TB(10);
write('Vstart (V) = '); WR(Vstart,10,5);               LF(1);   TB(10);
write('Vreversal (V) = '); WR(Vreversal,10,5);         LF(1);   TB(10);
write('dV (V) = '); WR(dV,10,5);                         LF(1);   TB(10);
write('b (V/s) = '); WR(b,10,5);                         LF(1);   TB(10);
```

```
   write('radius (cm) = '); WR(radius,10,5);                LF(1);   TB(10);
   write('A (cm2) = '); WR(A,10,5);                             LF(1);   TB(10);
   write('kappa (mho/cm) = '); WR(kappa,10,5);                LF(1);   TB(10);
   write('RotationSpeed (rpm) = '); WR(RotationSpeed,10,5);
                                                                LF(2);   TB(10);
   write('TotalSweeps (forward and reverse) = '); WI(TotalSweeps,1)

end;   { ReadAndPrintParameters }


procedure SetParameters;

var      i, j, k : integer;

begin   { SetParameters }

   tStart := 0;

   if ( b <> 0 ) then
      begin
         dt := abs(dV /b);
         tReversal := abs( (Vreversal - Vstart) /b )
      end;

   tRange := tReversal - tStart;

   if ( Vreversal > Vstart ) then
      b := abs(b)              {... anodic sweep first }
   else
      b := -abs(b);            {... cathodic sweep first }

   for j:= 1 to nRxns do              { accounts for charge leaving solution,}
      begin                              { and thus for charge transfer in }
         nPrime[j] := 0;              { reactions with n[j] = 0 }
         for i := 1 to Solutes do
            nPrime[j] := nPrime[j] - z[i] * s[i,j]
      end;

   for j := 1 to nRxns do
      begin
         Utheta[j] := 0;
         for i := 1 to (nSpcs - 2) do
            Utheta[j] := Utheta[j] - s[i,j] * MuTheta[i];
         Utheta[j] := Utheta[j] - sH2O[j] * MuH2O;
         Utheta[j] := Utheta[j] / ( nPrime[j] * F )
      end;

for j := 1 to nRxns do
   begin
      Uref[j] := Utheta[j] - Ure;
      for k := 1 to (nSpcs-1) do
         begin
            if ( s[k,j] = 0 ) or ( nPrime[j] = 0 ) then
               Uref[j] := Uref[j]
            else
               begin
                  if ( k <= Solutes ) then
                     Uref[j] := Uref[j] - (s[k,j]*R*Temp /(nPrime[j]*F))
                                                    * ln(Cref[k] /rhoZero)
                  else
```

```
                     Uref[j] := Uref[j] - (s[k,j]*R*Temp /(nPrime[j]*F))
                                                         * ln(GammaMax)
               end
         end
   end;

   for j := 1 to nRxns do
      for k := 1 to (nSpcs-1) do
         if ( s[k,j] > 0 ) then
            begin
               p[k,j] := s[k,j];
               q[k,j] := 0
            end
         else if ( s[k,j] < 0 ) then
            begin
               q[k,j] := -s[k,j];
               p[k,j] := 0
            end
         else
            begin
               p[k,j] := 0;
               q[k,j] := 0
            end;

   omega := RotationSpeed*(2.0*pi) /60.0;

   for k := 1 to Solutes do
      begin
         if ( omega = 0 ) then
            del[k] := 1.0
         else
            del[k] := 1.6117 * ( D[k] ** (1.0 /3.0) ) *
                        ( abs(omega) ** (-1.0 /2.0) ) * ( nu ** (1.0 /6.0) )
      end;

   LF(3); TB(20);
   write(´——— Derived Quantities ———`);                LF(2); TB(10);

   write(´dt (s) = `); WR(dt,10,5);                    LF(2); TB(10);
   write(´Utheta[j] (V) = `); WV(Utheta,10,5,nRxns);   LF(2); TB(10);
   write(´Uref[j] (V) = `); WV(Uref,10,5,nRxns);       LF(2); TB(10);
   write(´nPrime[j] = `); WV(nPrime,10,5,nRxns);       LF(2); TB(10);
   write(´p[i,j] = `); WM(p,10,5,(nSpcs-1),nRxns);     LF(2); TB(10);
   write(´q[i,j] = `); WM(q,10,5,(nSpcs-1),nRxns);     LF(2); TB(10);
   write(´omega (rad/s) = `); WR(omega,10,5);          LF(1); TB(10);
   write(´del[i] (cm) = `); WV(del,10,5,Solutes);      LF(1); TB(10);

end;   { SetParameters }


procedure RunSweep(   tBegin, tEnd : RealNumber;
                      StepBegin : integer;
                      var StepEnd : integer );

var   Step : integer;
      CPUTime : integer;
      j : integer;
      CStep : Vector;
      V : ResultStore;
      IBV : SurfaceValueArray;
```

```
function VSurf ( t : RealNumber ) : RealNumber;
begin
   VSurf := C[nSpcs,StepBegin] + b*(t - tBegin)
end;   { VSurf }


function BVrate( j : integer; C : Vector ) : RealNumber;

var    k : integer;
       PiA, PiC, rA, rC : RealNumber;

begin   { BVrate }
   PiA := 1.0;
   PiC := 1.0;

   for k := 1 to (nSpcs-1) do
      begin
         if ( p[k,j] > 0 ) then
            begin
               if ( C[k] = 0 ) then
                  PiA := 0
               else
                  PiA := PiA*( (C[k]/Cref[k])**p[k,j] )
            end;
         if ( q[k,j] > 0 ) then
            begin
               if ( C[k] = 0 ) then
                  PiC := 0
               else
                  PiC := PiC*( (C[k]/Cref[k])**q[k,j] )
            end
      end;

   rA := PiA*exp( (alphaA[j]*F/(R*Temp))*(C[nSpcs] - Uref[j]) );

   rC := PiC*exp( -(alphaC[j]*F/(R*Temp))*(C[nSpcs] - Uref[j]) );

   BVrate := ioRef[j] * ( rA - rC )
end;   { BVrate }


function SurfBC(   k : integer;
                   C, dCdx, OldC, OlddCdx : Vector;
                   t : RealNumber ) : RealNumber;

var    Sum : RealNumber;
       l : integer;


function FickFlux : RealNumber;
begin
   FickFlux := -D[k]*dCdx[k]
end;   { FickFlux }


function FaradayFlux( C: Vector ) : RealNumber;
var    j : integer;
```

```
        Sum : RealNumber;
begin
    Sum := 0;
    for j:=1  to nRxns do
        Sum := Sum - ( s[k,j] /nPrime[j] /F ) * BVrate(j,C);
    FaradayFlux := Sum
end;


function I( dCdx : Vector ) : RealNumber;
var     k:Integer;
        Sum : RealNumber;
begin
    Sum := 0;
    for k := 1 to Solutes do
        Sum := Sum - F * A * z[k] * D[k] * dCdx[k];
    I := Sum;
end;    { I }


begin    { SurfBC }

if ( k <= Solutes ) then
    SurfBC := FickFlux - FaradayFlux(C)    { = 0 }
else if ( k = nSpcs ) then
    SurfBC := C[nSpcs] - VSurf(t) + I(dCdx)/(4.0*kappa*radius)    { = 0 }
else if ( k = nSpcs - 1 ) then
    begin
        Sum := 1.0;
        for l := SolutesPlusOne to (nSpcs - 1) do
            Sum := Sum - C[l];
        SurfBC := Su    { = 0 }
    end
else
    if ( t < dt ) then
        SurfBC := FaradayFlux(C)    { = 0 }
    else
        SurfBC := 2.0 * GammaMax * ( (C[k] - OldC[k])/dt )
                                   - FaradayFlux(C) - FaradayFlux(OldC)    { = 0 }

end;    { SurfBC }


begin    { RunSweep }

SuperPose2(    nSpcs, nRxns, Solutes, StepBegin, StepEnd,
               D, del, Co, s, nu, omega,
               tBegin, tEnd, dt, C, dCdx, t,
               Iterations, Err1, Err2, CPUTime,
               SurfBC, MaximumIterations, Tolerance);

for Step := StepBegin to StepEnd do
    begin
        I[Step]    := 0;
        I2[Step]   := 0;
        for k:=1 to nSpcs do
            CStep[k] := C[k,Step];
        for k:=1 to Solutes do
```

```
          I[Step] := I[Step] - F * A * z[k] * D[k] * dCdx[k,Step];
       V[Step] := VSurf(t[Step]);
       for j := 1 to nRxns do
          IBV[j,Step] := A * BVrate(j,CStep);
       for j := 1 to nRxns do
          I2[Step] := I2[Step] + IBV[j,Step]
    end;

V[0] := Vstart;

PrintSweep(StepBegin, StepEnd, V, I, I2, IBV, C, dCdx, Iterations,
                                                      Err1, Err2, CPUTime)

end;   { RunSweep }



begin   { cv }

writeln;

PrintTitle;
ReadAndPrintParameters;
SetParameters;

{ Set initial concentrations (for n=0, t=-infinity) to bulk values }

for k := 1 to (nSpcs-1) do
   begin
      C[k,0] := Co[k];
      dCdx[k,0] := 0.0
   end;
C[nSpcs,0] := Vstart;

{   Make guesses to be used for the first time step, i.e. the
      steady-state result.  For subsequent steps, the result of the
      previous step will be used.   }

Co[1] := 5.12579E-003;
Co[2] := 5.63961E-006;
Co[3] := 0.00000E+000;
Co[4] := 1.59517E-007;
Co[5] := 8.00169E-001;
Co[6] := 1.99781E-001;
Co[7] := 4.93045E-005;
Co[5] := Co[6] - Co[7];

Co[nSpcs] := Vstart;

t[0] := tStart - dt;
I[0] := 0;

Err1[0] := false;
Err2[0] := false;
Iterations[0] := 0;

StartStep := 0;

if ( b <> 0 ) then
   for Sweep := 1 to TotalSweeps do
      begin
```

```
            tReversal := tStart + tRange;
            RunSweep(tStart, tReversal, StartStep, ReversalStep);
            tStart := tReversal;
            StartStep := ReversalStep;
            b := -b
        end
else
    begin
        LF(2); TB(20);
        write(‘WARNING: Sweep Rate (b) = 0’);
        LF(1); TB(20);
        write(‘==> dt undefined; PROGRAM EXECUTION HALTED.’);
        LF(5)
    end;

1 : LF(1); TB(20); writeln(‘Program End’);  LF(1)

end.  { cv }
```

APPENDIX F

Listing of Program *SuperPose2*

```
{...

    ==============================================================


    Procedure Title:   SuperPose2
    Modified by:       Paul Shain

    Date last modified:   November 2, 1989

            Based on:       SuperPose
            Written by:      Michael Matlosz
            Date:             December 15, 1984
            Copyright:      (c) 1985 M. Matlosz


    Purpose:       SuperPose  is a  subprogram  for  the solution of the
            unsteady-state,  multi-component  diffusion equations at
            an  electrode  during  cyclic voltammetry.  The routine
            uses a multidimensional Newton-Raphson routine (found in
            procedure  NewtonRaphson of module PoseMod) to determine
            the appropriate  change in concentration of each species
            at each time step,  such that the boundary conditions at
            the electrode  surface (supplied by the calling program)
            are  satisfied.   The  flux  of  each  species  to  the
            electrode  surface  (at each time  interval) is computed
            from the  superposition  theorem  (Duhamel's  integral),
            constructed  from  the  solution  to  the  unsteady-state
            diffusion equation in a semi-infinite stagnant medium or
            to  a rotating-disk  electrode  resulting  from  a  step
            change  in  surface   concentration.  The  integral  is
            evaluated by the method of Acrivos and Chambre.
                 SuperPose2 is a modification of SuperPose.   It is
            used with the program  CV to account  for adsorption and
            desorption of  chemical  species, and  for  chemical and
            electrochemical reaction of adsorbed species.
                 Superpose2  includes  the solution for convective-
            diffusion to a RDE  (in  addition  to  the  Nernst-layer
            solution  found in Superpose).  If the rotation speed is
            zero  the  stagnant-diffusion  case  is  used;  positive,
            convective-diffusion; negative, Nernst layer.


    ==============================================================

                                                              ...}
```

```
procedure SuperPose2(   NSpcs, nRxns, Solutes : integer;
                        nStart : integer;
                        var nStop : integer;
                        D, del, Co : Vector;
                        s : Matrix;
                        nu, omega : RealNumber;
                        tStart, tStop, dt : RealNumber;
                        var c, dcdx : SurfaceValueArray;
                        var t : ResultStore;
                        var Iterations : IterationStore;
                        var Err1, Err2 : ErrorStore;
                        var CPUTime : integer;
                        function SurfBC(   i : integer;
                                           c, dcdx, OldC, Olddcdx
                                                                 : Vector;
                                           t : RealNumber ) : RealNumber;
                        ItMax : integer;
                        Tolerance : RealNumber );

const   pi = 3.14159;
          gamma43 = 0.8928795116;
          aConst = 0.51023;

var     Acoeff : SurfaceValueArray;
        cGuess, cResult, IntegralSoFar, cDeriv, Ginf, ExtraTerm : Vector;
        i, n, ClockStart, ClockStop : integer;
        lambda, B : array [0..9] of RealNumber;


procedure SetCoeffs;

{...    Purpose:   Calculate the various coefficients needed for the
             calculation of the superposition integral.  These values are
             only functions of the index (n - k), where n is the step and
             k is the summation index.  Here, n is set to StepMax so that
             all possible Acoeff[n-k] can be evaluated.
                                                                         ...}

const   n = StepMax;

var     i, k : integer;


procedure SetLambdaB;
begin
   lambda[0] :=    7.21644439;   B[0] := 1.12818046;
   lambda[1] :=    18.1596045;   B[1] := 0.90505798;
   lambda[2] :=    31.1962389;   B[2] := 0.7907692;
   lambda[3] :=    45.7926549;   B[3] := 0.718387;
   lambda[4] :=    61.6691473;   B[4] := 0.666834;
   lambda[5] :=    78.6461928;   B[5] := 0.627481;
   lambda[6] :=    96.5966836;   B[6] := 0.596032;
   lambda[7] :=   115.424957;      B[7] := 0.570071;
   lambda[8] :=   135.05591;     B[8] := 0.548117;
   lambda[9] :=   155.42872;     B[9] := 0.52920
end;    { SetLambdaB }
```

```
procedure SetGinf;

var    System : ( SemiInfinite, RDE, NernstLayer );
       k : integer;

begin   { SetGinf }

   if (omega = 0) then System := SemiInfinite
   else if (omega < 0) then System := NernstLayer
   else System := RDE;

   case System of
      SemiInfinite:
         for k := 1 to Solutes do
            Ginf[k] := 0.0;
      RDE:
         for k := 1 to Solutes do
            Ginf[k] := - (aConst*nu/3.0/D[k]) ** (1.0/3.0)
                                                   * sqrt(omega/nu) * gamma43;
      NernstLayer:
         for k := 1 to Solutes do
            Ginf[k] := - 1.0 / del[k]
   end   { System cases }

end;   { SetGinf }



function a( i : integer; t : RealNumber ) : RealNumber;

var    System : ( SemiInfinite, RDE, NernstLayer );
       tau, Sum, Term : RealNumber;
       m : integer;

begin   { a }

if (t = 0) then a := 0.0
else
begin

   if (omega = 0) then
      System := SemiInfinite
   else if ( t <= exp( -1.5 * ln(10) ) * sqr( del[i]) / D[i] ) then
      System := SemiInfinite
   else if (omega < 0) then
      System := NernstLayer
   else
      System := RDE;

   case System of

      SemiInfinite:
         begin
            a := -2*sqrt( t/(pi*D[i]) )
         end;   { SemiInfinite case }

      RDE:
         begin
            tau := D[i] * t / sqr(del[i]);
            Sum := 0;
            for m := 0 to 9 do
```

```
               begin
                  Term := exp( - lambda[m] * sqr( gamma43 ) * tau );
                  Sum := Sum + B[m]*(1.0 - Term) / lambda[m] / gamma43
               end;
            a := - ( tau + sum ) * del[i] / D[i]

         end;   { RDE case }

      NernstLayer:
         begin
            Sum := 0;
            for m:= 1 to 10 do
               begin
                  Term := exp( - sqr ( m* pi / del[i] ) * D[i] * t )
                                                              / sqr(m);
                  Sum := Sum + Term;
               end;
            a := - t / del[i]
               - (2 * del[i] / D[i] / sqr(pi))
                                * (1.64493406684822643637 - Sum)
         end   { NernstLayer case }

   end   { System cases }

end

end;   { a }


begin   { body of SetCoeffs }

SetLambdaB;
SetGinf;

for k := 0 to (n - 1) do
   for i := 1 to Solutes do
      Acoeff[i,n-k] := a(i,(n-k)*dt) - a(i,(n-k-1)*dt)

end;   { SetCoeffs }


function LeadingTerms( i, n : integer ) : RealNumber;

{...   Purpose:  For a given step n, c ompute the "leading terms" in
         the summation representation of  the  superposition  integral,
         i. e., the sum from k=0 to n-2  (since the total sum runs from
         k=0 to n-1).  Because the weights for the present step are as
         yet unknown, the complete integral cannot be calculated.  How-
         ever,  all terms involving weights  from previous steps can be
         calculated,  and by computing this part of the integral  prior
         to the call to the Newton routine  (which involves iterations)
         the execution time of the program can be reduced substantially.

                                                                  ...}

var   k : integer;
      Sum : RealNumber;

begin   { body of LeadingTerms }
```

```
   if ( n < 2 ) then
      Sum := 0
   else
      begin
         Sum := 0;
         for k:= 0 to (n-2) do
            Sum := Sum + ( (c[i,k+1] - c[i,k])/dt )*Acoeff[i,n-k]
   end;   { else }

   LeadingTerms := Sum

end;   { LeadingTerms }


function CompleteIntegral( i, n : integer;
                           cNew : RealNumber;
                           IntegralSoFar : Vector ) : RealNumber;

{...   Purpose:  From the IntegralSoFar (i. e., the LeadingTerms),
                 compute a complete superposition integral, by adding the
                 last term (i. e., the term for k = n - 1).
                                                                   ...}
var    k : integer;

begin    { CompleteIntegral }
   if (n < 2) then
      CompleteIntegral := 0
   else
      CompleteIntegral := IntegralSoFar[i] +
                             ( (cNew - c[i,n-1]) / dt ) * Acoeff[i,1]
end;   { CompleteIntegral }


function BCTrial( eqn : integer; cTrial : Vector ) : RealNumber;

var    i : integer;
       dcdxTrial, OldC, OlddCdx : Vector;

begin    { body of BCTrial }

   for i := 1 to Solutes do
      begin
         if ( n = 1 ) then
            dcdxTrial[i] :=
                        CompleteIntegral(i,n,cTrial[i],IntegralSoFar)
                                  + ( cTrial[i] - C[i,0] ) * Ginf[i]
               {old version}   {+ ( cTrial[i] - Co[i] ) * Ginf[i]}
         else
            dcdxTrial[i] :=
                        CompleteIntegral(i,n,cTrial[i],IntegralSoFar)
                                  + ExtraTerm[i];
         OlddCdx[i] := dCdx[i,n-1]
      end;
   for i := 1 to nSpcs do
      OldC[i] := c[i,n-1];
   BCTrial := SurfBC(eqn,cTrial,dcdxTrial,OldC,OlddCdx,t[n])
```

```
end;   { BCTrial }


begin   { body of SuperPose2 }

   ClockStart := Clock;
   n := nStart;
   SetCoeffs;

   if ( n = 0 ) then
      begin
         n := 1;
         t[1] := 0.0;
         for i := 1 to Solutes do IntegralSoFar[i] := 0.0;
         {cGuess := Co;}
         for i:=1 to nSpcs do cGuess[i] := C[i,0];   {old version}
         NewtonRaphson( BCTrial, cGuess, NSpcs, Solutes, 50,
               Tolerance, Iterations[1], cResult, Err1[1], Err2[1] );
         for i := 1 to NSpcs do c[i,1] := cResult[i];
         for i := 1 to Solutes do
      ExtraTerm[i] := (c[i,1] - C[i,0]) * Ginf[i];
      {ExtraTerm[i] := (c[i,1] - Co[i]) * Ginf[i];}   {old version}
         for i := 1 to Solutes do dcdx[i,1] := ExtraTerm[i]
      end;

   repeat
      n := n + 1;
      t[n] := t[n-1] + dt;
      for i := 1 to Solutes do IntegralSoFar[i] := LeadingTerms(i,n);
      for i := 1 to NSpcs do cGuess[i] := c[i,n-1];
      NewtonRaphson( BCTrial, cGuess, NSpcs, Solutes, ItMax,
            Tolerance, Iterations[n], cResult, Err1[n], Err2[n] );
      for i := 1 to NSpcs do c[i,n] := cResult[i];
      for i := 1 to Solutes do
         dcdx[i,n] := CompleteIntegral( i, n, c[i,n], IntegralSoFar )
                                                        + ExtraTerm[i];
   until Err1[n] or (t[n] >= (tStop - dt/2)) or (n >= StepMax);

   nStop := n;
   writeln;writeln('nStop (Superpose2) = ',nStop:4);
   writeln;writeln('t[n] = ',t[n]:10:3);
   writeln;writeln('StepMax = ',StepMax:4);
   writeln;
   ClockStop := Clock;
   CPUTime := ClockStop - ClockStart

end;   { SuperPose2 }
```

APPENDIX G

Listing of Program *NewtonRaphson*

```
{...
       ================================================

       Procedure Title:  NewtonRaphson
       Written by:            Michael Matlosz

       Date:        April 17, 1984
       UpDated: August 26, 1984 and December 15,1984

                    Copyright (C) 1985 by Michael Matlosz
                    All rights reserved.

       Modified by:    Paul Shain
       Date:                 August 24, 1989


       Purpose:    Determine the solution vector c[i] that satisfies
          the system of equations F(i,c) = 0.  The technique employed
          is a multi-dimensional Newton-Raphson method.  Derivatives
          are determined numerically, and the coefficient matrix
          (Jacobian) is inverted using the matrix inversion algorithm
          of Newman (MATINV), which is reproduced here in Pascal as the
          subprogram MatrixInversion.  The routine is intended to be
          used with procedure Superpose to solve the multi-component
          diffusion equations occurring in cyclic-voltammetry problems.
          (See module PoseMod for a listing of the SuperPose procedure.)

       ================================================
                                                                        ...}


procedure NewtonRaphson(    function FTrial(    i : integer;
                                                c : Vector ) : RealNumber;
                            cGuess : Vector;
                            N, Solutes, ItLim : integer;
                            Tolerance : RealNumber;
                            var    TotalIterations : integer;
                            var    cResult : Vector;
                            var    Err1, Err2 : boolean        );

label   2;

type    Matrix = array [1..NMax, 1..NMax] of RealNumber;

var     cNew, cOld, cDiff, dcdxOld : Vector;
        Iteration : integer;
        Determinant : RealNumber;

        F : array [1..NMax] of RealNumber;
        dFdc, dFdcInverse : Matrix;

        DetermIsZero : boolean;

        i, j : integer;
```

```
procedure MatrixInversion(    M : Matrix;
                              var Minverse : Matrix;
                              N : integer                 );

{...   Purpose:  Determine the inverse, Minverse, of the square (N by N)
                 matrix M.

       Method:   Gaussian Elimination using elementary row operations.
                 The algorithm is adapted from Newman's subroutine
                 MATINV.  For each of the N row-elimination steps, the
                 following four steps are repeated:

                 Step 1:  Determination of the pivot.  Go through the
                 rows of M, one at a time (skipping rows already used),
                 in order to determine the location of the largest entry
                 (in absolute value) in the row with the smallest ratio
                 of second-largest entry to largest entry (i. e., the
                 smallest ratio NextToMaxEntry/MaxEntry).  Thus, the
                 BestMaxEntry is the largest entry in the row with the
                 BestRatio.  This BestMaxEntry will become the Pivot.
                 (This choice of pivot reduces roundoff error.)

                 Step 2:  Row interchange.  If the BestMaxEntry (the
                 choice for pivot) is not on the diagonal of M, then two
                 rows are interchanged such that BestMaxEntry is on the
                 diagonal.

                 Step 3:  Division by Pivot.  BestMaxEntry becomes the
                 Pivot, and each element of the row containing the Pivot
                 is divided by Pivot.  (The diagonal entry of this row
                 of M is now unity.)

                 Step 4:  Elimination.  All entries in the column
                 containing the Pivot (except the Pivot itself) are
                 eliminated by suitable row multiplications and
                 subtractions.

       Variables global to the routine:  from calling routine —
                        DetermIsZero, Size

                                                                  ...}

label   1;

var     Row, Column, PivotColumn, PivotRow, RowChoice : integer;
        NumberOfRowEliminations, ColWithMaxRowEntry : integer;

        UsedRow, UsedCol : array [1..NMax] of boolean;

        MaxRowEntry, NextToMaxEntry : RealNumber;
        PresentRatio, BestRatio : RealNumber;
        Multiplier, Pivot, SavedValue : RealNumber;


procedure Search(Row:integer);

{...   Purpose:  Search through a row of matrix M to find the largest
           entry of the row (MaxRowEntry) and the second-largest
           entry of the row (NextToMaxEntry).  Also, indicate the
           column containing the MaxRowEntry, and activate the
           DetermIsZero flag if the row contains only zeros.
```

```
    Variables global to the routine:      from MatrixInversion —
                                             M, N, ColWithMaxRowEntry,
                                             MaxRowEntry, NextToMaxEntry,
                                             UsedCol, DetermIsZero

                                                                        ...}

var    Column : integer;

begin    {body of Search}

   MaxRowEntry := 0;
   NextToMaxEntry := 0;

   for Column := 1 to N do
      if not UsedCol[Column] then
         if abs(M[Row,Column]) > MaxRowEntry then
            begin
               NextToMaxEntry := MaxRowEntry;
               MaxRowEntry := abs(M[Row,Column]);
               ColWithMaxRowEntry := Column
            end
         else if abs(M[Row,Column]) > NextToMaxEntry then
            NextToMaxEntry:=abs(M[Row,Column]);

   if MaxRowEntry=0 then
      begin
         DetermIsZero := true
      end

end;    {Search}


begin    {body of GaussElimination}

   {... Initializations ...}

   for Row := 1 to N do
      for Column := 1 to N do
         if (Row = Column) then Minverse[Row,Column] := 1
         else                   Minverse[Row,Column] := 0;

      DetermIsZero := false;

      for Row := 1 to N do UsedRow[Row]:=false;
      for Column := 1 to N do UsedCol[Column]:=false;

   {... Solve the equations ...}

   for NumberOfRowEliminations := 1 to N do

      begin    { row eliminations }

         {... Step 1:  Pivot Determination ...}

         BestRatio := 1.1;     {... setting BestRatio to 1.1 guarantees
                               that the test "if PresentRatio < BestRatio"
                               below will fail on the first pass ...}
```

```
for Row := 1 to N do
   if not UsedRow[Row] then
      begin
         Search(Row);
         if DetermIsZero then goto 1;

         PresentRatio := NextToMaxEntry/MaxRowEntry;

         if PresentRatio <= BestRatio then
            begin
               BestRatio := PresentRatio;

               RowChoice := Row;
               PivotColumn := ColWithMaxRowEntry
            end
      end;

PivotRow := PivotColumn;
UsedCol[PivotColumn] := true;

{... Step 2:  Row Interchange ...}

if RowChoice <> PivotRow then
   begin
      for Column := 1 to N do
         begin
            SavedValue := M[RowChoice,Column];
            M[RowChoice,Column] := M[PivotRow,Column];
            M[PivotRow,Column] := SavedValue
         end;

         for Column := 1 to N do
            begin
               SavedValue := Minverse[RowChoice,Column];
               Minverse[RowChoice,Column] :=
                                  Minverse[PivotRow,Column];
               Minverse[PivotRow,Column] := SavedValue
            end
   end;

UsedRow[PivotRow]:=true;

{... Step 3:  Divide by Pivot ...}

Pivot := M[PivotRow,PivotColumn];

for Column:=1 to N do
   M[PivotRow,Column]:= M[PivotRow,Column]/Pivot;

for Column:=1 to N do
   Minverse[PivotRow,Column]:= Minverse[PivotRow,Column]/Pivot;

{... Step 4:  Elimination ...}

for Row := 1 to N do
   if Row <> PivotRow then
      begin
         Multiplier := M[Row,PivotColumn];
         for Column := 1 to N do
            M[Row,Column] := M[Row,Column]
                             - Multiplier*M[PivotRow,Column];
```

```
                      for Column := 1 to N do
                          Minverse[Row,Column] := Minverse[Row,Column]
                                          - Multiplier*Minverse[PivotRow,Column]
                  end;

        end;   { row eliminations }


1:    if DetermIsZero then
        begin
            Err1 := true;
            writeln; writeln('*****    ZERO DETERMINANT!    *****');
            writeln('Itn= ', Iteration:2);
            write('cOld= '); WV(cOld,15,10,N);
            write('F[i]= '); WV(F,15,10,N);
            writeln('dFdc[i,j] = ') ; WM(dFdc,15,10,N,N);
            goto 2
        end

end;   { MatrixInversion }


function NumericalDeriv(   c : Vector;
                          i, j : integer ) : RealNumber;

{...  Purpose:  Compute dFdc by numerical differentiation   ...}

const   Small = 1e-6;

var    epsilon : RealNumber;
       cIncremented : Vector;

begin   { body of NumericalDeriv }

    cIncremented := c;
    epsilon := abs(c[j])*Small;

    if (epsilon < SystemZero) then epsilon := small;
    if (j = N) then epsilon := 0.0001;
    cIncremented[j] := c[j] + epsilon;

    NumericalDeriv := ( FTrial(i,cIncremented) - F[i] ) / epsilon

end;   { NumericalDeriv }


function Converged( cOld, cNew : Vector ) : boolean;

{...      Purpose:  Determine if a solution has been found   ...}

var       j : integer;

begin
    Converged := true;
    for j := 1 to N do
        if ( abs(cOld[j] - cNew[j]) > Tolerance * abs(cOld[j]) ) then
            Converged := false
end;   { Converged }
```

```
procedure MatrixVectorMult(   M : Matrix;
                              V : Vector;
                              var MV : Vector;
                              N : integer      );

{...  Purpose:  Multiply a square (N by N) matrix M by a vector V
          of length N to produce the vector MV.           ...}

var    i, j : integer;
       Sum : RealNumber;

begin
   for i:= 1 to N do
      begin
         Sum := 0;
         for j := 1 to N do
            Sum := Sum + M[i,j]*V[j];
         MV[i] := Sum
      end
end;   { MatrixVectorMult }


begin   { body of NewtonRaphson }

   Err1 := false;
   Err2 := false;

   Iteration := 0;
   cNew := cGuess;

   repeat

      Iteration := Iteration + 1;
      cOld := cNew;

      for i := 1 to N do
         begin
            F[i] := FTrial(i,cOld);
            for j := 1 to N do
               dFdc[i,j] := NumericalDeriv(cOld,i,j)
         end;


      MatrixInversion(dFdc, dFdcInverse, N);

      MatrixVectorMult( dFdcInverse, F, cDiff, N );

      for j := 1 to N do
         begin
            cNew[j] := cOld[j] - cDiff[j];
            if (j < N) then
               begin
                  if (cNew[j] <= 0) then
                     cNew[j] := 0.1*cOld[j];
                  if (j > Solutes) and (cNew[j] > 1.0) then
                     cNew[j] := 1.0 - 0.1 * (1.0 - cOld[j])
               end
         end;

   until   Converged(cOld,cNew) or (Iteration >= ItLim);
```

```
2:   cResult := cNew;
       TotalIterations := Iteration;

       if (Iteration >= ItLim) then Err2 := true

end;  { NewtonRaphson }
```

APPENDIX H

Listing of Program *PrintCV*

```
[GFLOATING,ENVIRONMENT('PrintCV.pen'),INHERIT('[shain.super]Pose2Mod.pen')]

module PrintCV( input, output );

const   Every = 10.0;

var   Step : integer;
      Counter : RealNumber;


procedure PrintSweep(   InitialStep, FinalStep : integer;
                         V, I, I2 : ResultStore;
                        IBV, C, dCdx : SurfaceValueArray;
                        IterationValue : IterationStore;
                        Err1, Err2 : ErrorStore;
                        ClockTime : integer );



procedure ErrorMessage;
begin
   LF(2); TB(20);
   write('WARNING: Zero determinant found');
   LF(1); TB(20);
   write('                    in MatrixInversion routine.');
   LF(1); TB(20);
   write('PROGRAM EXECUTION INTERRUPTED.');
   LF(2)
end;   { ErrorMessage }


procedure ResultLine;
var    j : integer;
begin
   LF(1);
   write(-V[Step]:5:3,'  ');
   I[Step]  := -I[Step]  * 1.0e3;
   write(I[Step]:13,'  ');
   for j := 1 to 3 do
      IBV[j,Step] := -IBV[j,Step] * 1.0e3;
   for j := 1 to 3 do
      write(IBV[j,Step]:13,'  ');
   for j := 1 to 6 do
      write(C[j,Step]:13,'  ');
   if ( Err2[Step] ) and ( Step <> InitialStep ) then write(' !')
   else write(IterationValue[Step]:2)
end;   { ResultLine }


begin
   LF(5); TB(25); write('Final Results: ');

   if ( C[8,FinalStep] > C[8,InitialStep] ) then
      write('Anodic-direction Sweep')
   else
      write('Cathodic-direction Sweep');

   LF(3); write('-V(V)');
   TB(1); write('-i(mA)');
   TB(1); write('-iBV1');
```

```
    TB(1); write('-iBV2');
    TB(1); write('-iBV3');
    TB(1); write('NO3-');
    TB(1); write('H+');
    TB(1); write('HNO2');
    TB(1); write('H2');
    TB(1); write('NO3-*');
    TB(1); write('Sites');
    TB(1); write('It`ns');
    LF(1);

    if (Step = InitialStep) then
     if ( Err1[Step] ) then ErrorMessage
       else ResultLine;

    Counter := 1.0;
    Step :=0;

    repeat
       Step := Step + 1;
       if (Counter >= 1.0) then
          begin
             ResultLine;
             Counter := 0
          end;
       Counter := Counter + 1.0 / Every;
    until Err1[Step] or (Step  = FinalStep);

    if Err1[Step] then
       begin
          Step:= Step - 1;
          ResultLine;
          ErrorMessage
       end;

    LF(3);
  TB(25); write(' Sweep End  (Execution Time = ');
  WI(ClockTime,1); write(' milli-seconds)'); LF(5)

end;  { PrintSweep }



procedure PrintTitle;

var    DateString, TimeString : packed array [1..11] of char;

begin

    Date(DateString);  Time(TimeString);          LF(5); TB(25);
    write(' Cyclic Voltammogram Program');            LF(2); TB(25);
    write(' Written by:  Michael Matlosz');      LF(2); TB(25);
    write(' Modified by:  Paul Shain');LF(1);     TB(25);
    write(' Last Update: November 2, 1989');      LF(3); TB(40);
    write(' Program begun at ',TimeString);      LF(1); TB(40);
    write('                  on ',DateString);      LF(5)

end;   { PrintTitle }

end. { PrintCV module }
```

APPENDIX I

Listing of Program *IOpkg*

```
{...
          IOPkg:  A collection of useful formatting procedures for
                  input to and output from BandAid calling routines.

   Written by Michael Matlosz
   Last modified by Paul Shain on August 23, 1989
                                                                    ...}


procedure FIND( c : char );   {... find next occurence of a character c }
var    ch : char;
begin
   repeat read(ch) until ( EOF or (ch = c) )
end;


procedure RR( var r : RealNumber );   {... read a RealNumber r }
begin
   FIND('='); read(r)
end;


procedure RI( var i : integer );   {... read an integer i }
begin
   FIND('='); read(i)
end;


procedure StringRead( var s : string );   {... read a string s,
                                                 (generic version,
                                                    used below) }
var    c : char;
       i : integer;
begin
   repeat read(c) until ( c = '<' );
   i := 0;
   repeat
      read(c);
      if ( c <> '>' ) then
         begin
            i := i + 1;
            s.chars[i] := c
         end
   until ( c = '>' );
   s.length := i
end;  { StringRead }


procedure RS( var s : string );   {... read a string s }
begin
   FIND('='); StringRead(s)
end;  { RS }


procedure RRowRead( var v : Vector );   {... read a vector v,
                                              or a row of a matrix
                                           (generic version, used below) }
var    c : char;
       i : integer;
begin
```

```
   i := 0;
   repeat  read(c) until ( c = ´<´ );
   repeat
      i := i + 1;
      read(v[i]);
      repeat  read(c) until ( c = ´,´ ) or ( c = ´>´ )
   until ( c = ´>´ ) or EOF
end;    { RRowRead }


procedure IRowRead( var v : IVector );    {...    read an integer–vector v,
                                                    or a row of a matrix
                                                    (generic version, used
                                                    below) }var    c : char;
      i : integer;
begin
   i := 0;
   repeat  read(c) until ( c = ´<´ );
   repeat
      i := i + 1;
      read(v[i]);
      repeat  read(c) until ( c = ´,´ ) or ( c = ´>´ )
   until ( c = ´>´ ) or EOF
end;    { IRowRead }


procedure RV( var v : Vector );    {... read a vector v }
begin
   FIND(´=´); RRowRead(v)
end; { RV }


procedure RIV( var v : IVector );    {... read an integer–vector v }
begin
   FIND(´=´); IRowRead(v)
end;  { RIV }


procedure RM( var m : Matrix );    {... read a matrix m }
var    c : char;
       j : integer;
begin
   FIND(´=´);
   j := 0;
   repeat  read(c)   until ( c = ´<´ );
   repeat
      j := j + 1;
      RRowRead(m[j]);
      repeat  read(c)   until   ( c = ´,´ ) or ( c = ´>´ )
   until ( c = ´>´ ) or EOF
end;    { RM }


procedure RIM( var m : IMatrix );    {... read an integer–matrix m }
var    c : char;
       j : integer;
begin
   FIND(´=´);
   j := 0;
   repeat  read(c)   until   ( c = ´<´ );
   repeat
```

```
      j := j + 1;
      IRowRead(m[j]);
      repeat  read(c)  until  ( c = ´,´ ) or ( c = ´>´ )
   until ( c = ´>´ ) or EOF
end;   { RIM }


procedure TB( n : integer );   {... print n blank spaces (tab) }
begin
   write(´ ´:n)
end;   { TB }


procedure LF( n : integer );   {... print n blank lines (line feeds) }
var    lines : integer;
begin
   for lines := 1 to n do writeln
end;   { LF }


procedure WR( r : RealNumber; e1, e2 : integer );   {...   write a Real-
                                                              Number r, e1
                                                              and e2 are
                                                              field lengths
                                                                     ...}begin
   if ( abs(r) = 0 ) then
      write(r:e1:e2,´ ´:4)    {... decimal notation ...}
   else if ( abs(r) >= 0.001 ) and ( abs(r) < 0.01 ) then
      write(r:(e1+2):(e2+2),´ ´:2)
   else if ( abs(r) >= 0.01 ) and ( abs(r) < 0.1 ) then
      write(r:(e1+1):(e2+1),´ ´:3)
   else if ( abs(r) >= 0.1 ) and ( abs(r) < 1 ) then
      write(r:e1:e2,´ ´:4)
   else if ( abs(r) >= 1 ) and ( abs(r) < 10 ) then
      write(r:(e1-1):(e2-1),´ ´:5)
   else if ( abs(r) >= 10 ) and ( abs(r) < 100 ) then
      write(r:(e1-2):(e2-2),´ ´:6)
   else if ( abs(r) >= 100 ) and ( abs(r) < 1000 ) then
      write(r:(e1-3):(e2-3),´ ´:7)
   else if ( abs(r) < 0.001 ) or ( abs(r) >= 1000 ) then
      begin
         if ( (e1-e2) >= 3) then
            write(´ ´:(e1-e2-3));
            write(r:(e2+6),´ ´:1)    {... scientific notation ...}
      end
   else
      write(r:e1:e2,´ ´:4);    {... decimal notation ...}
end;   { WR }


procedure WI( i : integer; e1 : integer );   {... write an integer i }
begin
   write(i:e1)
end;   { WI }


procedure WS( s : string );   {... write a string s }
var    i : integer;
begin
   for i := 1 to s.length do write(s.chars[i])
end;  { WS }
```

```
procedure WV(     v : Vector;              {... write a vector v }
                  e1, e2 : integer;
                  l : integer );
var    i : integer;
   begin
   for i := 1 to l do
      begin
         TB(1); WR(v[i],e1,e2)
      end;
   LF(1);
end;    { WV }


procedure WIV(    v : IVector;        {... write an integer-vector v }
                  e1 : integer;
                  l : integer );
var    i : integer;
begin
   for i := 1 to l do
   begin
      TB(1); WI(v[i],e1)
   end;
   LF(1);
end;    { WIV }


procedure WM(     m : Matrix;             {... write a matrix m }
                  e1, e2 : integer;
                  l1, l2 : integer );
var    i1, i2 : integer;
begin
   for i1 := 1 to l1 do
   begin
      LF(1);
      for i2 := 1 to l2 do
         begin
            TB(2); WR(m[i1,i2],e1,e2)
         end
   end;
   LF(1)
end;    { WM }


procedure WIM(    m : IMatrix;    {... write an integer-matrix m }
                  e1 : integer;
                  l1, l2 : integer );
var    i1, i2 : integer;
begin
   for i1 := 1 to l1 do
      begin
         LF(1); TB(15);
         for i2 := 1 to l2 do
            begin
               TB(2); WI(m[i1,i2],e1)
            end
      end;
   LF(1)
end;  { WIM }
```

APPENDIX J

Sample Data File

```
------- Data File for CycVolt Program -------

            (no3 reduction on a RDE)

nSpcs = 7
Solutes = 3
nRxns = 3

MaxIts = 40
Tolerance = 1.0e-8
GammaMax = 1.0e-7

D = < 1.9e-5, 9.312e-5, 1.0e-5 >
z = < -1, 1, 0, -1, -1, 0 >
Ure = 0.2415

T = 298.15
rhoZero = 1.0e-3
nu = 0.994e-2
MuTheta = < -111300, 0.0, 36000, -120.000e+3, -136.470e+3, 0.0 >

alphaA = < 1.0, 0.5, 0.5 >
alphaC = < 1.0, 0.5, 0.5 >
n = < 0, 2, 2 >
ioRef = < 1.0e+3, 1.0e-10, 1.0e-13 >

s = << 1, 0, 0  >,
    < 0, -2, -1  >,
    < 0, 0, 0.5 >,
    < -1, -1, 0  >,
    < 0, 1, -1  >,
    < 1, 0, 1  >>
sH2O = < 0, 1, 1 >

cRef = < 5.126e-3, 0.006e-3, 1.0e-13, 1.0e-13 > mol/cm3
Co = < 5.126e-3, 0.178e-3, 0.0, 0.0 >

Vstart = -0.100
Vreversal = -0.700
dV = 1.0e-3
b = 1.0e-3

radius = 0.56413 cm
kappa = 0.111 mho/cm
RotSpeed = 800
Sweeps = 1

-------- End of Data File ----------
```

APPENDIX K

Sample Output File

```
               Cyclic Voltammogram Program

               Written by:  Michael Matlosz

               Modified by:  Paul Shain
               Last Update: November 2, 1989


                              Program begun at 11:39:24.76
                                           on 21-JAN-1990




          Parameters:


          ----- Input Data -----

Number of unknowns (nSpcs) = 7
Number of solutes = 3
Solutes plus one = 4
Number of reactions (nRxns) = 3

MaximumIterations = 40
Tolerance =     1.000E-008

GammaMax (mol/cm2) =     1.000E-007

D[k] (cm2/s) =     1.900E-005     9.312E-005     1.000E-005


z[k] =  -1.0      1.0      0.00     -1.0      -1.0      0.00


Ure (V) =    0.24150

T (K) =  298.15
rhoZero (kg/cm3) =    1.000E-003
nu (cm2/s) =    0.0099400
```

```
MuTheta[k] (V) =      -1.113E+005       0.00000       3.600E+004    -1.000E+005    -1.365E+005     0.00000

alphaA[j] =  1.0      0.50     0.50

alphaC[j] =  1.0      0.50     0.50

number of electrons (n) =     0.00000        2.0000        2.0000


ioRef[j] (A/cm2) =     1.000E+003     1.000E-010     1.000E-013


    s[k,j] =
 1.0000          0.00000         0.00000
 0.00000        -2.0000         -3.0000
 0.00000         0.00000         0.50000
-1.0000         -1.0000          0.00000
 0.00000         1.0000         -1.0000
 1.0000          0.00000         1.0000


sH2O[j] =     0.00000        1.0000        1.0000



Cref[k] mol/cm3 =     0.0051260      6.000E-006     1.000E-013     1.0000        1.0000        1.0000



Co[k] mol/cm3 =     0.0051260      1.780E-004     0.00000


Vstart (V) =   -0.100000
Vreversal (V) =   -0.70000
dV (V) =    1.000E-003
b (V/s) =    1.000E-003
radius (cm) =    0.56419
A (cm2) =    1.0000
kappa (mho/cm) =    0.11100
RotationSpeed (rpm) =  800.00

TotalSweeps (forward and reverse) = 1
```

```
             ----- Derived Quantities -----

        dt  (s) =    1.0000

        Utheta[j] (V) =     0.11711       1.4178        0.28556


        Uref[j] (V) =    -0.16637       1.0449        0.011220


        nPrime[j] =     1.0000        2.0000        3.0000


        p[i,j] =
     1.0000          0.00000         0.00000
     0.00000         0.00000         0.00000
     0.00000         0.00000         0.50000
     0.00000         0.00000         0.00000
     0.00000         1.0000          0.00000
     1.0000          0.00000         1.0000


        q[i,j] =
     0.00000         0.00000         0.00000
     0.00000         2.0000          3.0000
     0.00000         0.00000         0.00000
     1.0000          1.0000          0.00000
     0.00000         0.00000         1.0000
     0.00000         0.00000         0.00000


        omega (rad/s) =   83.776
        del[i] (cm) =     0.0021787      0.0037009      0.0017591

nStop (Superpose2) =  601

t[n] =    600.000

StepMax =  601
```

```
                    Final Results: Cathodic-direction Sweep

-V(V) -i(mA)        -iBV1        -iBV2        -iBV3         NO3-          H+           HNO2          H2           NO3-*        Sites    It`ns

0.100 0.00000E+000  0.00000E+000 0.00000E+000 0.00000E+000 5.12600E-003 1.78000E-004 0.00000E+000 0.00000E+000 0.00000E+000 0.00000E+000 0
0.100 3.03260E-005 -7.58151E-006 1.51630E-005 2.27445E-005 5.12600E-003 1.78000E-004 8.66804E-012 3.62576E-011 1.00000E+000 7.79166E-013 3
0.110 3.68417E-005 -9.21042E-006 1.84208E-005 2.76312E-005 5.12600E-003 1.78000E-004 1.46893E-012 3.62576E-011 1.00000E+000 1.47664E-012 3
0.120 4.47572E-005 -1.11893E-005 2.23786E-005 3.35679E-005 5.12600E-003 1.78000E-004 3.26940E-012 3.62576E-011 1.00000E+000 2.82101E-012 3
0.190 1.74792E-004 -4.36979E-005 8.73959E-005 1.31095E-004 5.12600E-003 1.78000E-004 3.28473E-011 3.62574E-011 1.00000E+000 3.37751E-010 3
0.200 2.12346E-004 -5.30862E-005 1.06172E-004 1.59260E-004 5.12600E-003 1.78000E-004 4.13894E-011 3.62573E-011 1.00000E+000 6.93448E-010 3
0.210 2.57968E-004 -6.44912E-005 1.28982E-004 1.93477E-004 5.12600E-003 1.78000E-004 5.17666E-011 3.62570E-011 1.00000E+000 1.43472E-009 3
0.320 2.18977E-003 -5.44206E-004 1.08841E-003 1.64556E-003 5.12600E-003 1.77999E-004 4.92156E-010 3.59720E-011 9.99994E-001 5.83494E-006 3
0.330 2.65611E-003 -6.57090E-004 1.31418E-003 1.99902E-003 5.12600E-003 1.77999E-004 5.99353E-010 3.57535E-011 9.99987E-001 1.25482E-005 3
0.440 1.78431E-002 -1.06611E-003 2.13222E-003 1.67770E-002 5.12600E-003 1.77992E-004 5.08146E-009 6.82778E-012 9.87847E-001 1.21526E-002 3
0.450 2.11241E-002 -8.59218E-004 1.71844E-003 2.02649E-002 5.12600E-003 1.77991E-004 6.13941E-009 4.53080E-012 9.82464E-001 1.75361E-002 3
0.560 1.30355E-001 -2.04816E-005 4.09632E-005 1.30335E-001 5.12600E-003 1.77946E-004 3.95444E-008 1.28109E-014 7.49696E-001 2.50304E-001 3
0.570 1.47984E-001 -1.36717E-005 2.73434E-005 1.47970E-001 5.12600E-003 1.77939E-004 4.49015E-008 7.04927E-015 7.01654E-001 2.98346E-001 3
0.580 1.65644E-001 -9.04664E-006 1.80933E-005 1.65635E-001 5.12600E-003 1.77932E-004 5.02704E-008 3.84517E-015 6.47482E-001 3.52518E-001 3
0.680 7.64307E-002 -7.01527E-008 1.40305E-007 7.64306E-002 5.12600E-003 1.77968E-004 2.33154E-008 4.22715E-018 4.23474E-002 9.57653E-001 4
0.690 4.67215E-002 -3.97183E-008 7.94365E-008 4.67215E-002 5.12600E-003 1.77981E-004 1.42699E-008 1.96520E-018 2.12550E-002 9.78745E-001 4
```

CHAPTER 4

Using the Model

4.1. Fitting the experimental results

The model in the previous chapter was developed to test possible mechanisms and sets of kinetic and thermodynamic parameters to find how well each would match the results of the linear sweep voltammetry experiments on the reduction of nitrate in acidic nickel solutions discussed in Chapter 2. There are many reactions that may occur when nitrate is reduced. These reactions and their reduction potentials relative to the reduction of hydrogen are shown in Figure 4-1 for basic solutions and Figure 4-2 for acidic solutions. They may be used as a guide to reduce the number of reactions to a manageable amount. Some reactions can be ruled out as unlikely because of their large potentials (*e.g.*, the reduction of nitrogen). The figures can be used to find reactions to try with the model based on their potentials. If the results from using the model suggests that the potential of a step in the mechanism should have a certain value to match experimental results, one can use the figures to find an appropriate reaction to try instead. Many of the species listed on this figure are known to be present in solution when nitrate or nitrite is reduced.[2]

For each reaction there are several parameters that may be adjusted to fit the experiments. These parameters are listed in Table 4-1. Not all these parameters may be adjusted independently to fit the experimental results. For example, because it is desirable that the reactions be balanced properly, we can not arbitrarily vary the stoichiometric parameters s_{ij} and n_j. We can, however, change these parameters by changing the trial mechanism.

To ensure that nitrate desorbs sufficiently fast to stop the reaction at higher cathodic poten-

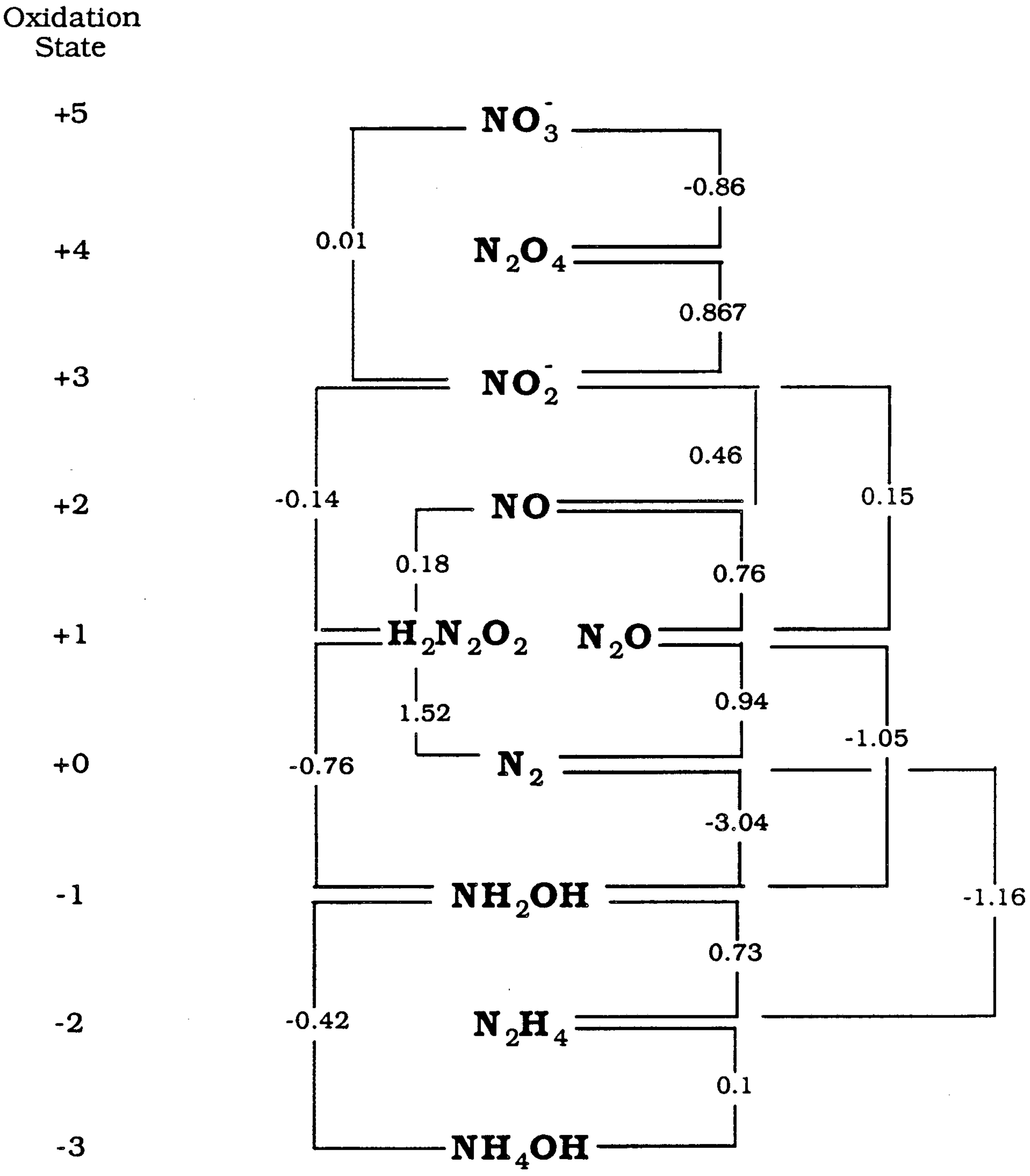


Figure 4-1. Electrode potentials for the reactions of nitrogen compounds in basic solution relative to the hydrogen electrode. (From Reference 1.)

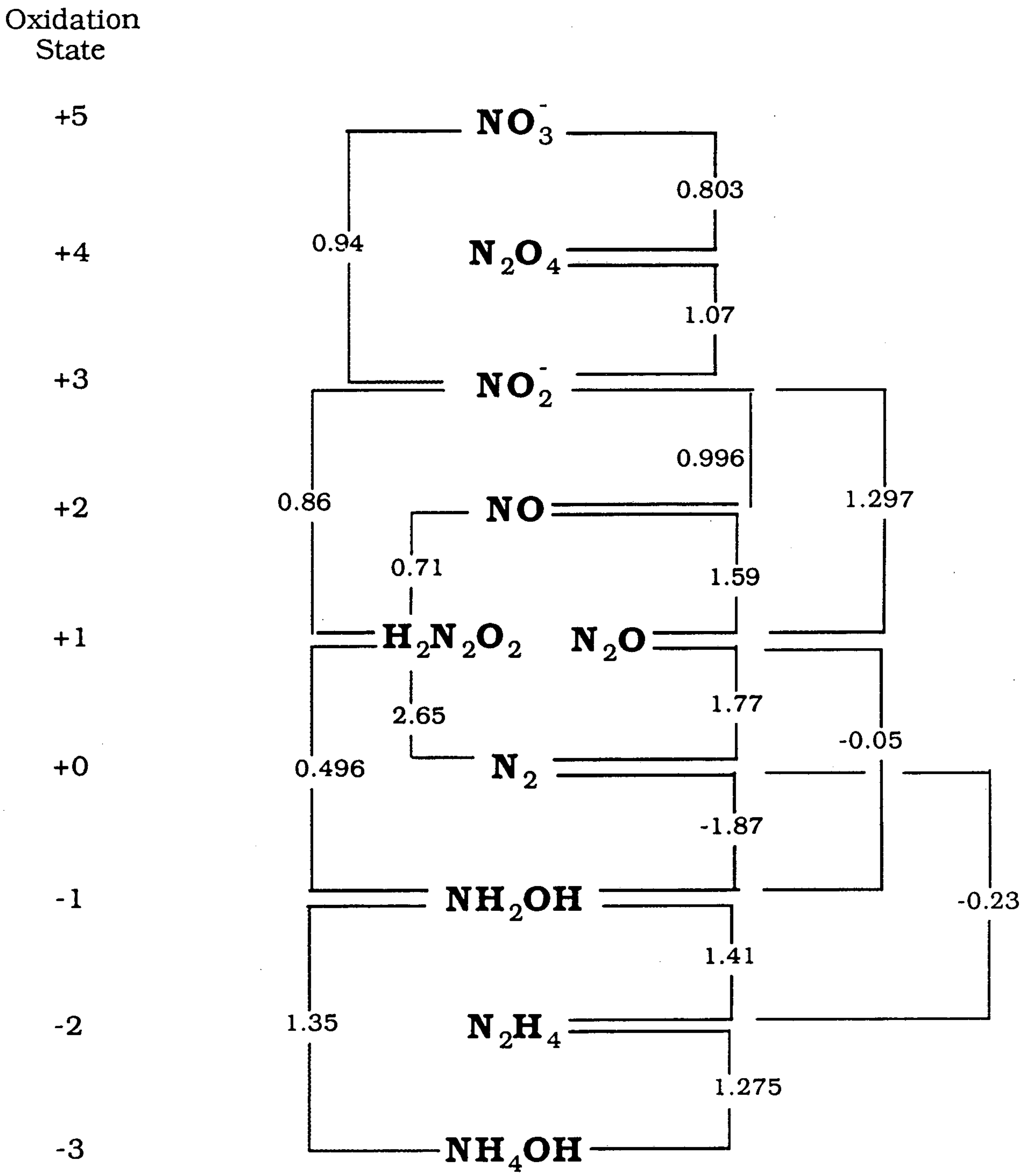


Figure 4-2. Electrode potentials for the reactions of nitrogen compounds in acidic solution relative to the hydrogen electrode. (From Reference 1.)

Stoichiometric	s_{ij}
	n_j
Kinetic	α_{aj}, α_{cj}
	i_{oj}
Thermodynamic	$\mu_i^{\theta}(\mathrm{ads})$
Other	Γ_{max}

Table 4-1. "Adjustable" parameters.

tials, the transfer coefficient for reduction must be less than that for adsorption of nitrate.

The exchange current densities can be varied to change the rates of the reactions. Peak heights often can be changed by changing an exchange current density.

We might try shifting the position of a peak by changing the open circuit potential of a reaction. However, this potential is not an "adjustable" parameter. The program calculates it from the free energies of formation of the species involved in the reaction. This ensures that the equilibrium potentials for all the reactions are consistent. It also changes—and we hope reduces—the number of fitting parameters from one (potential) for each reaction to one (free energy) for each adsorbed species: the formation energies are tabulated for solutes. Because any adsorbate may be involved in more than one reaction (either as a reactant or a product), the equilibrium potentials for the reactions may not be independent. Thus we may not be able to change one without changing another by a proportionate amount. The direction of change may not be the same unless the adsorbate is a reactant or product in both reactions. A change in an equilibrium potential resulting from a change in the free energy of an involved species may result in a shift in the location of a peak, a change in the height of a peak, or both.

The transfer coefficients are adjustable, but we would assume that they are small numbers adding up to the number of electrons in the elementary, rate-controlling step, *i.e.* one or two. As mentioned above, the transfer coefficient for reduction should be less than that for desorption of nitrate. If this is not the case, peaks from different reactant concentrations may not occur at the same potential and may not be caused by the same reaction.

The final unknown parameter that we can change is the maximum concentration of adsorbed species on the surface, Γ_{max}. Based on the size of a nitrate ion, we can estimate that a monolayer of adsorbed nitrate would have a surface concentration of about $3 \cdot 10^{-9}$ mole/cm^2. Bernardi[3] claimed that to predict reasonable currents she had to use the large value of $1.43 \cdot 10^{-6}$ mole/cm^2. We did not find it necessary to use a value of Γ_{max} as high as Bernardi's, but we couldn't use one as low as $3 \cdot 10^{-9}$.

A recent paper by Ho and Jorné examined the impregnation of nickel hydroxide in porous electrodes with and without flow. They credit Kandler[4] with proposing that the precipitation is caused by a change in pH from the reduction of nitrate. They acknowledge the work of others in determining the overall stoichiometry of the reaction under different conditions but recognize that the details of the mechanism are not completely understood.[5]

Bernardi proposed a mechanism that accounts qualitatively for the features of her experimental results.[6] The following four equations are her proposed reaction sequence.

$$NO_3^- + \text{Site} \rightarrow NO_3^-(\text{ads}) \tag{4-1}$$

$$NO_3^-(\text{ads}) + 3H^+ + 2e^- \rightarrow NO_2^-(\text{ads}) + H_2O \tag{4-2}$$

$$NO_2^-(ads) + 7H^+ + 6e^- \rightarrow NH_4^+ + 2H_2O + Site \tag{4-3}$$

$$2H^+ + 2e^- \rightarrow H_2 \tag{4-4}$$

4.2. Difficulties in fitting the experimental data

One difficulty in finding a mechanism and the values of parameters that will fit the experimental results is that the experimental results tell us only the total current, not how much is produced by which reaction or what reactions are taking place. To obtain more information of this kind it is necessary to do additional experiments. In the next section we will suggest some experiments that may answer the questions we have.

A large part of the problem is finding the parameters for a set of reactions that best fit the experimental results. A recent article[7] suggested an interesting method for finding a set of numbers. The method is supposedly based on the principle of natural selection. To use this method one guesses several sets of numbers, uses each, and decides which are best. Each pair of good sets (genes) is then cut into two parts and mated with the other member of the pair. There will be two "offspring": one will contain the first part of the "mother" and the second part of the "father"; the other, the first part of the "father" and the second part of the "mother." Other pairs (of the same numbers) are split in other places and spliced to produce different "offspring." The results of these pairings (the "offspring") are tested and the best are "bred" to produce new sets of numbers until the perfect set is evolved. Provision for mutation may be allowed to avoid dead ends. The algorithm might include a small probability that a number might be changed during the splicing ("mating") process rather than be copied correctly.

4.3. Proposed experiments

The problem of not knowing the partial currents illustrates how using techniques such as photospectrometry and ring-disk electrodes would be helpful.

If any of the products or reactants of any reaction absorbs visible or ultraviolet light, one can conduct the electrochemical experiment in a spectrophotometer and follow the change in concentration of that particular species. This will show the rate of a particular reaction or a partial current.

Instead of starting by reducing the nitrogen species with the highest oxidation state (nitrate), one might start with the species with the next-to-lowest oxidation state (hydrazine) and reduce it to the lowest state (ammonia). In other words we could start at the bottom of Figure 4-2 and work our way up to nitrate one reduction reaction at a time. This way there would be but one reaction with unknown parameters being studied at a time.

Another way to detect and identify intermediate or product species is by using ring-disk electrodes. The reaction of interest can be effected on the disk; the products, detected on the ring.

4.4. Choice of trial mechanism and parameters

To find the mechanism and parameters to fit the experimental data, it is best to start with as simple a case as possible. The mechanism may be made more complicated as reasons are found to justify changes or additions. The mechanism suggested by Bernardi (Equations 4-1 through 4-4) is simple, but it is not the simplest imaginable. The reduction of hydrogen ion can be eliminated until the rest of the mechanism is decided as this will reduce CPU and turn-around time. One could try the mechanism:

$$NO_3^- + \text{Site} \rightarrow NO_3^-(\text{ads}) \tag{4-5}$$

$$NO_3^-(\text{ads}) + 3H^+ + 2e^- \rightarrow HNO_2 + H_2O + \text{Site} \tag{4-6}$$

which could lead to a peak without a shoulder. One advantage of this two-step mechanism is that an expression for the steady-state current may be easily derived and used to determine parameters that will yield a peak of the correct height at the correct potential. The steady-state current is represented by the dashed line on Figure 2-6. The model predicts the steady-state potential at the first time step of a sweep so it can be run at different starting potentials or at a very slow sweep rate (about 1 mV/s here) to find the steady-state current-potential relationship. Examination of this mechanism also led us to conclude the relation of the transfer coefficients is as described above. This mechanism was used with the parameters listed in Table 4-2 to generate Figure 4-3. This figure shows the peak height decreasing to a point with decreasing nitric acid concentration.

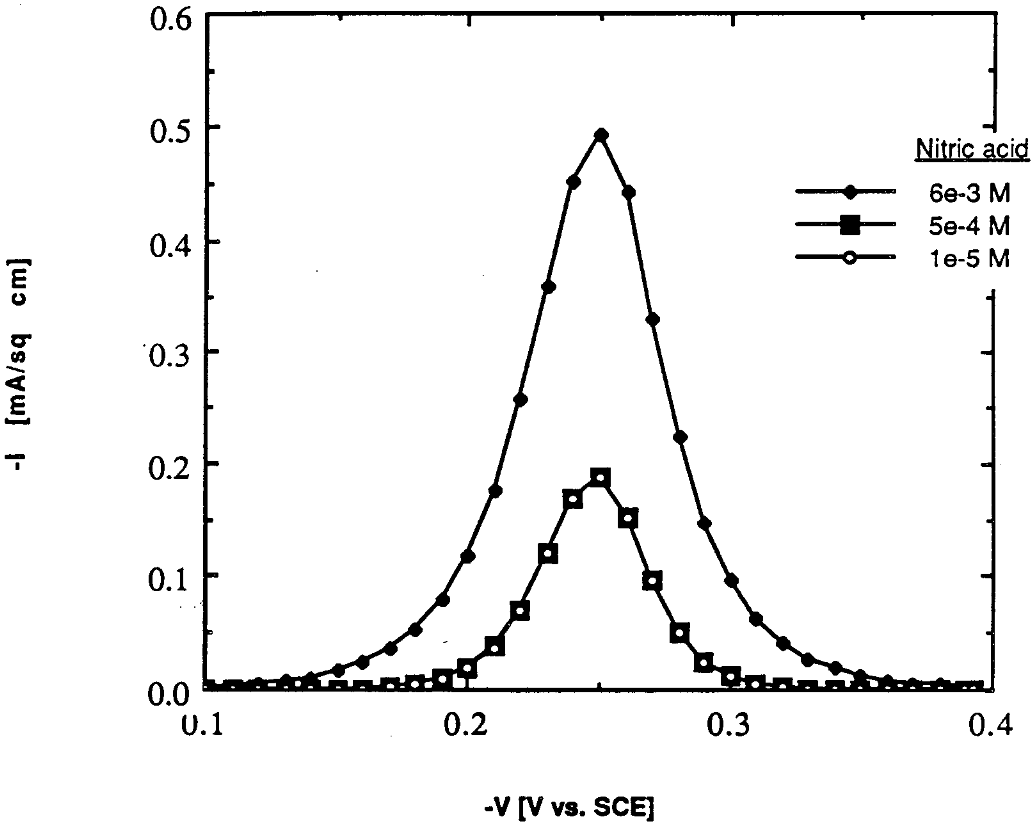


Figure 4-3. Simulated voltammograms for different concentrations of nitric acid. For other parameters see Table 4-2.

$$NO_3^- + \text{Site} \rightarrow NO_3^-(\text{ads}) \qquad (1)$$

$$NO_3^-(\text{ads}) + 3H^+ + 2e^- \rightarrow HNO_2 + H_2O + \text{Site} \qquad (2)$$

Reaction	α_a	α_c	i_o^{ref}, A/cm^2
1	1	1	0.73
2	1	1	10^{15}

species	**D, cm^2/s**	**μ^θ, J/mole**	**c^{ref}, mole/cm^3**	**c_o, mole/cm^3**
NO_3^-	1.9×10^{-5}	-111,300	5.126×10^{-3}	$5.126 \times 10^{-3} + c_{HNO_3}$
H^+	9.312×10^{-5}	0	6×10^{-6}	c_{HNO_3}
HNO_2	10^{-5}	-55,600	10^{-13}	0
NO_3^-(ads)		-107,770		0
Sites		0		0

Table 4-2. Mechanism and parameters used to generate Figure 4-3. The concentrations of adsorbed species are expressed as fractional coverages ($\theta = \Gamma_i/\Gamma_{max}$) and are dimensionless. The reference concentrations of these species were taken to be unity.

Instead of reducing and desorbing nitrate in one step, we can reduce nitrate to another adsorbed species that may be reduced further or desorbed. Such a mechanism is contained in Table 4-3 as are the parameters used to generate Figure 4-4. This figure shows the peak height increasing with increasing nickel nitrate concentration.

The same mechanism was used with the parameters in Table 4-4 to generate Figure 4-5. This figure shows the peak height increasing with rotation speed.

So far we have neglected any changes in solution composition due to hydrolysis. We have taken the initial concentrations of hydrogen and nitrate ions to be equal to the nitric acid and the nitric acid

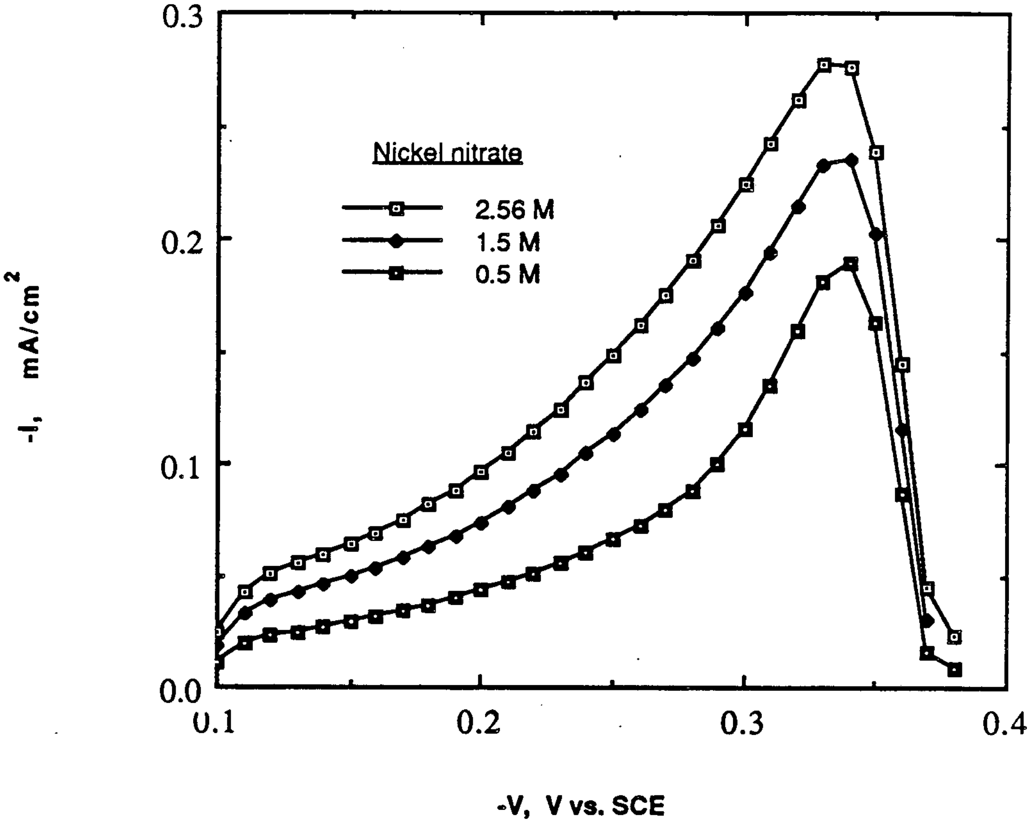


Figure 4-4. Simulated voltammograms for different concentrations of nickel nitrate. For other parameters see Table 4-3.

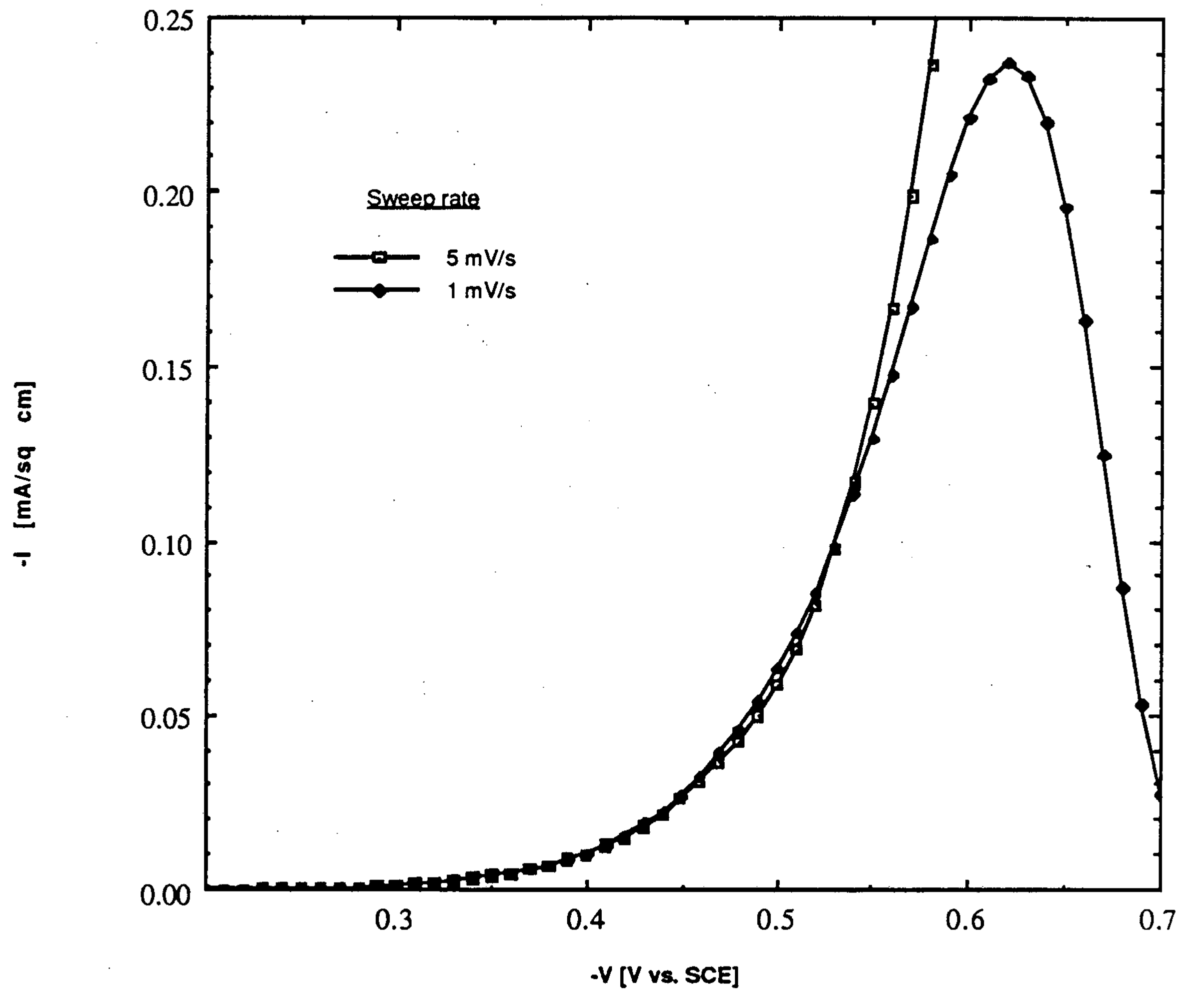


Figure 4-5. Simulated voltammograms for different rotation speeds. For other parameters see Table 4-4.

$$NO_3^- + \text{Site} \rightarrow NO_3^-(\text{ads}) \tag{1}$$

$$NO_3^-(\text{ads}) + 2H^+ + 2e^- \rightarrow NO_2(\text{ads}) + H_2O \tag{2}$$

$$NO_2^-(\text{ads}) + H^+ \rightarrow HNO_2 + \text{Site} \tag{3}$$

Reaction	α_a	α_c	i_o^{ref}, A/cm^2
1	1	1	0.731
2	½	½	10^{-10}
3	1	1	1

species	D, cm^2/s	μ^θ, J/mole	c^{ref}, mole/cm^3	c_o, mole/cm^3
NO_3^-	1.9×10^{-5}	-111,300	5.126×10^{-5}	$6 \times 10^{-6} + 2c_{Ni(NO_3)_2}$
H^+	9.312×10^{-5}	0	6×10^{-6}	6×10^{-6}
HNO_2	10^{-5}	-55,600	10^{-13}	0
$NO_3^-(\text{ads})$		-107,770		0
$NO_2^-(\text{ads})$		-90,000		0
Sites		0		0

Table 4-3. Mechanism and parameters used to generate Figure 4-4.

$$NO_3^- + Site \rightarrow NO_3^-(ads) \quad (1)$$

$$NO_3^-(ads) + 2H^+ + 2e^- \rightarrow NO_2(ads) + H_2O \quad (2)$$

$$NO_2^-(ads) + H^+ \rightarrow HNO_2 + Site \quad (3)$$

Reaction	α_a	α_c	i_o^{ref}, A/cm^2
1	1	1	10
2	½	½	10^{-9}
3	1	1	1

species	**D, cm^2/s**	**μ^θ, J/mole**	**c^{ref}, mole/cm^3**	**c_o, mole/cm^3**
NO_3^-	1.9×10^{-5}	-111,300	5.126×10^{-3}	5.126×10^{-3}
H^+	9.312×10^{-5}	0	6×10^{-6}	6×10^{-6}
HNO_2	10^{-5}	-55,600	10^{-13}	0
$NO_3^-(ads)$		-107,770		0
$NO_2^-(ads)$		-90,000		0
Sites		0		0

Table 4-4. Mechanism and parameters used to generate Figure 4-5.

plus twice the nickel nitrate concentrations, respectively. Bernardi ignored the effects of hydrolysis, but she measured and reported the pH of solutions with different nitric acid and nickel nitrate concentrations.[8] Her results are reproduced in Figure 4-6. It can be seen that the pH of a nickel nitrate, nitric acid solution is not in general the same as the p(HNO_3). This is caused by hydrolysis due to the presence of nickel.

Hydrolysis can be accounted for by reading the pH from Figure 4-6 to get a number to use instead of the nitric acid concentration for the initial hydrogen ion concentration. The mechanism and

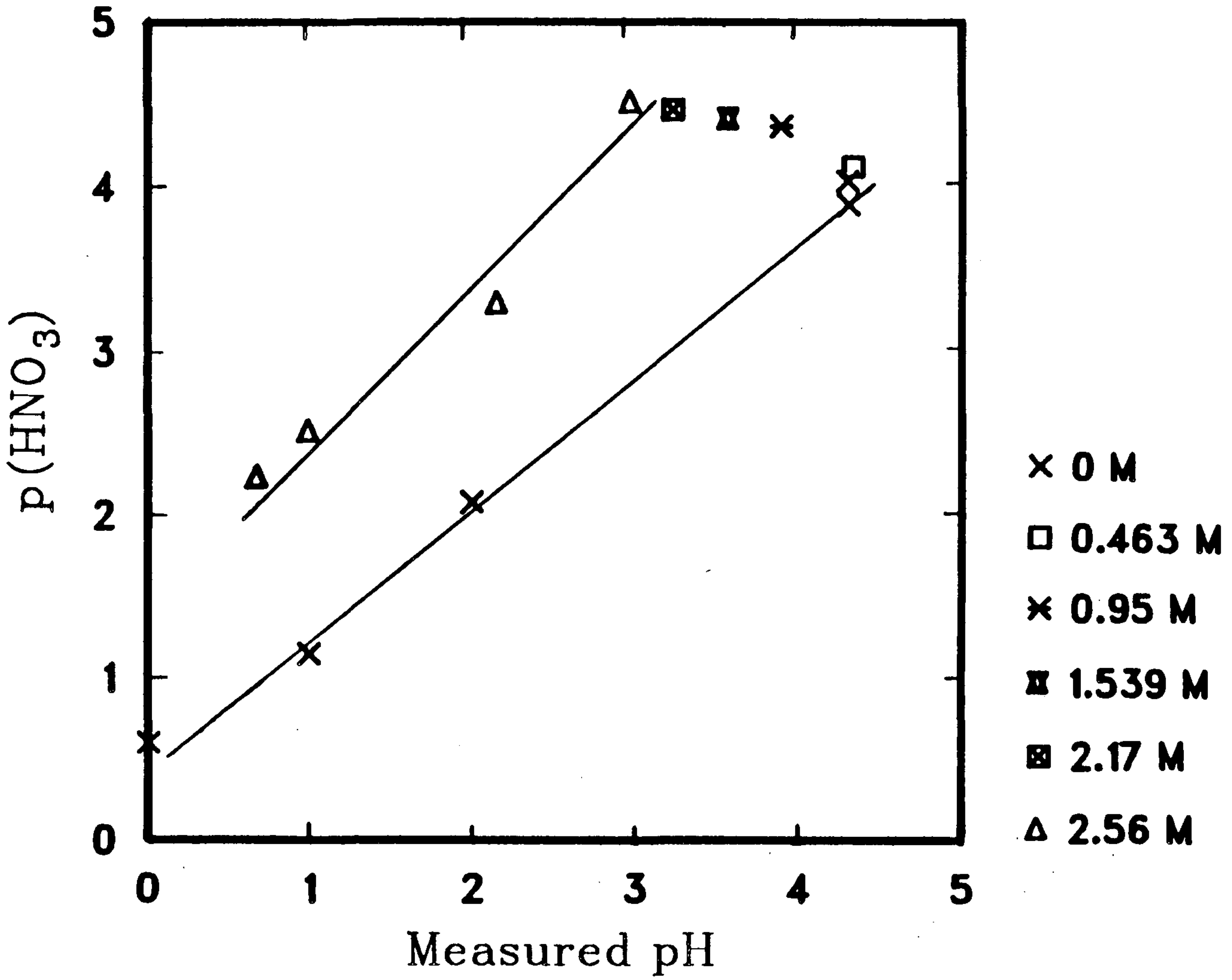


Figure 4-6. Electrometric pH measurement as a function of the concentrations of nitric acid and nickel nitrate. From Reference 7.

parameters (for which hydrolysis is taken into account) in Table 4-5 were used to generate Figure 4-7. This figure shows the effect of changing the sweep rate. The interesting feature is the way the curves overlap. At the high sweep rate, the increase in current occurs at a slightly higher potential than at the slow sweep rate. This agrees with experimental observations, but is not easy to produce with the model.

The experimental results have been qualitatively produced by the model although with different sets of parameters. Finding a mechanism and parameters with a model solely by trial and error is

$$NO_3^- + \text{Site} \rightarrow NO_3^-(\text{ads}) \tag{1}$$

$$NO_3^-(\text{ads}) + 2H^+ + 2e^- \rightarrow NO_2(\text{ads}) + H_2O \tag{2}$$

$$NO_2^-(\text{ads}) + 3H^+ + 2e^- \rightarrow \tfrac{1}{2}H_2N_2O_2 + H_2O + \text{Site} \tag{3}$$

Reaction	α_a	α_c	i_o^{ref}, A/cm^2
1	1	1	1000
2	½	½	10^{-10}
3	½	½	10^{-13}

species	**D, cm^2/s**	**μ^θ, J/mole**	**c^{ref}, mole/cm^3**	**c_o, mole/cm^3**
NO_3^-	1.9×10^{-5}	-111,300	5.126×10^{-3}	5.126×10^{-3}
H^+	9.312×10^{-5}	0	6×10^{-6}	1.78×10^{-4}
$H_2N_2O_2$	10^{-5}	36,000	10^{-13}	0
NO_3^-(ads)		-107,770		0
NO_2^-(ads)		-136,500		0
Sites		0		0

Table 4-5. Mechanism and parameters used to generate Figure 4-7.

difficult. Use of the suggestions for further experimental work discussed in Section 4.3 would make this task easier.

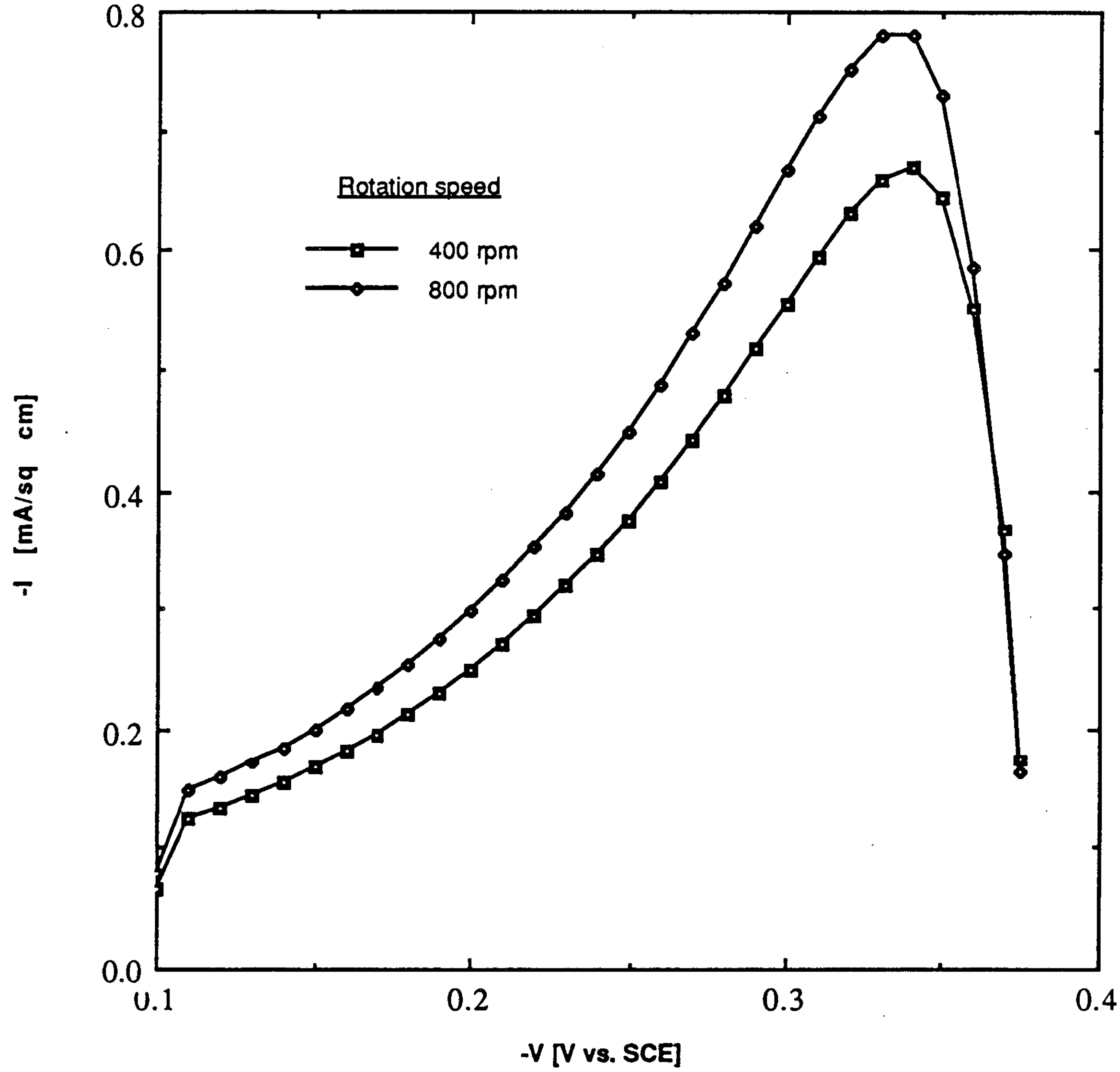


Figure 4-7. Simulated voltammograms for different sweep rates. For other parameters see Table 4-5.

References

1. Joseph T. Maloy, "Nitrogen, Phosphorus, Arsenic, Antimony, and Bismuth," in Bard *et al.*, eds. *Standard Potentials in Aqueous Solutions*, Marcel Dekker, Inc., New York (1985).

2. References 3 through 8 of Chapter 2.

3. Dawn Bernardi, *Mathematical Modeling of Lithium(Alloy), Iron Sulfide Cells and The Electrochemical Precipitation of Nickel Hydroxide*, Ph. D. Dissertation, University of California, Berkeley, pp. 235-6 (1986) [LBL-20858].

4. L. Kandler, Brit. Pat. 917,291 (1963).

5. Kuo-Chuan Ho and Jacob Jorné, "Electrochemical Impregnation of Nickel Hydroxide: Flow-Through *vs.* Stagnant Electrodes," *J. Electrochem. Soc.*, 137, 149-158 (1990).

6. Dawn Bernardi, *op. cit.*, p. 231.

7. Charles T. Walbridge, "Genetic Algorithms: What Computers Can Learn from Darwin," *Technology Review*, pp. 47-53 (January, 1989).

8. Bernardi, *op. cit.*, p. 209.

CHAPTER 5

Electrochemical Packed-Bed Reactors

Electrochemical packed-bed reactors resemble the packed-bed reactors familiar to chemical engineers in many ways: the reactor is filled with a high-surface-area, porous medium on which a chemical reaction occurs and contains void volumes through which a fluid containing a reactant species (electrolyte) flows. The packing may contain or be composed of material with catalytic properties. The advantages of packed-bed reactors with their large amounts of surface area per unit volume are well known to chemical engineers. Unlike the traditional packed-bed reactor, the electrochemical version is the site of an *electrochemical* reaction. Energy is required or released by the reactions not only as heat but also as electricity. Thus the packing and fluid must both be electrically conducting; the packing acts as an electrode on which an electron transfer reaction occurs. In such reactors we are concerned not only with concentration and temperature distributions, but also with current and electric potential distributions. For a complete circuit to exist, the reactor must contain at least one other electrode where another electron transfer reaction will occur. This counterelectrode could be placed upstream or downstream from the first electrode. In either case the current will flow in the same direction as the electrolyte and this configuration is known as a "flow-through" porous electrode. This counterelectrode could be placed in parallel or concentric with the first electrode. In these cases the current will flow perpendicular to the electrolyte; this configuration is known as a "flow-by" porous electrode. The flow-by configuration is inherently two-dimensional and has been analyzed by Alkire and Ng.[1,2] It is not necessary that a flow-by porous electrode have a constant width in the direction of fluid flow (perpendicular to the counterelectrode). Fedkiw and Safemazandarani analyzed this configuration and quoted Kreysa's argument that it more efficiently uses the available electrical driving force and has a higher space-time yield at the limiting current.[3] The electrodes are often separated by an ion-exchange membrane to prevent mixing of anolyte and catholyte.

In this work we are concerned with the flow-through configuration: with the development of mathematical models describing the operation of systems with this configuration and their experimental verification. Models of these reactors are a combination of models of packed-bed (chemical) reactors and of porous electrodes (without convection). Models developed in this work are based on that of Trainham and Newman.[4,5] Newman and Tiedemann reviewed the theory of porous electrodes and its development.[6-8]

Newman and Tobias[9] presented a macroscopic model of the behavior of porous electrodes. In the limit of linear kinetics and constant concentrations, their general model reduces to that of Euler and Nonnenmacher.[10]

The modeling of porous electrodes in this work is based on Trainham's model, which in turn was based on the above-mentioned review by Newman and Tiedemann.[6] By mentioning only the work of our own laboratory, I do not mean to recommend that the work of others be ignored, but to show the development of the model used in the present work.

Trainham solved several problems dealing with porous flow-through electrodes.[5] He estimated the lowest concentration possible in the effluent of a flow-through porous electrode.[11] He developed a model for the removal of metal ions from a dilute stream.[4] The model of the present work is based on this model. Trainham investigated the effect of design and operating parameters on the performance of flow-through porous electrodes.[12] He showed that a flow-through porous electrode would be effective for removing mercury from brine. Both he[13] and Risch[14,15] compared the performance of flow-through and flow-by porous electrodes. Trainham concluded that the flow-by configuration is superior for redox battery applications.

5.1. Uses of flow-through porous electrodes

This section is a summary of the uses of electrochemical, packed-bed reactors. They can be used for chemical synthesis, wastewater treatment, and energy storage (batteries); this last use is what motivated this work.

5.1.1. Chemical synthesis

Porous flow-through (or flow-by) electrodes may be used for electro-organic syntheses. Electrochemical synthetic processes may have the advantages of high yields and (stereo)selectivities (by controlling the electrode potential) and mild operating conditions.[16] Swann and Alkire compiled a bibliography of electro-organic chemical reactions.[17] Textbooks on organic electrochemistry are also available.[18]

Electro-organic syntheses may have other advantages: electricity may be cheaper than other reducing or oxidizing agents, and may eliminate reagents that are dangerous to use or difficult to dispose of. High-priced organics such as pharmaceuticals may be attractive targets for electrochemical routes since the price of electricity will be relatively small.

There are few commercial examples of electro-organic syntheses, the best known is Monsanto's process for producing adiponitrile (a component of nylon). There are fewer still that use a packed-bed electrode. One of these is Nalco's process for producing lead alkyls (*e.g.* tetraethyl lead) from alkyl magnesium halides and the cell's lead pellet anodes.[19] A recent pair of articles presented a model for the syntheses of *2,3*-butanediol, gluconate, and sorbitol in packed-bed flow reactors.[20]

5.1.2. Wastewater decontamination

Studies in this lab have shown how metal ions can be removed from an aqueous solution by depositing them on the surface of a porous electrode. The ions can be removed and concentrated by reversing the cell polarity to dissolve the deposits. Bennion and Newman described using a porous flow-through electrode to remove copper from a dilute solution. Their device reduced the concentration of copper from 667 μg/ml to less than 1 μg/ml.[21] Van Zee and Newman reduced the silver concentration of a simulated photographic fixing solution from 1000 mg/l to less than 1 mg/l.[22] Trost removed up to 98% of the lead from a 1 *M* sulfuric acid solution containing 4 mg/l Pb.[23] Matlosz and Newman used a flow-through porous electrode to remove mercury from brine solutions. They reduced the concentration of mercury in 4.3 *M* sodium chloride solution from 40 ppm to as little as 40 ppb. They also verified the applicability of Trainham and Newman's model.[24,25] Kuhn and Houghton reduce the concentration of antimony in 1 *M* sulfuric acid from 100 ppm to 5 ppm.[26]

A few studies have looked at the possibility of removing metal ions or salt from solution by adsorption in the electrical double layer at the electrode surface. Johnson and Newman presented a model, supported by experimental evidence, for the desalting of water by the sorption of ions in the electrical double layer at the interface between a brine solution and a porous flow-through electrode.[27] Sisler used a similar system to remove zinc from sodium chloride solutions by double-layer adsorption.[28] Alkire and Eisinger presented a model[29] for the potential-dependent adsorption of neutral organic solutes on flow-through porous electrodes and the experimental confirmation of the model[30] for the adsorption of β-naphthol on glassy carbon. Zabasajja and Savinell[31] treated theoretically the potential-dependent adsorption of organics on carbon. They developed an experimental method to determine isotherm parameters and demonstrated it by adsorbing *n*-pentanol and *n*-heptanol on graphite and activated carbon. Mayne and Shackleton[32] used such a process for adsorbing bacteria and proteins.

Packed-bed electrochemical reactors can also be used to destroy cyanide present in electrochemical plating baths. The cyanide can be oxidized to carbon dioxide and nitrogen.[33,34]

5.2. Development of the iron-chromium redox flow battery

This section is a discussion of the use of electrochemical flow cells for energy storage. Redox flow batteries in general and the iron-chromium battery in particular are discussed.

5.2.1. Redox flow batteries

A redox flow cell is an electrochemical reactor used for storing energy by charging and discharging ionic species in solution (*i.e.* a battery). The ionic species constitute redox couples that are completely soluble. This type of battery differs from the more well-known types used in automobiles, watches, and flashlights where the active materials remain more or less fixed in place and are often insoluble. These redox flow batteries could be used for load leveling by electrical utilities or for storing energy produced by intermittent sources, *e.g.* solar cells and windmills.

Flow batteries are promising candidates for large-scale energy storage. In addition to iron-chromium there are zinc-bromine, zinc-chlorine, and zinc-ferricyanide batteries.[35] Flow batteries are attractive for various reasons: cheap reactants, easily mass-produced cell stacks, easy scaleup, simple thermal management owing to the electrolyte circulation, mild operating conditions (allowing cheap materials of construction to be used), no cycle life or depth-of-discharge limitations, and the existence of a technology base in the fuel cells and electrodialysis areas.[34,36]

The reactants are stored in tanks and pumped through the electrochemical cells where they are charged or discharged and then returned to the storage tanks. The bipolar stacks of cells consist of porous electrodes made of graphitized or carbonized felt. The anodes and cathodes are separated by selective, ion-exchange membranes to keep the reactants apart. The electrolyte is fed in parallel through electrodes and flows perpendicular to the direction of current flow (the "flow-by" configuration). A cell is provided to rebalance the state of charge between the positive and negative couples. This may become unbalanced by hydrogen generation in the chromium cells or by the presence of oxygen in the iron solution. Another cell is available to monitor the open-circuit voltage, which is related to the state of charge.

One advantage of this type of battery lies in the separate design of storage and reactor components allowing for simple changes or enlargement. Since the reactants and products are both soluble, there are no phase changes and none of the associated life-limiting processes.[37]

5.2.2. The iron-chromium battery

Several groups have worked on developing iron-chromium redox energy storage systems. NASA developed the system proposed by Thaller.[38,39] Another large effort was put forth by the Electrotechnical Laboratory in Tsukuba, Japan.[40] Several other institutions, including the Universidad de Alicante in Spain, had smaller efforts.[41]

The DOE-supported NASA program to develop a redox flow battery began as a response to the oil embargo in the early seventies and the resulting increase in fuel prices. Large batteries could be used by electric utilities for load-leveling, which would allow the maximum use of the most efficient power plants. They could also be used to store energy produced by intermittent sources. NASA was selected to develop a 1 kW, 13 kWh system because of their expertise in fuel cells from the Apollo project. The iron and chromium couples were chosen for reasons of chemical and materials compatibility, safety, cell voltage, reaction kinetics, and reactant vapor pressure. The half-cell reactions are:

$$\text{Positive electrode} \qquad Fe^{3+} + e^- \underset{\text{charge}}{\overset{\text{discharge}}{\rightleftarrows}} Fe^{2+}$$

and

$$\text{Negative electrode} \qquad Cr^{2+} \underset{\text{charge}}{\overset{\text{discharge}}{\rightleftarrows}} Cr^{3+} + e^- \, .$$

The system was originally intended to operate at ambient temperatures (25° C), but the operating temperature was increased to 65° C to shift the chromium equilibrium from an inert to an electroactive complex. This reduced membrane selectivity but allowed the use of membranes with lower resistivity. Since the membranes are not selective, reactants are mixed: the fully charged positive reactant is 1 *M* $FeCl_2$, 1 *M* $CrCl_3$; the fully charged negative reactant, 1 *M* $FeCl_3$, 1 *M* $CrCl_2$ both with 2 to 3 *N* HCl. The project's final report discusses the advantages and disadvantages of the mixed-reactant operation. It also discusses design considerations and tradeoffs: shunt currents *versus* pumping losses, pumping losses *versus* cell efficiency, flow distribution in the cells, reactant and product mixing in the tanks, and the use of two tanks (for supply and return, instead of returning the reactant to the supply tank).[36]

Johnson and Reid identified the Cr(III) species as $Cr(H_2O)_6^{3+}$ and $Cr(H_2O)_5Cl^{2+}$ and followed the reaction spectrophotometrically to show that it is the divalent species that is the predominant reactant during discharge. $Cr(H_2O)_5Cl^+$ is the predominant reactant during charge. This can be explained by different equilibrium potentials and reduction rates for the hexa- and pentahydrate couples.[42]

The existence of chromium complexes with different electrochemical activities causes problems after several charge-discharge cycles because of the loss of active material.[43,44] The group at the Electrotechnical Laboratory (ETL) in Japan used chromous/chromic chloride and ferric/ferrous chloride in HCl, but unlike NASA used a cation- rather than an anion-exchange membrane. They used electrode materials that did not require electrocatalysts. (NASA's battery required gold and lead: lead to increase the overpotential for hydrogen evolution and gold to catalyze the chromium reaction and to provide a substrate for lead deposition.) The ETL workers increased current and energy efficiencies by adding lead chloride to the electrolyte.[39] They gave no reason for the increase; it is possible that the lead increased the hydrogen overpotential as it did in the NASA battery. They tested different electrode materials and treatments including plasma etching in an oxygen atmosphere. The treatments increased the surface area, introduced functional groups onto the electrode surface, and decreased the internal resistance. They used 3 to 4 *M* HCl with 1 *M* chromous/chromic chloride. Substitution of bromide for chloride ion does not influence the electrode reaction. Substitution of sulfate and especially of perchlorate for chloride increased the cell resistance. Perchlorate seemed to degrade the carbon fabric electrodes.[45] They also described how to monitor state of charge by measuring the open-circuit voltage.[46]

A 1979 cost estimate prepared for NASA concluded that the costs were dominated by the cost of reactants, electrolyte, and storage tanks for both 100-MWh (10 MW for 10 hours) and 500-kWh (10 kW for 50 hours) systems. The large system would have an electric utility load-leveling application; the small, residential or commercial application including stand-alone photovoltaic systems. The major costs are for the chromium chloride reactant and the large tanks needed to store the electrolytes. The electrolytes were assumed to be aqueous solutions of 1 *M* $CrCl_2$ or $FeCl_3$ for the 500 kWh design, 2 *M* for the 10 MWh, both in 2 *M* HCl. In addition to cost, they gave sizes, layout, and methods of manufacture for various components.[35]

Watts and Fedkiw present a mathematical model of NASA's redox flow cell. They conclude that optimum electrode thickness and electrolyte flow rate exist and that the charging procedure affects the amount of hydrogen evolved.[47] They conducted a parametric study to aid in the design, operation, and scale-up of the battery. They concluded that countercurrent flow is better than cocurrent and performance improves at 65° C (as opposed to 25°). They relaxed Trainham and Newman's assumptions of potential variations only in the current flow direction and concentration variations only in the fluid flow direction[12,48]

References

1. Richard Alkire and Patrick K. Ng, "Two-Dimensional Current Distribution Within a Packed-Bed Electrochemical Flow Reactor," *J. Electrochem. Soc.*, **121**, 95 (1974).

2. Richard Alkire and Patrick K. Ng, "Studies on Flow-By Porous Electrodes Having Perpendicular Directions of Current and Electrolyte Flow," *J. Electrochem. Soc.*, **124**, 1220 (1977).

3. Peter S. Fedkiw and Paykan Safemazandarani, "Porous Electrode Theory Analysis of a Variable-Depth, Flow-By Porous Electrode," *Chem. Eng. Commun.*, **36**, 27 (1985).

4. James A. Trainham and John Newman, "A Flow-Through Porous Electrode Model: Application to Metal-Ion Removal from Dilute Streams," *J. Electrochem. Soc.*, **124**, 1528 (1977).

5. James Arthur Trainham, III, *Flow-Through Porous Electrodes*, Ph.D. dissertation, University of California, Berkeley (1979) LBL-9565.

6. John Newman and William Tiedemann, "Porous-Electrode Theory with Battery Applications," *AIChE J.*, **21**, 25 (1975).

7. John S. Newman and William Tiedemann, "Flow-through Porous Electrodes," *Adv. Electrochem. Electrochem. Eng.*, **11**, 353 (1978).

8. John Newman, *Electrochemical Systems*, second ed., Prentice-Hall (in press).

9. John S. Newman and Charles W. Tobias, "Theoretical Analysis of Current Distribution in Porous Electrodes," *J. Electrochem. Soc.*, **109**, 1183 (1962).

10. J. Euler and W. Nonnenmacher, "Stromverteilung in porösen Elektroden," ["Current Distribution in Porous Electrodes,"] *Electrochim. Acta*, **2**, 268 (1960).

11. J. A. Trainham and J. Newman, "A thermodynamic estimation of the minimum concentration attainable in a flow-through porous electrode reactor," *J. Appl. Electrochem.*, **7**, 287 (1977).

12. James A. Trainham and John Newman, "The Effect of Electrode Placement and Finite Matrix Conductivity on the Performance of Flow-Through Porous Electrodes," *J. Electrochem. Soc.*, **125**, 58 (1978).

13. James A. Trainham and John Newman, "A Comparison Between Flow-Through and Flow-By Porous Electrodes for Redox Energy Storage," *Electrochim. Acta*, **26**, 26 (1981).

14. T. Risch, *The Transport Properties of Sodium Polysulfide Melts and A Theoretical Comparison of Flow-Through and Flow-By Porous Electrodes at the Limiting Current*, M.S. thesis, University of California, Berkeley (1983) LBL-17160.

15. Tim Risch and John Newman, "A Theoretical Comparison of Flow-Through and Flow-By Porous Electrodes at the Limiting Current," *J. Electrochem. Soc.*, **131**, 2551 (1984).

16. Gary G. Trost, Victoria Edwards, and John Newman, "Electrochemical Reaction Engineering," in James J. Carberry and Arvind Varma, editors, *Chemical Reaction and Reactor Engineering*, p. 923, Marcel Dekker, New York (1987).

17. Sherlock Swann, Jr., and Richard Alkire, *Bibliography of Electro-Organic Syntheses 1801-1975*, The Electrochemical Society, Inc., Princeton, NJ (1980).

18. Manuel M. Baizer and Henning Lund, editors, *Organic Electrochemistry: An Introduction and a Guide*, Marcel Dekker, New York (1983).

19. Charles L. Mantell, "Electro-Organic Processing: Poised for New Successes?," *Chem. Eng.*, 74, 128 (June 5,1967).

20. Jimmy C. Yu, M. M. Baizer, and Ken Nobe, "Mathematical Modeling of the Paired Electro-organic Syntheses in Packed Bed Flow Reactors: I. 2-Butanone from 2,3-Butanediol, II. Gluconate and Sorbitol from Glucose," *J. Electrochem. Soc.*, **135**, 1392, 1400 (1988).

21. Douglas N. Bennion and John Newman, "Electrochemical removal of copper ions from very dilute solutions," *J. Appl. Electrochem.*, **2**, 113 (1972).

22. John Van Zee and John Newman, "Electrochemical removal of Silver Ions from Photographic Fixing Solutions Using a Porous Flow-Through Electrode," *J. Electrochem. Soc.*, **124**, 707 (1977).

23. G. G. Trost, *Applications of Porous Electrodes to Metal-Ion Removal and the Design of Battery Systems*, Ph.D. dissertation, University of California, Berkeley (1983) LBL-16852.

24. Michael Matlosz and John Newman, "Experimental Investigation of a Porous Carbon Electrode for the Removal of Mercury from Contaminated Brine," *J. Electrochem. Soc.*, **133**, 1850 (1986).

25. M. J. Matlosz, *Experimental Methods and Software Tools for the Analysis of Electrochemical Systems*, Ph.D. dissertation, University of California, Berkeley (1985) LBL-19375.

26. A. T. Kuhn and R. W. Houghton, "Antimony removal from dilute solutions using a restrained bed electrochemical reactor," *J. Appl. Electrochem.*, **4**, 69 (1974).

27. A. M. Johnson and John Newman, "Desalting by Means of Porous Carbon Electrodes," *J. Electrochem. Soc.*, **118**, 510 (1971).

28. G. Sisler, *Adsorption of Dilute-Aqueous Zinc Ions in the Electrical Double Layer of a Porous-Carbon Electrode*, M.S. Thesis, University of California, Berkeley (1987) LBL-24406.

29. Richard C. Alkire and Ronald S. Eisinger, "Separation by Electrosorption of Organic Compounds in a Flow-Through Porous Electrode: I. Mathematical Model for One-Dimensional Geometry," *J. Electrochem. Soc.*, **130**, 85 (1983).

30. Ronald S. Eisinger and Richard C. Alkire, "Separation by Electrosorption of Organic Compounds in a Flow-Through Porous Electrode: II. Experimental Validation of Model," *J. Electrochem. Soc.*, **130**, 85 (1983).

31. J. Zabasajja and R. F. Savinell, "Electrosorption of *n*-Alcohols on Graphite Particles," *AIChE J.*, **35**, 755 (1985).

32. P. J. Mayne and R. Shackleton, "Adsorption on packed bed electrodes," *J. Appl. Electrochem.*, **15**, 745 (1985).

33. D.-T. Chin and B. Eckert, "Destruction of Cyanide Wastes with a Packed-Bed Electrode," *Plat. Surf. Finish.*, **63**, 38 (1976).

34. R. M. Sabarathinam, C. Ahmed Basha, and R Vijayavalli, "Electrochemical Oxidation as a Tool for Pollution Control—Part IV Studies on Double Packed Bed Reactor for the Destruction of Cyanide," *Bulletin of Electrochemistry*, **2**, 143 (1986).

35. James McBreen, "Flow Batteries: Present Status and Research Areas," Abstract No. 113 in *Extended Abstracts Battery Division Electrochemical Society Fall Meeting*, p. 163 (1987).

36. Kenneth Michaels and Gene Hall, *Cost Projections for Redox Energy Storage Systems: Final Report, April 30, 1979—September 30, 1979*, DOE/NASA/0126-1, NASA CR-165260, FCR-1784 (1980).

37. Norman H. Hagedorn, "NASA Redox Storage System Development Project: Final Report," DOE/NASA/1276-24, NASA TM-83617 (1984).

38. Norman H. Hagedorn, *NASA Redox Storage System Development Project: Final Report*, DOE/NASA/12726-24, NASA TM-83677 (1984).

39. L. H. Thaller, "Electrically Rechargeable Redox Flow Cells,rq, *9th Inter-Society Energy Conversion Engineering Conference Proceedings*, pp. 924-928 (1974).

40. K. Nozaki and T. Ozawa, "ETL 1 kW Redox Flow Cell" *Progress in Batteries & Solar Cells*, **5**, 327 (1984).

41. P. Garces, M. A. Climent, M. Lopez Segura, and A. Aldaz, "Redox Flow Battery Using Fe^{3+}/Fe^{2+} and Cr^{3+}/Cr^{2+} Couples," *Extended Abstracts International Society of Electrochemistry 36th Meeting*, Salamanca, Spain, p. 07120 (1985).

42. David A. Johnson and Margaret Reid, "Chemical and Electrochemical Behavior of the Cr(III)/Cr(II) Half-Cell in the Iron-Chromium Redox Energy Storage System," *J. Electrochem. Soc.*, **132**, 1058 (1985) DOE/NASA/12726-17, NASA TM-82913.

43. P. Garces, M. A. Climent, and A. Aldaz, "Sistemas de Almacenamiento de Energía Electrica: I. Primeros Resultados de un Acumulador Redox Hierro-Cromo, ["Electrical Energy Storage Systems: I. First Results from an Iron-Chromium Redox Battery,"] *An. Quim.*, **83**, 9 (1987).

44. M. A. Climent, P. Garces, M. Lopez-Segura, and A. Aldaz, "Sistemas de Almacenamiento de Energía Electrica: II. Acumulador Redox Fe/Cr de Tipo Filtro-Prensa," ["Electrical Energy Storage Systems: II. Filter-Press-Type Iron-Chromium Redox Battery,"] *An. Quim.*, **83**, 12 (1987).

45. K. Nozaki, A. Negishi, H. Kaneko, and T. Ozawa, "Internal Resistance of Redox flow Cells and Characteristics of Carbon Fabrics for Electrodes," Abstract No. 111 in *Extended Abstracts Battery Division Electrochemical Society Fall Meeting*, p. 161 (1987).

46. K. Nozaki, H. Kaneko, Y. Akai, Y. Wada, F. Ohishi, and T. Ozawa, "A Monitor for Redox Flow Battery," Abstract No. 112 in *Extended Abstracts Battery Division Electrochemical Society Fall Meeting*, p. 162 (1987).

47. Rick W. Watts and Peter S. Fedkiw, "A Mathematical Model of NASA's Redox Flow Cell," in Richard C. Alkire *et al.*, editors, *Proceedings of the Symposium on Electrochemical Process and Plant Design*, Proceedings Volume 83-6, The Electrochemical Society, Inc., Pennington, NJ (1983).

48. Peter S. Fedkiw and Rick W. Watts, "A Mathematical Model for the Iron/Chromium Redox Battery," *J. Electrochem. Soc.*, **131**, 701 (1984).

CHAPTER 6

Frequency Response Analysis

This chapter is a discussion of frequency response analysis. A description of the technique is followed by examples of its use in heat transfer and chemical kinetics. Following this background material is a discussion of impedance measurements and their application to electrochemical systems. Our approach to modeling problems in this area is included.

6.1. Description

If one varies periodically some property of a linear system's* input, a property of the output will exhibit a periodic variation at the same frequency. We can measure the system's response and compare it to the input variation. By comparing these measurements, we can determine the gain (ratio of output to input amplitudes) and phase shift (time lag between the input and output) of the system. We hope to learn something about what is happening inside the system by measuring the gain and phase shift at a variety of frequencies.

6.2. Examples

Frequency-response techniques have existed for over a century. In 1861, Ångstrom[1] described a frequency-response method for determining the thermal conductivity of a solid bar by varying the temperature at one end of the bar and measuring the temperature fluctuation at some distance from the end.

*If the system is not linear, it should respond as though it were if the stimulating signal is sufficiently small.

More recently, Naphtali and Polinski[2] introduced a frequency-response technique to heterogeneous catalysis. They used their method to study adsorption kinetics for the hydrogen-nickel system by sinusoidally varying the pressure and following the amount of adsorbed gas.

Frequency-response techniques have been used by Evnochides and Henley[3] to measure solubility and diffusion coefficients of a gas in a polymer. They suspended a polymer in a gas, varied the gas pressure sinusoidally, and measured the variation of the weight of the polymer. The diffusion coefficient for the gas was determined from the phase shift; the Henry's law solubility, from the gain and the phase shift.

Yasuda and coworkers[4-6] measured the pressure oscillations that resulted from sinusoidally varying the volume of a chamber containing zeolite and gas. They then calculated the diffusion coefficient of the gas in the zeolite pores. This technique was used by Goodwin *et al.*[7] to study hydrogen adsorption on CO hydrogenation catalysts.

Li *et al.*[8] determined adsorption and desorption rate constants for carbon monoxide on silica-supported platinum by varying the inlet gas concentration and measuring surface coverage.

The use of chopped molecular beams to supply reactant to a solid target while monitoring the volatile products to gain insight into the reaction mechanisms and kinetics is another example of a frequency response technique in chemical kinetics.[9]

6.3. Impedance

Electrical impedance is a transfer function: it relates the response of an electrical circuit (a network of resistors, capacitors, inductors, voltage sources, current sources, *etc.*) to an excitation. Usually, the properties of the individual elements of these networks, which may be either lumped or distributed, are constant and independent of the frequency of the excitation signal. In these systems, one measures either the voltage response to a change in current, or the current response to a change in voltage. The impedance (or transfer function) of a circuit is the ratio of the voltage perturbation to the current perturbation.

6.4. Electrochemical impedance

Electrochemical systems also have impedances. Impedance measurements made over a range of frequencies can separate the effects of the various processes that occur in these systems if the time constants of the processes are sufficiently different. This type of measurement is also referred to as frequency-response analysis or electrochemical impedance spectroscopy. People have studied the response of electrochemical systems to alternating currents for nearly a century.[10-13]

Small-amplitude perturbations are applied so that the frequency response of a system will be linear. Linear systems respond to a signal of frequency ω only at that frequency. If the sinusoidal perturbation is too large, the nonlinear response will not be sinusoidal and will contain components of frequencies that are multiples of the frequency of the applied signal, *i.e.* the harmonic frequencies. This property can be used to determine if a sufficiently small perturbation is being used. Since linear systems don't respond at second (or higher) harmonics, a response at a frequency of 2ω shows that the system is not linear and a smaller excitation should be used.

Bard and Faulkner[14] discussed different types of voltammetry based on the concept of impedance. With an excitation of frequency ω one can measure the direct current response. With excitations of frequency ω_1 and ω_2 one can measure the response at frequencies ω_1-ω_2 or ω_1+ω_2. The advantage to these techniques is that the double-layer capacitance is more linear than the charge-transfer processes so its effects are eliminated from these measurements. The disadvantage is that the (non-linear) analysis is more difficult.

Impedance measurements can be used with a mathematical model to determine physical properties of components of and chemical mechanisms occurring in the system being studied. Because it is a nonintrusive, nondestructive diagnostic technique, it could be applied to quality control in plating, battery manufacturing, or electrolytic or hydrometallurgic processes.

Suchanski[15] claimed that frequency-response analysis can determine the state of charge of lithium/carbon monofluoride batteries. However, according to Sandifer,[16] the state-of-charge of these batteries can not be determined by impedance techniques as attempted by himself and Suchanski[17] because of the insensitivity of the impedance parameters or because of time-after-discharge dependences. He claimed that chronopotentiometry is a good way to measure the state-of-charge. (He drained the batteries at a constant current and recorded the potential drop.)

Jardy *et al.*[18] studied the improvement in corrosion resistance in galvanized steel caused by a coating of zinc diphosphate. From their impedance data, they developed a criterion for quality control.

O'Keefe *et al.*[19] used cyclic voltammetry to monitor additive concentrations in an electrolyte. They proposed comparing a polarization curve obtained from a solution of unknown composition with those obtained from standard solutions to determine the additive content of the solution. Since automated impedance experiments are fast, it might be better to use this instead of cyclic voltammetry.

Another electrochemical transfer function is the electrohydrodynamic impedance. This function relates the current response of a rotating-disk system to perturbations in the rotation speed. This technique was originated by Tokuda *et al.*[20] Tribollet and Newman[21] claimed that this method is useful for determining diffusion coefficients from Schmidt numbers and found excellent agreement between theory and experiment for a redox reaction below the limiting current under galvanostatic regulation.

6.5. Equivalent circuits

Electrochemical systems may behave, or be conceived of as behaving, like a network of electrical circuit elements such as resistors, capacitors, inductors, *etc.* These analogies are useful conceptually: they allow us to think of the response of an electrochemical system to an alternating current (or voltage) signal in terms that are familiar to us. They are useful pedagogically: they present a convenient starting point for learning about electrochemical impedance. However, they may not be the best way to interpret the frequency response of electrochemical systems.

We believe in interpreting the frequency response of electrochemical systems by comparing them to predictions based on the application of fundamental knowledge of the phenomena known to occur in the system. If we can develop a mathematical model of a system that behaves as a real system, we can be certain that we understand what is happening in the real system. One reason for avoiding the equivalent circuit analogy is that to reproduce the behavior of an electrochemical system, often one must either use a circuit network that does not contain a finite number of elements, or use a network containing elements for which their values (resistance or capacitance) are functions of frequency unlike real elements. Another reason is that mathematical models can be changed easily to reflect a chemical or geometric change in a system by changing physical properties input or boundary conditions; this is not necessarily true with equivalent circuits. Or conversely one wants to be able to extract from experiments or include in a model independently determinable physical properties of the components of the system, not the size of electric circuit elements that have no direct connection to the problem of interest. Grahame recognized this in 1952:[22]

> Many persons who have worked on this problem in the past have attempted to represent the behavior of a metal-solution interface by means of an equivalent electrical circuit. The objection to this procedure is that one has no way of knowing whether or not a given equivalent circuit is, in fact, equivalent to the interface under consideration except by carrying out an independent analysis of the problem which it is the objective of those who use this method to avoid. It will be found, in fact, that in most instances no finite combination of resistors, condensers, and inductors can represent the variation with frequency of a metal-solution interface across which a current flows. Indeed, there is no reason to expect that it should be otherwise, except that we have not heretofore met a case where the equivalent circuit concept has been demonstrated to be fallacious.

Impedance data can be displayed graphically in many forms. Two popular ways of plotting data are Nyquist plots— -Im(Z) *vs.* Re(Z)—and Bode plots—both |Z| and ϕ *vs.* frequency. It is often difficult to determine resistances and capacitances of an equivalent circuit from these plots. These values are are found from the plots' slopes, inflection points, and extrema. Resistances and capacitances thus found must then be related to physical properties of the system such as diffusivities, conductivities, exchange current densities, open-circuit potentials, *etc.* It is easy to compare graphically model predictions with experimental results.

6.5.1. Examples

What are the impedances of resistors and capacitors? Ohm's law applies to resistors; their impedance is the same as their resistance, a real number or a point on the real axis in the complex plane.

For capacitors the charge stored is proportional to the applied voltage: $q=CV$. This relation can be differentiated with respect to time to find that the current is proportional to the rate of change of the potential:

$$i = C\frac{dV}{dt} . \qquad (6\text{-}1)$$

If the applied voltage is a sine wave of frequency ω and magnitude V,

$$v = V\sin(\omega t) , \qquad (6\text{-}2)$$

and the current response has the magnitude I, frequency ω, and phase shift ϕ:

$$i = I\sin(\omega t+\phi) , \qquad (6\text{-}3)$$

then v from Equation 6-2 can be substituted into Equation 6-1 to obtain:

$$i = CV\frac{d}{dt}\sin(\omega t) = CV\omega\cos(\omega t) = CV\omega\sin(\omega t-\frac{\pi}{2}) . \qquad (6\text{-}4)$$

So the magnitude of the current response $I=\omega CV$ and the phase shift is -90°. The magnitude of the impedance is given by $|Z|=V/I=1/\omega C$. In the complex plane the magnitiude of the impedance will be inversely proportional to the frequency and the direction will always be in the negative imaginary direction. (Rotating by 90° is the same as multiplying by -j or dividing by j.) Thus the capacitor's complex impedance is

$$Z = \frac{1}{j\omega C} . \qquad (6\text{-}5)$$

This is the negative imaginary axis in the complex plane.

6.5.2. Coordinate system

We have described the relation or transfer function between these signals in terms of a magnitude (gain) and an angle (phase shift) as though using polar coordinates. We could also use the in- and out-of-phase components, or the real and imaginary parts, as though using Cartesian (rectangular) coordinates.

We prefer to discuss impedance in terms of its real and imaginary parts rather than its gain and phase shift because of the advantages in using rectangular instead of polar coordinates. Using rectangular coordinates means replacing sine and cosine functions with exponential functions containing exp(jωt). This has some of the advantages found in using Laplace transforms: we use frequency as the independent variable instead of time, and partial derivatives of a quantity with respect to time become the product of jω and that quantity. Another advantage is that the exp(jωt) terms cancel. The most important advantage is that it is much easier to work with exponentials than with the sines and cosines of composite angles.

These points can be demonstrated by considering the impedance of a capacitor using complex notation. The variables are, in general, the sum of constant and transient terms. The transient term is the product of a complex function of position and $e^{j\omega t}$, a sinusoidal variation in time.[23] For the voltage this gives

$$v = \bar{v} + \tilde{v}e^{j\omega t} . \tag{6-6}$$

The current can be expressed in these terms and is proportional to the rate of change of the voltage:

$$i = \bar{i} + \tilde{i}e^{j\omega t} = C\frac{dv}{dt} = C\tilde{v}j\omega e^{j\omega t} . \tag{6-7}$$

The impedance is the ratio of the voltage and current perturbations, giving

$$Z = \frac{\tilde{v}}{\tilde{i}} = \frac{1}{C\omega j} = -\frac{j}{C\omega} \tag{6-8}$$

as before.

6.5.3. Networks

Electrochemical systems often behave similarly to simple networks of resistors and capacitors. The ohmic drop is a resistance to the flow of current. The charge transfer (faradaic) resistance also acts as a resistor. The double layer acts as a capacitor. A resistor in series with a resistor and capacitor in parallel can represent the ohmic drop of a solution in series with the parallel double-layer capacitance and charge-transfer resistance at the solution-electrode interface. This representation is the "Randles circuit" shown in figure 6-1. The impedance of this circuit is a semicircle with a diameter equal to the charge-transfer resistance lying above the positive real axis in the complex (-Im(Z) *vs*. Re(Z)) plane. The semicircle is displaced from the origin a distance equal to the ohmic resistance. The frequency of the maximum in the curve is the reciprocal of the time constant for the parallel resistor and capacitor.

Porous electrodes are often approximated as transmission lines, since there are two parallel conductors: a solid phase and an electrolyte phase, connected by a faradaic resistance and a double-layer capacitance. Cheng discusses the behavior of infinite and finite transmission lines.[24]

6.5.4. Warburg impedance

Two circuit elements commonly used in constructing "equivalent circuits" are inductances and the Warburg impedance. Inductance is the complement to capacitance: the potential is proportional to the speed at which the current changes. Diffusion to a planar surface with a sinusoidal flux (current) or driving force (concentration, potential) causes what is known as the Warburg impedance. The response is a sine wave which decreases in amplitude with distance from the surface and which has a 45-degree phase shift (*i.e.* the real and imaginary parts of the impedance are equal). This problem is analogous to determining the temperature profile into earth due to temperature variation at the surface.[25,26]

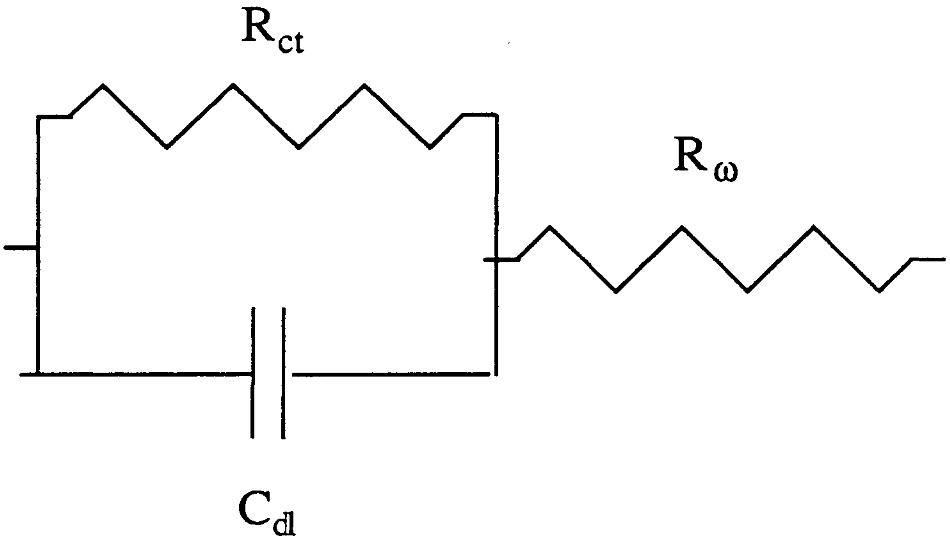


Figure 6-1. Randles equivalent cicuit.

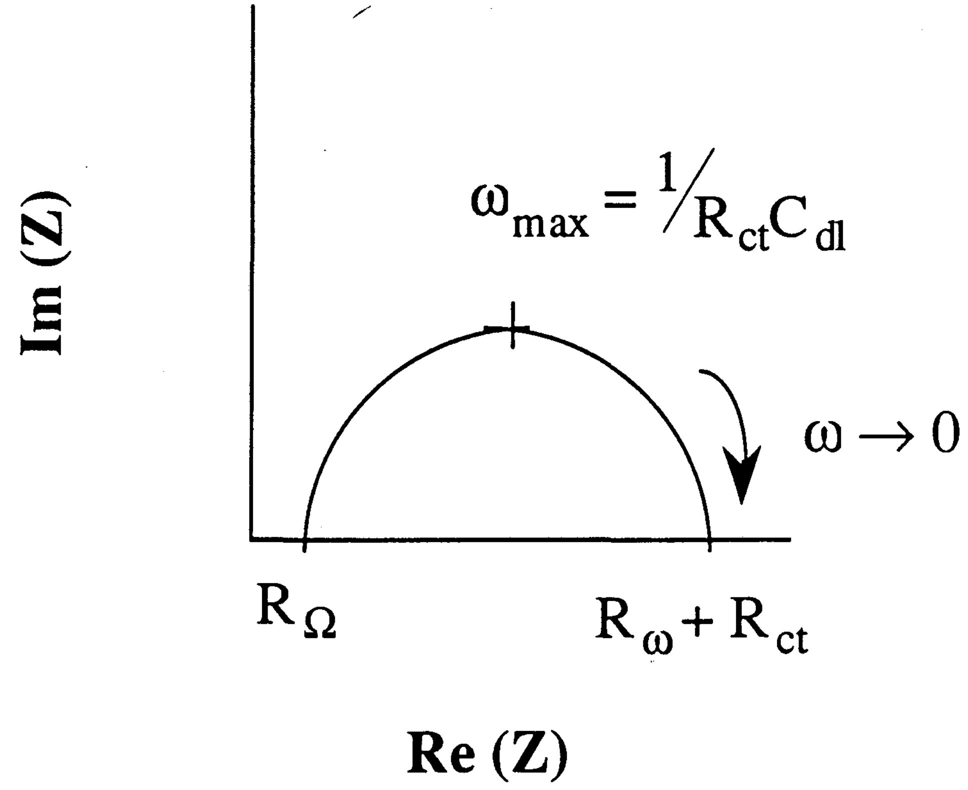


Figure 6-2. Nyquist plot of the impedance of the Randles equivalent circuit in Figure 6-1.

References

1. A. J. Ångstrom, "Neue Methode, das Wärmeleitungsvermögen der Körper zu bestimmen," ["A New Method of Determining the Thermal Conductivity of a Body,"] *Ann. Phys. Chem.*, **114**, 398 (1861).

2. L. M. Naphtali and L. M. Polinski, "A Novel Technique for Characterization of Adsorption Rates on Heterogeneous Surfaces," *J. Phys. Chem.*, **67**, 369 (1963).

3. S. K. Evnochides and E. J. Henley, "Simultaneous Measurement of Vapor Diffusion and Solubility Coefficients in Polymers by Frequency Response Techniques," *J. Polym. Sci.: Part A-2*, **8**, 1987 (1970).

4. Yusuke Yasuda, "Determination of Vapor Diffusion Coefficients in Zeolite by the Frequency Response Method," *J. Phys. Chem.*, **86**, 1913 (1982).

5. Yusuke Yasuda and Goichi Sugasawa, "A Frequency Response Technique to Study Zeolitic Diffusion of Gases," *J. Catal.*, **88**, 530 (1984).

6. Yusuke Yasuda and Akemi Yamamoto, "Zeolitic Diffusivities of Hydrocarbons by the Frequency Response Method," *J. Catal.*, **93**, 176 (1985).

7. J. G. Goodwin, Jr., J. E. Lester, G. Marcelin, and S. F. Mitchell, "Frequency Response Chemisorption Studies of Carbon Monoxide Hydrogenation Catalysts," in *Catalyst Characterization Science, ACS Symp. Ser.* **288**, p.67 (1985).

8. Yao-En Li, David Willcox, and Richard D. Gonzalez, "Determination of Rate Constants by the Frequency Response Method: CO on Pt/SiO_2," *AIChE J.* **35**, 423 (1989).

9. James A. Schwarz and Robert J. Madix, "Modulated Beam Relaxation Spectrometry: Its Application to the Study of Heterogeneous Kinetics," *Surf. Sci.*, **46**, 317 (1974).

10. Max Wien, "Ueber die Polarisation bei Wechselstrom," ["On Polarization in the case of Alternating Current,"] *Ann. Phys. Chem.*, **58**, 37 (1896).

11. E. Warburg, "Ueber das Verhalten sogenannter unpolarisirbarer Elektroden gegen Wechselstrom," ["On the Behavior of So-Called Unpolarizable Electrodes Toward Alternating Current,"] *Ann. Phys. Chem.*, **67**, 493 (1899).

12. Elsa Neumann, "Ueber die Polarisationscapacität umkehrbarer Elektroden," ["On the Polarization Capacity of Reversible Electrodes,"] *Ann. Phys. Chem.*, **67**, 500 (1899).

13. E. Warburg, "Ueber die Polarisationscapacität des Platins," ["On the Polarization Capacity of Platinum,"] *Ann. Phys. Chem.*, **6**, 125 (1901).

14. Allen J. Bard and Larry R. Faulkner, *Electrochemical Methods:* Fundamentals and Applications, John Wiley & Sons, 318ff, 354ff (1980).

15. Mary R. Suchanski, "AC Impedance of the Carbon Monofluoride Electrode," *J. Electrochem. Soc.*, **132**, 2059 (1985).

16. James R. Sandifer, "State-of-charge measurement of the lithium/carbon monofluoride battery by chronopotentiometry," *J. Appl. Electrochem.*, **16**, 307 (1986).

17. J. R. Sandifer and M. R. Suchanski, "Electrochemical characteristics of the lithium/carbon monofluoride battery and its component half-cells," *J. Appl. Electrochem.*, **14**, 329 (1984).

18. A. Jardy, R. Rosset, and R. Wiart, "Diphosphate coatings for protection of galvanized steel: quality control by impedance measurements," *J. Appl. Electrochem.*, **14**, 537 (1984).

19. T. J. O'Keefe, S. F. Chen, E.R. Cole, Jr., and M. Dattilo, "Electrochemical monitoring of electrogalvanizing solutions," *J. Appl. Electrochem.*, **16**, 913 (1986).

20. Koichi Tokuda, Stanley Bruckenstein, and Barry Miller, "The Frequency Response of Limiting Currents to Sinusoidal Speed Modulation at a Rotating Disk Electrode," *J. Electrochem. Soc.*, 122, 1316 (1975).

21. Bernard Tribollet and John Newman, "The Modulated Flow at a Rotating Disk Electrode," *J. Electrochem. Soc.*, **130**, 2016-2026, 1983.

22. David C. Grahame, "Mathematical Theory of the Faradaic Admittance (Pseudocapacity and Polarization Resistance)," *Journal of the Electrochemical Society*, **99**, 370C (1952).

23. Bernard Tribollet and John Newman, "Impedance Model for a Concentrated Solution: Application to the Electrodissolution of Copper in Chloride Solutions," *J. Electrochem. Soc.*, 131, p. 2781 (1984).

24. David K. Cheng, *Analysis of Linear Systems*, Addison-Wesley, Reading, Mass. (1959).

25. R. Byron Bird, Warren E. Stewart, and Edwin N. Lightfoot, "Problem 11.L Periodic Heating of the Earth's Crust," *Transport Phenomena*, Wiley & Sons, New York, 1960.

26. H. S. Carslaw and J. C. Jaeger, *Conduction of Heat in Solids*, 2nd ed., pp. 64-70,81-83, Clarendon Press, Oxford (1986).

CHAPTER 7

Rotating Disk Experiments[†]

To determine transport and kinetic parameters for use in an iron redox battery model, we conducted linear sweep voltammetry experiments on a solution of 0.25 M $FeCl_3$, 0.25 M $FeCl_2$, and 1 M HCl. The model is developed in Chapter 8. We determined three parameters: the diffusion coefficient of the ferric-containing species, the cathodic transfer coefficient, and the exchange current density.

7.1. Theoretical

The diffusion coefficient was calculated from the Levich equation once the limiting current was measured as a function of the rotation speed:

$$I_{\ell,d} = 0.62nFAD^{2/3}\nu^{-1/6}\omega^{1/2}C_b \,, \tag{7-1}$$

where $I_{\ell,d}$ is the limiting current to the disk (mA); C_b, the bulk concentration of the electro-active species (M); A, the electrode area (cm^2); D, the diffusion coefficient (cm^2/s); ν, the kinematic viscosity (cm^2/s); and ω, the angular velocity of the electrode (rad/s). A plot of limiting current *versus* square root of rotation speed yields a straight line. The diffusion coefficient was calculated from the slope.

The kinetic parameters we determined were the cathodic transfer coefficient and the exchange current density. The cathodic transfer coefficient is the fraction of the overpotential that is applied to the cathodic part of the reaction. The anodic transfer coefficient can be calculated from the cathodic coefficient since the sum of the two equals unity (assuming the reaction is an elementary, or one-step, process). The exchange current density is the forward or (equal) reverse reaction rate at the equilibrium potential.

[†] I am grateful to Mr. John Sukamto for his careful performance of these experiments.

Both the cathodic transfer coefficient and the exchange current density were calculated from the Butler-Volmer equation:

$$i = i_o\left[e^{(1-\alpha)nf\eta} - e^{-\alpha nf\eta}\right] , \qquad (7\text{-}2)$$

where α is the cathodic transfer coefficient; f = F/RT = 38.92 V (at 25° C); i_o, the exchange current density (A/cm^2); n, the number of electrons transferred; and η, the overpotential (V). (The overpotential is the difference in potential between the electrode and the solution adjacent to it. It is equal to the applied potential difference between the working and reference electrodes corrected for the ohmic potential drop in the solution. This correction is important because not all the applied potential is available to drive the reaction.) At high (positive or negative) overpotentials one exponential term can be neglected, leaving either

$$i = i_o e^{(1-\alpha)nf\eta} \text{, or} \qquad (7\text{-}3)$$

$$i = i_o e^{-\alpha nf\eta} . \qquad (7\text{-}4)$$

A plot of log $|i|$ *versus* overpotential (a Tafel plot) has linear branches at high overpotentials. The cathodic transfer coefficient can be calculated from the slope of the cathodic branch (negative overpotential); extrapolation of either branch to zero overpotential yields the exchange current density.

The Tafel plot is only valid at high overpotentials. To use lower overpotentials, the Butler-Volmer equation can be rewritten as

$$i = i_o e^{-\alpha nf\eta}\left[e^{nf\eta} - 1\right] . \qquad (7\text{-}5)$$

Thus a plot of log $\left|\frac{i}{e^{nf\eta} - 1}\right|$ *versus* overpotential (an Allen-Hickling plot) yields a straight line. The cathodic transfer coefficient can be calculated from the slope; the exchange current density can be calculated from the intercept (the value of the ordinate at zero overpotential).

7.2. Experimental

The equipment used included: a Pine Instruments RD3 potentiostat, two Keithley 173A multimeters, a Hewlett Packard 7047A x-y recorder, a Nicolet 206 digital oscilloscope, a Pine Instruments

PIR rotator, and a Pine Instruments glassy carbon rotating-disk electrode. The potentiostat regulated the potential difference between the working and (calomel) reference electrode. The multimeters were used to monitor the voltage and the current that were recorded by the oscilloscope and the x-y recorder.

The electrolyte was a solution of 0.25 M $FeCl_3$, 0.25 M $FeCl_2$, and 1 M HCl.

Since we could not measure the complete anodic plateau, we applied our linear voltage sweeps to the cathodic plateau. An optimal sweep rate (one that was neither so slow that it would waste time, nor so fast that the reaction could not keep pace) was found by sweeping at various rates. A rate as high as 7 V/min could be used without changing the results of the sweep. A faster sweep rate might have been used but the chosen rate was convenient for use with the oscilloscope.

Limiting currents were determined for different rotation speeds: 4900, 3600, 2500, 1600, 900, and 400 rpm. The solution was purged with nitrogen for thirty minutes between each set of six sweeps and the working electrode was and polished between sweeps. The limiting currents are plotted *versus* the square root of the rotation speeds in Figure 7-1.

The oscilloscope recorded the currents to a precision of 1 part in 2048 on a scale of ±1 V or about three significant figures.

7.3. Conclusions

The slope of the line in Figure 7-1 gave a diffusion coefficient of 4.20×10^{-6} cm^2/s. Diffusion coefficients in aqueous solutions are usually of this order of magnitude, *i.e.*, 10^{-5}.

The Tafel plot (Figure 7-2) and the Allen-Hickling plot (Figure 7-3) were used to calculate the kinetic parameters. The cathodic transfer coefficient was 0.26; the exchange current density, 0.227 mA/cm^2. Although one might expect a transfer coefficient of 0.5, the value of 0.26 is not surprising and indicates that the reaction is not an elementary one. The is lower than the 7.5 mA/cm^2 estimated by Fedkiw and Watts[1] (for a solution of 0.55 M Fe^{2+}, 0.55 M Fe^{3+}). However, it does agree with experimen-

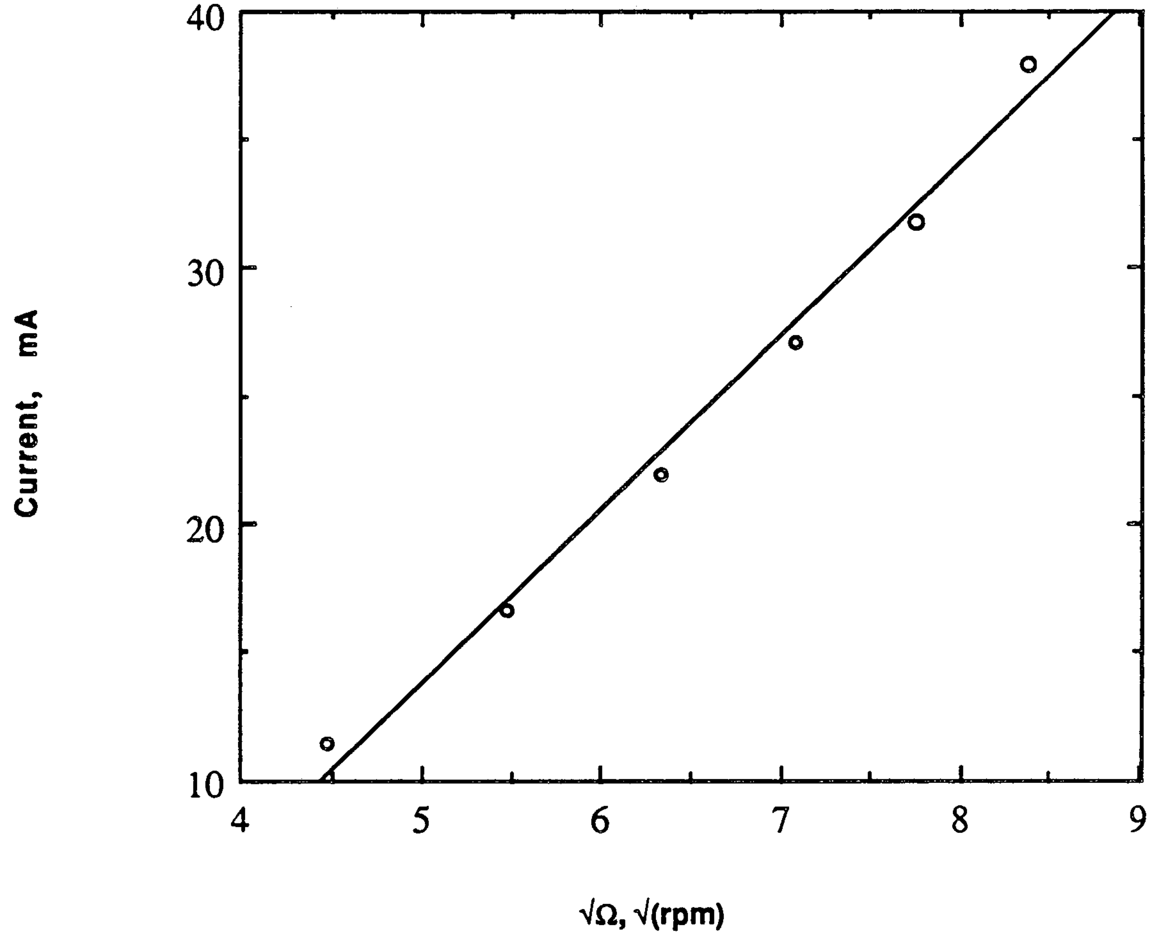


Figure 7-1. Limiting currents for different rotation speeds plotted *versus* the square root of rotation speed. Used to determine the diffusion coefficient for the ferric-containg species in aqueous HCl.

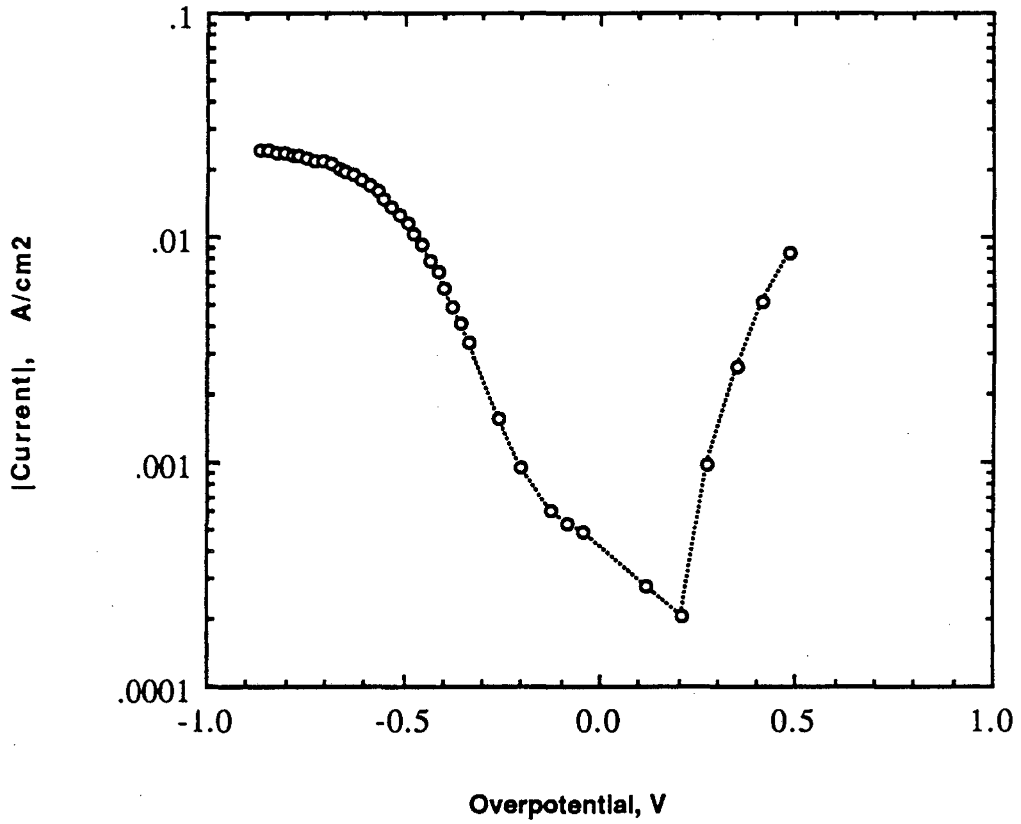


Figure 7-2. A Tafel plot. Used to determine kinetic parameters for the reduction of ferric ion in aqueous HCl.

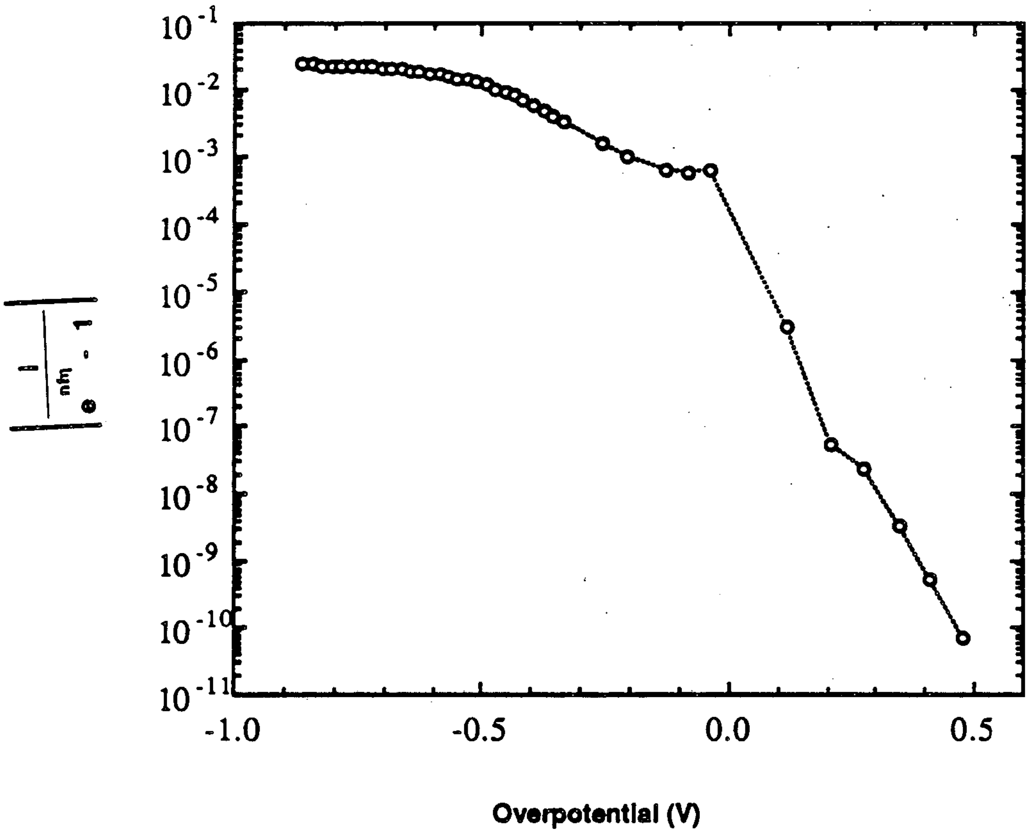


Figure 7-3. An Allen-Hickling plot. Used to determine kinetic parameters for the reduction of ferric ion in aqueous HCl.

tal work on the Fe^{3+}/Fe^{3+} electrode summarized by Vetter.[2] Accounting for the concentration of ferric ion, our value agrees with that obtained by Gerischer[3] in H_2SO_4 and by Randles and Somerton[4] in $HClO_4$ on platinum electrodes. Gerischer and others[5, 6] determined values of the transfer coefficient near 0.5. This with Gerischer's determination that the electron-transfer reaction is the same as the overall reaction indicate that, at least in sulfuric acid, the reaction is an elementary one. However, on a different surface and especially in a different solution with the possibility of complexation of the reactant, the mechanism may be different. In chloride solution, the ferric and ferrous ions are likely to by complexed with chloride. The difference in speciation may result in a difference in mechanism and kinetic parameters.

References

1. Peter S. Fedkiw and Rick W. Watts, "A Mathematical Model for the Iron/Chromium Redox Battery," *J. Electrochem. Soc.*, **131**, 704 (1984).

2. Klaus J. Vetter, *Electrochemical Kinetics: Theoretical and Experimental Aspects*, pp. 455-457, Academic Press, New York (1967).

3. H. Gerischer, *Z. Elektrochem.*, **54**, 366 (1950).

4. J. E. B. Randles and K. W. Somerton, *Trans. Faraday Soc.*, **48**, 937 (1952).

5. E. Lewartowicz, *J. Chim. Phys.*, **49**, 573 (1952).

6. J. V. Petrocelli and A. A. Paolucci, *J. Electrochem. Soc.*, **98**, 291 (1950).

CHAPTER 8

Flow-Through Porous Electrode Model

This chapter presents a model of the frequency response of flow-through porous electrodes. The model is based on a first-principles approach—not on equivalent circuits. To use this transient model, one must have the results of a steady-state model. This model is presented first.

8.1. Steady-state model

Past work on modeling porous flow-through electrodes was summarized in Chapter 5. Here we present Trainham and Newman's derivation of the governing equations for this system. The derivations are based on Newman and Tiedemann's review. For the details the reader can refer to the original work.[1-3]

The model is that of the reduction of a soluble reactant to a soluble product in excess supporting electrolyte. There may be a side reaction characterized by its rate at the half-wave potential of the main reaction. The products of the side reaction are also dissolved. The bed is one-dimensional, of length L, porosity ε, specific area a, and conductivity σ. These properties do not change with time or position. The effective solution conductivity is κ (corrected for void fraction). There is a superficial fluid velocity v and an average mass transfer coefficient k_m.

The governing equations are material and charge balances. The material balance for the reactant is

$$\frac{dN_R}{dx} = aj_{Rn} \ , \tag{8-1}$$

where the superficial flux of the reactant in the direction of fluid flow is

$$N_R = -\varepsilon(D_R + D_a)\frac{dc_R}{dx} + c_R v \qquad (8\text{-}2)$$

and the local flux from the pore wall to the flowing solution is given by

$$j_{Rn} = k_m(c_{Rw} - c_R) \ . \qquad (8\text{-}3)$$

The concentrations c_{R_w} and c_R are wall and pore concentrations of the reactant averaged over the volume of the pores. A correlation for the average mass transfer coefficient k_m (Sh *vs.* Pe) was determined by Matlosz[4] for his experimental 9 system. D_R is the effective diffusion corrected for tortuosity and D_a is the axial dispersion coefficient

$$D_a = \frac{3v}{a\varepsilon}(1 - \varepsilon) \qquad (8\text{-}4)$$

The transport of electrons in the matrix and current in the solution are governed by Ohm's law:

$$i_1 = -\sigma\frac{d\Phi_1}{dx} \quad \text{and} \qquad (8\text{-}5)$$

$$i_2 = -\kappa\frac{d\Phi_2}{dx} \ . \qquad (8\text{-}6)$$

Because charge is conserved between phases

$$\frac{di_1}{dx} + \frac{di_2}{dx} = 0 \ . \qquad (8\text{-}7)$$

The transfer current is the sum of the currents from the individual reactions:

$$\frac{di_2}{dx} = -\frac{anF}{s_R}j_{Rn} + ai_{nS} \ . \qquad (8\text{-}8)$$

Generally the average transfer current density from any reaction may be given in Butler-Volmer form:

$$i_{nR} = i_{oR,ref}\left\{\frac{c_{Pw}}{c_{Pf}}\exp\left[\frac{\alpha_{aR}F}{RT}(\Phi_1 - \Phi_2)\right] - \frac{c_{Rw}}{C_{Rf}}\exp\left[-\frac{\alpha_{cR}F}{RT}(\Phi_1 - \Phi_2)\right]\right\} \qquad (8\text{-}9)$$

and

$$i_{nS} = i_{oS,ref}\left\{\exp\left[\frac{\alpha_{aS}F}{RT}(\Phi_1 - \Phi_2 - \Delta U)\right] - \exp\left[-\frac{\alpha_{cS}F}{RT}(\Phi_1 - \Phi_2 - \Delta U)\right]\right\} \qquad (8\text{-}10)$$

where ΔU is the difference in open-circuit potentials between the side and main reactions ($\Delta U = U_S - U_R$).

The local overpotential η is Φ_1 - Φ_2. Therefore its derivative is

$$\frac{d\eta}{dx} = \frac{d(\Phi_1 - \Phi_2)}{dx} = -\frac{i_1}{\sigma} + \frac{i_2}{\kappa} . \tag{8-11}$$

It will be convenient to eliminate the concentrations of reactants and products near the wall. We can replace c_{P_w} with c_R and c_{R_w} because of the stoichiometry. We can eliminate c_{R_w} from the expression for j_{Rn} using the Butler-Volmer equation for i_{nR} and substitute this and the equation for N_R into the material balance to arrive at

$$v\frac{dc_R}{dx} = \\ \varepsilon(D_R + D_a)\frac{d^2c_R}{dx^2} - a\frac{\dfrac{c_R}{c_{Rf}} - \left(1 + \dfrac{c_{Rf} - c_R}{c_{Pf}}\right)\exp\left[\dfrac{(\alpha_{aR} + \alpha_{cR})F\eta}{RT}\right]}{\dfrac{1}{c_{Rf}k_m} - \dfrac{nF}{s_R i_{oR,ref}}\exp\left(\dfrac{\alpha_{cR}F}{RT}\eta\right) + \dfrac{1}{k_{mP}c_{Pf}}\exp\left[\dfrac{(\alpha_{aR} + \alpha_{cR})F\eta}{RT}\right]} . \tag{8-12}$$

The potential distribution is obtained by eliminating i_1, i_2, and i_{nS} and by differentiating the equation for the derivative of η:

$$\frac{d^2\eta}{dx^2} = \left(\frac{1}{\kappa} + \frac{1}{\sigma}\right)a \times \\ \left\{i_{oS,ref}\left[\exp\left(\frac{\alpha_{aS}F}{RT}(\eta - \Delta U)\right) - \exp\left(-\frac{\alpha_{cS}F}{RT}(\eta - \Delta U)\right)\right]\right. \\ \left. - \left(\frac{nF}{s_R}\right)\frac{\dfrac{c_R}{c_{Rf}} - \left(1 + \dfrac{c_{Rf} - c_R}{c_{Pf}}\right)\exp\left[\dfrac{(\alpha_{aR} + \alpha_{cR})F\eta}{RT}\right]}{\dfrac{1}{c_{Rf}k_m} - \dfrac{nF}{s_R i_{oR,ref}}\exp\left(\dfrac{\alpha_{cR}F}{RT}\eta\right) + \dfrac{1}{k_{mP}c_{Pf}}\exp\left[\dfrac{(\alpha_{aR} + \alpha_{cR})F\eta}{RT}\right]}\right\} . \tag{8-13}$$

These equations may be expressed in dimensionless form. The material balance is

$$\frac{d\theta}{dy} = D'\frac{d^2\theta}{dy^2} - \frac{\theta - \left(1 + \dfrac{1-\theta}{\theta_{Pf}}\right)P_1\exp\left[\left(\dfrac{\alpha_{aR}}{\alpha_{cR}} + 1\right)\eta'\right]}{1 + \exp(\eta') + \dfrac{P_1P_7}{\theta_{Pf}}\exp\left[\left(\dfrac{\alpha_{aR}}{\alpha_{cR}} + 1\right)\eta'\right]} ; \tag{8-14}$$

the potential distribution,

$$\frac{d^2\eta'}{dy^2} =$$

$$P_2\left[P_3\exp\left(-\frac{\alpha_{cS}\eta'}{\alpha_{cR}}\right)\left[1 - P_4\exp\left(\frac{\alpha_{aS}+\alpha_{cS}}{\alpha_{cR}}\eta'\right)\right] + \frac{\theta - \left(1+\frac{1-\theta}{\theta_{Pf}}\right)P_1\exp\left[\left(\frac{\alpha_{aR}}{\alpha_{cR}}+1\right)\eta'\right]}{1+\exp(\eta') + \frac{P_1P_7}{\theta_{Pf}}\exp\left[\left(\frac{\alpha_{aR}}{\alpha_{cR}}+1\right)\eta'\right]}\right] . \qquad (8\text{-}15)$$

The variables were made dimensionless and the dimensionless parameters carefully defined to emphasize the relevant phenomena. The concentration was made dimensionless with respect to the feed concentration:

$$\theta = \frac{c_R}{c_{Rf}} ; \qquad (8\text{-}16)$$

the distance, with respect to the penetration depth:

$$y = \alpha x = \frac{ak_m}{v}\,x . \qquad (8\text{-}17)$$

The quantity D′ combines the effects of axial diffusion and dispersion:

$$D' = \frac{(D_R + D_a)ak_m\varepsilon}{v^2} . \qquad (8\text{-}18)$$

The overpotential was made dimensionless using:

$$\exp(\eta') = -\frac{nFk_mc_{Rf}}{s_Ri_{oR,ref}}\exp\left(\frac{\alpha_{cR}F\eta}{RT}\right) . \qquad (8\text{-}19)$$

The backward term of the main reaction is characterized by

$$P_1 = \left(-\frac{s_Ri_{oR,ref}}{nFk_mc_{Rf}}\right)^{1+\alpha_{aR}/\alpha_{cR}} . \qquad (8\text{-}20)$$

The importance of the ohmic potential drop is characterized by

$$P_2 = \frac{\alpha_{cR}nF^2v^2c_{Rf}}{s_Rak_mRT}\left(\frac{1}{\kappa}+\frac{1}{\sigma}\right) . \qquad (8\text{-}21)$$

The side reaction is characterized by

$$P_3 = -\frac{s_Ri_{oS,ref}}{nFk_mc_{Rf}}\exp\left(\frac{\alpha_{cS}F\Delta U}{RT}\right)\left(-\frac{nFk_mc_{Rf}}{s_Ri_{oR,ref}}\right)^{\alpha_{cS}/\alpha_{cR}} . \qquad (8\text{-}22)$$

The backward term of the side reaction is characterized by

$$P_4 = \left(- \frac{s_R i_{oR,ref}}{nFk_m c_{Rf}} \right)^{\frac{\alpha_{aR} + \alpha_{cR}}{\alpha_{aR}}} \exp\left[- \frac{F(\alpha_{aS} + \alpha_{cS})\Delta U}{RT} \right] . \tag{8-23}$$

P_5 and P_6 characterized the ohmic potential drops in the solution and matrix phases and are related to P_2 as follows:

$$P_5 = - \frac{\sigma P_2}{\sigma + \kappa} , \tag{8-24}$$

$$P_6 = - \frac{\kappa P_2}{\sigma + \kappa} , \tag{8-25}$$

thus

$$-P_2 = P_5 + P_6 . \tag{8-26}$$

The final two dimensionless terms are

$$\theta_{Pf} = \frac{C_{Pf}}{C_{Rf}} \tag{8-27}$$

and

$$P_7 = \frac{k_{mR}}{k_{mP}} . \tag{8-28}$$

The current density was made dimensionless with respect to the current density that would be produced if all the reactant in the feed were consumed:

$$I^* = \frac{s_R i}{nFvc_{Rf}} . \tag{8-29}$$

This current density is expressed in units of amps per cm^2 of electrode cross-sectional (not surface) area.

The concentration boundary conditions are the Danckwerts,[5] Wehner-Wilhelm[6] conditions at the inlet

$$c_{Rf} v = c_R v - \varepsilon(D_R + D_a)\frac{dc_R}{dx} \text{ at } x = 0 \tag{8-30}$$

and at the exit

$$\frac{dc_R}{dx} = 0 \text{ at } x = L . \tag{8-31}$$

The boundary conditions for the current and potential depend on the electrode configuration.

These are listed for the potential in Table 8-1.[7,8]

These equations are solved by a program called *steady.pas* to give the steady-state solution that is needed to solve the transient problem.

8.2. Frequency response

To derive the governing equations for the frequency response of the flow-through porous electrode, we start with the usual governing equations and make the substitutions described in section 6.5.2 as was done by Tribollet and Newman for the rotating-disk electrode.[9] To the steady-state material and charge balance equations we add accumulation and doule-layer charging terms, respectively. We take these equations and separate the concentration, potential, and current terms into steady and transient parts. For example, the concentration variable becomes

$$\theta = \bar{\theta} + \tilde{\theta}e^{j\omega t} \ . \qquad (8\text{-}32)$$

Many terms in the resulting equations can be eliminated by subtracting the steady-state equations.

The starting equations are the material balance:

Configuration	$\left.\frac{d\eta'}{dy}\right\vert_{y=0}$	$\left.\frac{d\eta'}{dy}\right\vert_{y=\alpha L}$
UD	P_5I^*	$-P_6I^*$
DU	P_6I^*	$-P_5I^*$
UU	$-P_2I^*$	0
DD	0	P_2I^*

Table 8-1. Boundary conditions on the potential for various electrode configurations. The first letter of the two-letter designation refers to the counterelectrode placement; the second, the current collector: U = upstream, D = downstream.

$$-\frac{\partial\theta}{\partial\tau}+D'\frac{\partial^2\theta}{\partial y^2}-\frac{\partial\theta}{\partial y}-\frac{\theta-\left[1+\frac{1-\theta}{\theta_{Pf}}\right]P_1e^{\beta\eta'}}{1+e^{\eta'}+\frac{P_1P_7}{\theta_{Pf}}e^{\beta\eta'}}=0\ . \tag{8-33}$$

The charge balance is the same as for the steady state except for the addition of a term that accounts for charging the double layer:[†]

$$\begin{aligned}\frac{\partial^2\eta'}{\partial y^2}&=\\ &P_2\left[P_3\exp\left(-\frac{\alpha_{cS}\eta'}{\alpha_{cR}}\right)\left[1-P_4\exp\left(\frac{\alpha_{aS}+\alpha_{cS}}{\alpha_{cR}}\eta'\right)\right]+\frac{\theta-\left[1+\frac{1-\theta}{\theta_{Pf}}\right]P_1\exp\left[\left(\frac{\alpha_{aR}}{\alpha_{cR}}+1\right)\eta'\right]}{1+\exp(\eta')+\frac{P_1P_7}{\theta_{Pf}}\exp\left[\left(\frac{\alpha_{aR}}{\alpha_{cR}}+1\right)\eta'\right]}\right]\\ &+P_2C_{dl}\frac{\partial\eta'}{\partial\tau}\ .\end{aligned} \tag{8-34}$$

After making the substitutions and simplifying, we end with the following:

$$\begin{aligned}&\Delta^2\left[\tilde\theta j\Omega-D'\frac{d^2\tilde\theta}{dy^2}+\frac{d\tilde\theta}{dy}\right]\\ &+\bar\eta'\left[\left[e^{\bar\eta'}+\frac{P_1^2P_7}{\theta_{Pf}}\beta e^{\beta\bar\eta'}\right]\left[P_1F_1e^{\beta\bar\eta'}-\bar\theta\right]\right]\\ &+\bar\theta\Delta\left[1+\frac{P_1}{\theta_{Pf}}e^{\beta\bar\eta'}\right]=0\end{aligned} \tag{8-35}$$

and

[†] The steady-state case included no double-layer charging current. To determine the contribution of this current to the governing equations we considered an electrode with no faradaic current. In this case the potential distribution is given by

$$\frac{\partial^2\eta}{\partial x^2}=\left[\frac{1}{\sigma}+\frac{1}{\kappa}\right]aC\frac{\partial\eta}{\partial t}\ .$$

The right-hand side of this equation can be non-dimensionalized and added to the previously derived governing equation.

$$\Delta^2\left[\frac{d^2\bar{\eta}}{dy^2} - P_2C_{dl}j\Omega\bar{\eta}'\right]$$
$$P_2\bar{\eta}'\left\{(F_1P_1e^{\beta\bar{\eta}'} - \bar{\theta})(e^{\bar{\eta}'} + g\beta e^{\beta\bar{\eta}'}) - \Delta^2P_3e^{-\frac{\alpha_{cS}}{\alpha_{cR}}\bar{\eta}'}\left[\frac{\alpha_{cS}}{\alpha_{cR}} + \frac{\alpha_{aS}}{\alpha_{cR}}P_4e^{-\frac{\alpha_{aS}+\alpha_{cS}}{\alpha_{cR}}\bar{\eta}'}\right] - \Delta P_1F_1\beta e^{\beta\bar{\eta}'}\right\} \tag{8-36}$$
$$- \Delta P_2\tilde{\theta}\left[1 + \frac{P_1}{\theta_{Pf}}e^{\beta\bar{\eta}'}\right] = 0 \ ,$$

where Δ is given by:

$$\Delta = 1 + e^{\eta'} + \frac{P_1P_7}{\theta_{Pf}}e^{\beta\eta'} \tag{8-37}$$

and β by:

$$\beta = 1 + \frac{\alpha_{aR}}{\alpha_{cR}} \ . \tag{8-38}$$

Each quantity with a tilde has a real and an imaginary part. Therefore these two equations represent four equations for the four unknowns: Re($\tilde{\theta}$), Im($\tilde{\theta}$), Re($\tilde{\eta}'$), and Im($\tilde{\theta}'$). The additional dimensionless terms are

$$\Omega = \frac{\omega}{ak_{mR}\varepsilon} \tag{8-39}$$

and

$$\tau = \frac{ak_{mR}}{\varepsilon}t \ . \tag{8-40}$$

8.3. Porous electrodes without flow

Many useful batteries contain porous electrodes but do not force reactants to flow through the electrode. Therefore it is interesting to include this case as well. The governing equations are essentially the same, but lack the convection terms and are non-dimensionalized in a slightly different way because dividing quantities by a zero velocity will lead to problems. We will use k_{mR} instead of v to non-dimensionalize the various quantities. The resulting dimensionless groups may not have exactly same significance but the governing equations need not be changed much.

If the fluid velocity is zero, there will be no dispersion and D_a will be zero. The equation for the concentration loses its convective term and becomes

$$0 = D'\frac{d^2\theta}{dy^2} - \frac{\theta - \left(1 + \frac{1-\theta}{\theta_{Pf}}\right)P_1\exp\left[\left(\frac{\alpha_{aR}}{\alpha_{cR}} + 1\right)\eta'\right]}{1 + \exp(\eta') + \frac{P_1P_7}{\theta_{Pf}}\exp\left[\left(\frac{\alpha_{aR}}{\alpha_{cR}} + 1\right)\eta'\right]} \; ; \tag{41}$$

For the transient case, the equation for the concentration perturbation becomes:

$$\begin{aligned} &\Delta^2\left(\tilde{\theta}j\Omega - D'\frac{d^2\tilde{\theta}}{dy^2}\right) \\ &+ \tilde{\eta}'\left[\left(e^{\bar{\eta}'} + \frac{P_1^2P_7}{\theta_{Pf}}\beta e^{\beta\bar{\eta}'}\right)\left(P_1F_1e^{\beta\bar{\eta}'} - \bar{\theta}\right)\right] \\ &+ \tilde{\theta}\Delta\left(1 + \frac{P_1}{\theta_{Pf}}e^{\beta\bar{\eta}'}\right) = 0 \end{aligned} \tag{42}$$

The definitions of several dimensionless groups will change as follows:

$$P_2 = \frac{\alpha_{cR}nF^2k_{mR}c_{Rf}}{as_RRT}\left(\frac{1}{\kappa} + \frac{1}{\sigma}\right) \; , \tag{8-43}$$

$$D' = \frac{a\varepsilon D}{k_{mR}} \; , \tag{8-44}$$

$$I^* = \frac{s_Ri}{nFk_{mR}c_{Rf}} \; , \tag{8-45}$$

and

$$\alpha = a \; . \tag{8-46}$$

If the mass transfer coefficient is a flux divided by a concentration difference, instead of using a correlation for k_{mR} we can use

$$k_{mR} = \frac{D_R}{r_{pore}} \; . \tag{8-47}$$

For a porous electrode without electrolyte flow, the concentration boundary conditions are different than for the electrode with flow. If the end of the electrode is near a large reservoir of electrolyte, the boundary condition is

$$\bar{\theta} = 1; \;\; \tilde{\theta} = 0 \; , \tag{48}$$

if it is a physical barrier to flow (like a current collector) the boundary condition is

$$\frac{d\bar{\theta}}{dy} = 0; \quad \frac{d\tilde{\theta}}{dy} = 0 \ . \tag{49}$$

8.4. Calculation of the impedance

So far we have shown how the potential and concentration distributions are calculated. From the potential at one end of the electrode and the applied current, we can calculate the quantity of interest—the measured impedance of the system.

The time-varying potential and currents are

$$v = \mathrm{Re}(\tilde{v}e^{j\omega t}) \tag{50}$$

and

$$i = \mathrm{Re}(\tilde{i}e^{j\omega t}) \ , \tag{51}$$

respectively.

The impedance is the ratio of $\tilde{v}$ to $\tilde{i}$:

$$Z = \frac{\tilde{v}}{\tilde{i}} = \frac{\mathrm{Re}(\tilde{v}) + j\mathrm{Im}(\tilde{v})}{\mathrm{Re}(\tilde{i}) + j\mathrm{Im}(\tilde{i})} \ . \tag{52}$$

Multiplying the numerator and denominator by the complex conjugate of the denominator yields:

$$\frac{\mathrm{Re}(\tilde{v})\mathrm{Re}(\tilde{i}) + \mathrm{Im}(\tilde{v})\mathrm{Im}(\tilde{i}) + j[\mathrm{Im}(\tilde{v})\mathrm{Re}(\tilde{i}) - \mathrm{Re}(\tilde{v})\mathrm{Im}(\tilde{i})]}{[\mathrm{Re}(\tilde{i})]^2 + [\mathrm{Im}(\tilde{i})]^2} \ . \tag{53}$$

If $i = \Delta\sin(\omega t)$, then $\mathrm{Re}(\tilde{i}) = 0$ and $\mathrm{Im}(\tilde{i}) = -\Delta i$. Thus

$$\mathrm{Re}(Z) = -\frac{\mathrm{Im}(\tilde{v})}{\Delta i} \tag{54}$$

and

$$\mathrm{Im}(Z) = \frac{\mathrm{Re}(\tilde{v})}{\Delta i} \ . \tag{55}$$

Since the program calculates η' as a function of position, it can pick the value of eta at the appropriate end of the electrode (depending on the current collector position), convert it to volts, divide it by Δi, and change the sign if appropriate. The impedance is calculated in units of $\Omega\text{-cm}^2$.

8.5. Description of programs

Parts of the programs used to solve the equations are in separate modules that are compiled separately and linked together. The governing equations and boundary conditions for the steady-state and frequency response problems are contained in the programs *Steady* and *Impedance*, respectively. The declaration statements for these programs are in the modules *SteadyDcl* and *ImpedanceDcl*. These programs read their data files *steady.dat* and *imped.dat* and write to their output files *steady.out* and *imped.out* using the modules *SteadyIO* and *ImpedanceIO*. *SteadyIO* also creates the file *barprofiles.dat* (read by *Impedanceio*) that contains the steady-state concentration and potential profiles ($\bar{\theta}(x)$ and $\bar{\eta}'(x)$) needed by *Impedance*. *Impedance* contains a loop to solve the governing equations for different values of the applied frequency at some value of the applied current. *Aidmod* contains global declarations and includes the procedures *BandAid* and *BandShell* as well as useful input and output routines. Program listings are found in appendices following this chapter.

The governing equations are simultaneous, one-dimensional, ordinary, partial differential equations. They are solved by a finite-difference method using Matlosz's *BandShell* procedure. A listing of this procedure is not included here, but can be found elsewhere.[10] A more complete description of *BandAid* and *BandShell* is found in References 10 and 11. *BandAid* sets parameters (*e.g.*, the type of difference approximations and whether image points are used) *to be passed to BandShell*. (These parameters that relate to the numerical method, are set by *Impedance* and *Steady* so the program *BandAid* is not needed.) *BandShell* is a finite-difference program that solves simultaneous, ordinary, differential equations by using Newman's BAND routine to set up and invert (using Newman's MATINV routine) tridiagonal matrices.[12-15]

8.6. Possible source of difficulty

When solving second order, ordinary differential equations by finite difference methods, one must be sure that the second order terms do not disappear. Consider the equation

$$D'\theta'' - \theta' = f(\eta,\theta) \ . \tag{8-56}$$

If D′ is small compared to the mesh size, the second order term will be negligible with respect to the first, and the finite-difference program will solve the equation with different boundary conditions for the odd and even mesh points and will converge on different solutions for the odd and even points. This can be avoided by making sure that the mesh spacing h is not much greater than the coefficient D′.

References

1. John Newman and William Tiedemann, "Porous Electrode Theory with Battery Applications," *AIChE J.*, **21**, 25 (1975).

2. James A. Trainham and John Newman, "A Flow-Through Porous Electrode Model: Application to Metal-Ion Removal from Dilute Streams," *J. Electrochem. Soc.*, **124**, 1527 (1977).

3. James Arthur Trainham, *Flow-Through Porous Electrodes*, Ph. D. Dissertation, pp. 35-47 and 132-138, University of California, Berkeley (1979)

4. Michael John Matlosz, *Experimental Methods and Software Tools for the Analysis of* Electrochemical Systems, pp. 31-33. Ph.D. dissertation, University of California, Berkeley (1985) LBL-19375.

5. P. V. Danckwerts, "Continuous Flow Systems: Distribution of Residence Times," *Chem. Eng. Sci.*, **2**, 1 (1953).

6. J. F. Wehner and R. H. Wilhelm, "Boundary conditions of a flow reactor," *Chem. Eng. Sci.*, **6**, 89 (1956).

7. James Trainham and John Newman, "The Effect of Electrode Placement and Finite Matrix Conductivity on the Performance of Flow-Through Porous Electrodes," *J. Electrochem. Soc.*, **125**, 58 (1978).

8. Trainham, *op. cit.*, pp. 75-81.

9. Bernard Tribollet and John Newman, "Impedance Model for a Concentrated Solution: Application to the Electrodissolution of Copper in Chloride Solutions," *J. Electrochem. Soc.*, **131**, 2780 (1984).

10. Matlosz, *op. cit.*

11. M. Matlosz and J. Newman, "Solving 1-D Boundary-Value Problems with *BandAid*: A Functional Programming Style and a Complementary Software Tool," *Comput. Chem. Eng.*, **11**, 45 (1987).

12. John Newman, *Numerical Solution of Coupled, Ordinary Differential Equations*, Lawrence Radiation Laboratory, University of California, Berkeley, August, 1967, UCRL-17739.

13. John Newman, "Numerical Solution of Coupled, Ordinary Differential Equations," *Ind. Eng. Chem. Fund.*, **7**, 514 (1968).

14. John Newman, *Electrochemical Systems*, Appendix C, Prentice-Hall, Englewood Cliffs, NJ (1973).

14. Ralph White, Charles M. Mohr, Peter Fedkiw, and John Newman, *The Fluid Motion Generated by a Rotating Disk: A Comparison of Solution Techniques*, LBL-3910, Lawrence Berkeley Laboratory, November, 1975.

APPENDIX L

Listing of Program *Steady*

```
[INHERIT('AidMod.pen','SteadyDcl.pen','SteadyIO.pen')]

{...  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

          Program Title:  Steady

          Written By:      Paul Shain

          Date written:   December 15, 1983

          Date Modified:  May 21, 1990

          Purpose:   This program calculates the steady-state concentration
          and potential distribution in a flow-through porous electrode
          according to the model of Trainham and Newman ( ELECTROCHIMICA
          ACTA, 26, p. 455 (1981) ).

          Variable Number 1 == dimensionless concentration
          Variable Number 2 == dimensionless potential

          Parameters are those for reduction of a reactant in a redox
          system.  A side reaction and axial diffusion and dispersion are
          included.


      ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++   ...}


program Steady( input, output );

{...The constant, type, and variable declarations below are actually
      declared in a another file, 'SteadyDcl.pas'.  This program was
      written in separate pieces, or modules, to remove from view parts,
      like input and output routines, which do not contribute to an
      understanding of how the program works.  The following list details
      the pascal files that make up the program and their contents.

      AidMod      -   Contains const, type, and var declarations required by
                      BandShell, and the BandShell procedure.  Is inherited
                      by all the other files.

      SteadyDcl -   Contains all declarations needed by the program Steady.
                    Is inherited by SteadyIO and Steady.

      SteadyIO  -   Contains I/O routines used by Steady.  Is inherited by
                    Steady.

      Steady      -   Contains the equations to be solved by BandShell and
                    their boundary conditions.

Inheriting is a nonstandard feature available in VAX Pascal.  A
"module" containing declarations and procedures can be compiled to
produce an object file of filetype "obj" and an environment file of
filetype "pen".  Other files can "inherit" environments in order to
use the declarations or procedures contained therein.  After a program
is compiled, its object file is linked to the object files of the
environments it has inherited to produce an executable file.  There
should be similar procedures on other machines.  It is possible to
combine the separate modules into one program.
```

```
const          (... BandAid Parameters ...)

        nEqns = 2;
        ImageFirstPoint = true;
        ImageLastPoint = true;
        FactorIncrement = 1e-6;
        AbsoluteIncrement = 1e-6;
        ReduceTimeOption = true;

            (... Physical Constants ...)

        F = 96487.0;   (Faraday's constant, coulombs/equivalent)
        R = 8.314;   (ideal gas constant, joules/mole-K)


type   DateType = packed array[1..11]of char;


var          (... BandAid Parameters ...)

    Guess, FinalResult, Deviation, Residual : ValueArray;
    j, jMax, ItMax : integer;
    Tolerance : RealNumber;

            (... Physical Parameters ...)

    v, e, CRf, CPf, AlphaaR, AlphacR, AlphaaS, AlphacS : RealNumber;
    L, a, D, Da, kmR, rPore, sR, ioRref, ioSRef, Theta, kmP : RealNumber;
    i, kappa, sigma, DeltaU, T : RealNumber;
    n : integer;

            (... Dimensionless Parameters ...)

    P1, P2, P3, P4, P5, P6, P7 : RealNumber;
    IStar, DPrime, AlphaL, ThetaPf : RealNumber;
    yMin, yMax : RealNumber;

            (... Other Parameters ...)

    Configuration : (UD, DU, UU, DD);
    CounterElectrode, CurrentCollector : char;
    ThetaGuess, EtaGuess : RealNumber;

    BarProfiles : text;
    pFactors : text;

    MeshSize : RealNumber;
    CPUTimeUsed, ClockInitial : integer;
    NumberOfIterations : integer;
    BeginDate, BeginTime : DateType;
                                          ...}


function Equation( i, j : integer;
            y, h : RealNumber;
            var NewResult : ValueArray;
             function cInterp( k : integer;
                      y : RealNumber;
                      var Result : ValueArray) : RealNumber;
             function dnFdyn( n : integer;
                     function F( y : RealNumber ) : RealNumber;
```

```
                      y : RealNumber;
                      Approx : DiffApprox ) : RealNumber
                             ) : RealNumber;


   function Var1 ( y : RealNumber ) : RealNumber;
   begin Var1 := cInterp(1,y,NewResult) end;


   function Var2 ( y : RealNumber ) : RealNumber;
   begin Var2 := cInterp(2,y,NewResult) end;

   function Theta ( y : RealNumber ) : RealNumber;
   begin    Theta := Var1(y)          end;

   function Eta ( y : RealNumber ) : RealNumber;
   begin    Eta := Var2(y)            end;


   function dThetady ( y : RealNumber ) : RealNumber;
   begin
            dThetady := dnFdyn(1,Var1,y,Cendiff)
   end;

   function dEtady ( y : RealNumber ) : RealNumber;
   begin
              dEtady := dnFdyn(1,Var2,y,CenDiff)
   end;

   function d2Thetady2( y : RealNumber ) : RealNumber;
   begin
            d2Thetady2 := dnFdyn(2,Var1,y,Cendiff)
   end;

   function d2Etady2 ( y : RealNumber ) : RealNumber;
   begin
            d2Etady2 := dnFdyn(2,Var2,y,CenDiff)
   end;


function UpStrmBC( i : integer; y : RealNumber ) : RealNumber;

begin
   case i of
      1: if (v = 0) then
            UpStrmBC := Theta(y) - 1    { = 0 }
         else
            UpStrmBC := DPrime*dThetady(y) - Theta(y) + 1;    { = 0 }

      2: case Configuration of

         UD:        UpStrmBC := dEtady(y) - P5*IStar;  { = 0 }
         DU:        UpStrmBC := dEtady(y) - P6*IStar;  { = 0 }
         UU:        UpStrmBC := dEtady(y) + P2*IStar;  { = 0 }
         DD:        UpStrmBC := dEtady(y)    { = 0 }

      end    { Configuration cases }

end    { Equation cases }
```

```
end;  { UpStrmBC }




function DiffEQ( i : integer; y : RealNumber ) : RealNumber;

var     DispersionTerm, Numerator, Denominator,
        MainReactionTerm, SideReactionTerm : RealNumber;
        Factor1, Factor2, Alpha1, Alpha2 : RealNumber;

begin   { body of DiffEQ }

        Factor1 := 1 + (1 - Theta(y)) / ThetaPf;
        Factor2 := 1 + (AlphaaR / AlphacR);
        Alpha1 := -AlphacS / AlphacR;
        Alpha2 := (AlphaaS + AlphacS) / AlphacR;

        Numerator :=  Theta(y) - Factor1 * P1 * exp(Factor2*Eta(y) );
        Denominator := 1 + exp( Eta(y) )
                          + (P1 * P7 / ThetaPf) * exp( Factor2*Eta(y) ) ;

        MainReactionTerm := Numerator/Denominator;

        SideReactionTerm := exp( Alpha1 * Eta(y) ) *
            ( P3 - P3 * P4 * exp( Alpha2 * Eta(y)) );

        DispersionTerm := DPrime * d2Thetady2(y);


        case i of

        1:      If (v = 0) then
                    DiffEQ := DispersionTerm - MainReactionTerm    { = 0 }
                else
                    DiffEQ := DispersionTerm - MainReactionTerm
                              -dThetady(y); { = 0 }

        2:      DiffEQ := P2 * ( SideReactionTerm
                              + MainReactionTerm ) - d2Etady2(y)    { = 0 }

        end  { i cases }

end;  { DiffEQ }


function DnStrmBC( i : integer; y : RealNumber ) : RealNumber;

begin

        case i of

   1:   DnStrmBC := Theta(y) - 1;  { = 0 }


   2:       case Configuration of
```

```
      UD:      DnStrmBC := dEtady(y) + P6*Istar;  { = 0 }
      DU:      DnStrmBC := dEtady(y) + P5*IStar;  { = 0 }
                 UU:      DnStrmBC := dEtady(y);  { = 0 }
      DD:      DnStrmBC := dEtady(y) - P2*Istar   { = 0 }

      end  { Configuration cases }

        end  { i cases }

end;  { DnStrmBC }


begin  { body of Equation }

   If ( j = 1 ) then

                 case i of

                 1:      Equation := UpStrmBC(1,yMin);
                 2:      Equation := UpStrmBC(2,yMin)

                 end  { i cases }

   else If ( j = jMax ) then

                 case i of

                 1:      Equation := DnStrmBC(1,yMax);
                 2:      Equation := DnStrmBC(2,yMax)

                 end   { i cases }

        else    case i of

                 1:      Equation := DiffEQ(1,y);
                 2:      Equation := DiffEQ(2,y)

                 end  { i cases }

end;  { Equation }


function Converged( function y( j : integer ) : RealNumber;
          h : RealNumber;
          var NewResult, Deviation : ValueArray;
          var Residual : ValueArray;
                    function cInterp( k : integer;
                     y : RealNumber;
                    var Result : ValueArray ) : RealNumber;
               function dnFdyn(  n : integer;
                       function F( x : RealNumber ) : RealNumber;
                  y : RealNumber;
                  Approx : DiffApprox ) : RealNumber
```

```
                                    ) : boolean;


var      k, j : integer;

begin
         Converged := true;


         for k := 1 to nEqns do
             for j := 1 to jMax do
                 if ( abs(Deviation[k,j] /NewResult[k,j]) > Tolerance ) then
                                                         Converged := false

end;  { Converged }



procedure NonBandCalcs(    LastIteration : boolean;
          Iteration, CPUTime : integer;
          function y( j : integer ) : RealNumber;
          h : RealNumber;
          var NewResult, Deviation : ValueArray;
          var Residual : ValueArray;
          function cInterp( k : integer;
                   y : RealNumber;
                  var Result : ValueArray ) : RealNumber;
          function dnFdyn( n : integer;
                              function F( y : RealNumber ) : RealNumber;
                  y : RealNumber;
                  Approx : DiffApprox ) : RealNumber );



var    j : integer;

begin     { body of NonBandCalcs }

if ( LastIteration ) then
begin
   NumberOfIterations := Iteration;   { rename the parameters passed from }
   MeshSize := h;                       { BandAid for use in the i/o routines }
   CPUTimeUsed := CPUTime;

{*}PrintParameters;
   RunTimeDiagnostics;
   PrintProfiles( y, NewResult );     {*}
   PrintBarProfiles( NewResult );
{*}PrintTimeSummary                    {*}
end

end;    { NonBandCalcs }


procedure SetParameters;


procedure Error;
```

```
begin
   writeln; write('Configuration ',CounterElectrode,CurrentCollector,
         ' is not UD, DU, UU, or DD. ');
   writeln; write('Check the data file and try again. ');
   writeln; writeln;
   halt

end;   { Error }


begin   { body of SetParameters }

Da := 3 * v * (1 - e) / a / e;

if (v = 0) then
   DPrime := e * D * a / kmR
else
   DPrime := e * (D + Da) * a * kmR / (v * v);

if (v = 0) then
   begin
      kmR := D / rPore;
      kmP := kmR
   end;

P1 := ( ( -sR * ioRref ) / ( n * F* kmR * CRf) ) **
                                   ( 1 + ( AlphaaR / AlphacR ) ) ;

if (v = 0) then
   P2 := AlphacR * n * F * F * kmR * CRf / ( sR * a * R * T )
                         * ( 1/kappa + 1/sigma )
else
   P2 := AlphacR * n * sqr(F * v) * CRf / ( sR * a * kmR * R * T )
                         * ( 1/kappa + 1/sigma );

P3 := ( -sR * ioSref / ( n * F * kmR * CRf ) )
      * exp( AlphacS * F / (R*T) * DeltaU )
      * ( -n * F * kmR * CRf / ( sR * ioRref ) ) ** (AlphacS/AlphacR );

P4 := (- sR*ioRref/(n*F*kmR*CRf)) ** ( (AlphaaS+AlphacS)/AlphacR )
                           * exp( -  F*(AlphaaS+AlphacS)*DeltaU/(R*T) );

P5 := -sigma * P2 / ( sigma + kappa );

P6 := kappa / sigma * P5;

P7 := kmR / kmP;

if (v = 0) then
   IStar  := sR * i / ( n * F * kmR * CRf )
else
   IStar  := sR * i / ( n * F * v * CRf );

ThetaPf:= CPf / CRf;

if (v = 0) then
   AlphaL := a * L
else
   AlphaL := a * kmR * L / v;

{*******
reset( pFactors );
readln( pFactors, P1 );
readln( pFactors, P2 );
readln( pFactors, P3 );
readln( pFactors, P4 );
readln( pFactors, P5 );
readln( pFactors, P6 );
readln( pFactors, P7 );
readln( pFactors, thetapf );
readln( pFactors, Dprime );
readln( pFactors, alphaL );
readln( pFactors, istar );
*******}

case CounterElectrode of

        'U':    case CurrentCollector of

            'U':    Configuration := UU;
            'D':    Configuration := UD;
            otherwise    Error

            end;  { CurrentCollector cases }

        'D':    case CurrentCollector of

            'U':    Configuration := DU;
            'D':    Configuration := DD;
            otherwise     Error

            end;  { CurrentCollector cases }

        otherwise    Error

    end;  { CounterCollector cases }


yMin := 0;
yMax := AlphaL;

end;  { SetParameters }



begin       { body of Steady }

    ClockInitial := SystemClock;
    SetTimeString(BeginTime);
    SetDateString(BeginDate);

    ReadParameters;

    SetParameters;

{*} PrintTitle;

    EchoParameters;    {*}

    for j := 1 to jMax do
    begin
        Guess[1,j] := ThetaGuess;
        Guess[2,j] := EtaGuess
    end;

BandShell( nEqns, jMax, ItMax,
           yMin, yMax, FactorIncrement, AbsoluteIncrement,
           ImageFirstPoint, ImageLastPoint, ReduceTimeOption,
           Guess, FinalResult, Deviation, Residual,
           Equation, Converged, NonBandCalcs )

end.  { Steady }
```

APPENDIX M

Listing of Program *SteadyDcl*

```
[INHERIT('AidMod.pen'), ENVIRONMENT('SteadyDcl.pen')]

module steadydcl( input, output, BarProfiles, pFactors );


const                      {... BandAid Parameters ...}

   nEqns = 2;
   ImageFirstPoint = true;
   ImageLastPoint = true;
   FactorIncrement = 1e-6;
   AbsoluteIncrement = 1e-6;
   ReduceTimeOption = true;

                           {... Physical Constants ...}

   F = 96487.0; { Faraday's constant, coulombs/equivalent }
   R = 8.314;   { ideal gas constant, joules/mole-K }


type    DateType = packed array[1..11]of char;


var                        {... BandAid Parameters ...}

   Guess, FinalResult, Deviation, Residual : ValueArray;
   j, jMax, ItMax : integer;
   Tolerance : RealNumber;

                           {... Physical Parameters ...}

   v, e, CRf, CPf, AlphaaR, AlphacR, AlphaaS, AlphacS : RealNumber;
   L, a, D, Da, kmR, rPore, sR, ioRref, ioSRef, Theta, kmP : RealNumber;
   i, kappa, sigma, DeltaU, T : RealNumber;
   n : integer;

                           {... Dimensionless Parameters ...}

   P1, P2, P3, P4, P5, P6, P7 : RealNumber;
   IStar, DPrime, AlphaL, ThetaPf : RealNumber;
   yMin, yMax : RealNumber;

                           {... Other Parameters ...}

   Configuration : (UD, DU, UU, DD);
   CounterElectrode, CurrentCollector : char;
   ThetaGuess, EtaGuess : RealNumber;

   BarProfiles : text;
           pFactors : text;

   MeshSize : RealNumber;
   CPUTimeUsed, ClockInitial : integer;
   NumberOfIterations : integer;
   BeginDate, BeginTime : DateType;

end.
```

APPENDIX N

Listing of Program *SteadyIO*

```
[INHERIT('AidMod.pen','SteadyDcl.pen'), ENVIRONMENT('SteadyIO.pen')]

module SteadyIO;


procedure SetTimeString( var TimeString : DateType );
begin  time(TimeString)  end;


procedure SetDateString( var DateString : DateType );
begin  date(DateString)  end;


{... +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

        Input /Output Routines:
                The following procedures perform the input and output
                      functions for the BandShell calling program, steady.pas.

                      The following procedures are found in BandShell
                      (or actually in the file IOPkg.pas which is included
                      in AidMod.pas):

                      LF(n)       prints n blank lines (line feed)
                      RI(n)       reads an integer
                      RR(n)       reads a real number
                      TB(n)       prints n blank spaces (tab)
                      WR(n,i,j)               writes a real number (i and j are field lengths)

        +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ ...}


procedure ReadParameters;

var     ch1, ch2 : char;

begin
        Find('*');
        RI(jMax);                  RI(ItMax);
        RR(Tolerance);             RI(n);
        RR(L);                     RR(T);
        RR(D);
        RR(e);                     RR(a);
        RR(v);                     RR(kappa);
        RR(sigma);                 RR(sR);
        RR(CRf);                   RR(CPf);

        { Convert CRf and CPf from units of moles per liter to moles
          per cubic centimeter: }

                CRf := CRf / 1000;        CPf := CPf / 1000;

        RR(kmR);                   RR(kmP);
                        RR(rPore);
        RR(AlphaaR);               RR(AlphacR);
        RR(AlphaaS);               RR(AlphacS);
        RR(ioRref);                RR(ioSref);
        RR(DeltaU);                RR(i);
        RR(ThetaGuess);            RR(EtaGuess);
        Find('='); repeat     read(ch1)       until not(ch1 = ' ');
                read(ch2);
                CounterElectrode := ch1;
                CurrentCollector := ch2;
                readln;

end;  { ReadParameters }


procedure PrintTitle;

begin

        LF(5); TB(25); write('Flow-Thru Porous Electrode Redox Program');
        LF(2); TB(25); write('                  written  by');
        LF(2); TB(25); write('                   Paul Shain');
        LF(2); TB(29); write('(Date Written : 11 January 1984)');
        LF(1); TB(29); write('(      Revised : 21 May 1990)');
        LF(2); TB(40); write('Program begun at ', BeginTime);
        LF(2); TB(40); write('                on ', BeginDate);
        LF(3);

end;  { PrintTitle }


procedure EchoParameters;

procedure PrintReal( number : RealNumber );
begin  WR(number,10,5)  end;

begin
   LF(5); TB(15); write('Input Parameters — ');
   LF(2);
   TB(30); write('—— Dimensional Values ——');
   LF(2); TB(20);
   write('Number of Electrons Transferred in Main Reaction: '); writeln(n:3);
   LF(2); TB(20);
   write('Electrode Length (L) ='); PrintReal(L); write(' cm');
   LF(1); TB(20);
   write('Temperature (T) = '); PrintReal(T); write(' K');
   LF(1); TB(20);
   write('Reactant Diffusivity (D) = '); write(D:10,' sq cm/s');
   LF(1); TB(20);
   write('Dispersion coefficient (Da) = '); write(Da:10,' sq cm/s');
   LF(2); TB(20);
   write('Electrode Void Fraction (epsilon) = '); PrintReal(e);
   LF(1); TB(20);
   write('Electrode Surface-Area/Volume (a) = ');
                                         PrintReal(a); write(' sq cm/cu cm');
   LF(1); TB(20);
   write('Fluid Superficial Velocity (v) = '); PrintReal(v); write(' cm/s');
   LF(1); TB(20);
   write('Effective Solution Conductivity (kappa) = ');
                                          PrintReal(kappa); write(' mho/cm');
   LF(1); TB(20);
   write('Electrode conductivity (sigma) = ');
                                         PrintReal(sigma);  write(' mho/cm');
   LF(2); TB(20);
   write('Reactant Stoichiometric Coefficient (sR) = '); PrintReal(sR);
   LF(1); TB(20);
   write('Reactant feed concentration (cRf) = ');
```

```
                                          PrintReal(CRf);  write(' mol/cc');
LF(1); TB(20);
write('Reactant mass-transfer coefficient (kmR) = ');
                                          PrintReal(kmR); write(' cm/s');
write('Product mass-transfer coefficient (kmP) = ');
                                          PrintReal(kmP); write(' cm/s');
write('Pore radius (rPore) = '); PrintReal(rPore); write(' cm');
LF(2); TB(20); write('Main reaction');
LF(1); TB(25); write('anodic alpha = '); PrintReal(alphaaR);
LF(1); TB(25); write('cathodic alpha = '); PrintReal(alphacR);
LF(1); TB(20); write('Side reaction');
LF(1); TB(25); write('anodic alpha = '); PrintReal(alphaaS);
LF(1); TB(25); write('cathodic alpha = '); PrintReal(alphacS);

LF(2); TB(20); write('Exchange current densities ' );
LF(1); TB(25);
write('main reaction (ioRref) = '); PrintReal(ioRref);
                                                                  write(' amps/sq cm');
LF(1); TB(25);
write('side reaction (ioSref) = '); PrintReal(ioSref);
                                                                  write(' amps/sq cm');
LF(1); TB(20);
write('Delta U = '); PrintReal(DeltaU);  write(' volts');
LF(2); TB(20);
write('Current density = '); PrintReal(i);  write(' amps/sq cm');
LF(3); TB(20); write('Electrode Configuration = ');
case Configuration of

   UD:   begin
           write('UD — Upstream Counterelectrode,');
           LF(1); TB(46);
           write('        Downstream Current Collector')
         end;

   DU:   begin
           write('DU — Downstream Counterelectrode,');
           LF(1); TB(20);
           write('        Upstream Current Collector')
         end;

   UU:   begin
           write('UU — Upstream Counterelectrode,');
           LF(1); TB(20);
           write('        Upstream Current Collector')
         end;

   DD:   begin
           write('DD — Downstream Counterelectrode,');
           LF(1); TB(20);
           write('        Downstream Current Collector')
         end

   end; { Configuration cases }

LF(5); TB(30); write('— Dimensionless Groups — ');
LF(2); TB(20); write('P1 = ');  PrintReal(P1);
LF(1); TB(20); write('P2 = ');  PrintReal(P2);
LF(1); TB(20); write('P3 = ');  PrintReal(P3);
LF(1); TB(20); write('P4 = ');  PrintReal(P4);
LF(1); TB(20); write('P5 = ');  PrintReal(P5);
LF(1); TB(20); write('P6 = ');  PrintReal(P6);
```

```
   LF(1); TB(20); write('P7 = ');  PrintReal(P7);

   LF(2); TB(20); write('ThetaPf = '); PrintReal(ThetaPf);
   LF(1); TB(20); write('AlphaL = ');  PrintReal(AlphaL);
   LF(2); TB(20); write('DPrime = ');  PrintReal(DPrime);
   LF(2); TB(20); write('IStar = ');  PrintReal(IStar);
   LF(5);

end;  { EchoParameters }


procedure Indent;  { Indent }
begin  TB(5)  end;  { Indent }


procedure NL;  { NewLine }
begin  LF(1);  Indent; Indent  end;  { NewLine }


procedure PrintParameters;

{... Purpose:  List all of the characteristic information used in the
               execution of BandAid.

     Variables global to routine:
                              from BandAid — nEqns,
                                             yMin, yMax,
                                             MeshSize (via NonBandCalcs),
                                             ImageFirstPoint, ImageLastPoint,
                                             ItMax,
                                             FactorIncrement,
                                             ReduceTimeOption
                                                                    ...}

var      PrintTime, PrintDate : DateType;

begin  { body of PrintParameters }

   SetTimeString(PrintTime);  SetDateString(PrintDate);
   LF(11); TB(25);
   write(' BandAid — Version 3  (October 20, 1984)');
   writeln;
   TB(32);
   write('(Start  Time = ',PrintTime, ', Date = ',PrintDate,')');
   LF(4); write(' Procedure Specifications — ');
   LF(1); Indent; write('Number of Equations and Unknowns = ',nEqns:3);
   LF(2); Indent;  write('X-Direction specifications —');
   LF(1); NL; write('Minimum Distance (XMin) = '); WR(yMin,10,5);
   NL; write('Maximum Distance (XMax) = '); WR(yMax,10,5);
   NL; write('Number of Mesh Points = ',jMax:4);
   NL; write('Mesh Size = '); WR(MeshSize,10,5);
   NL; write('ImagePoint at ');
   if (ImageFirstPoint) and (ImageLastPoint) then
      write('the First and Last Mesh Points ')
   else if (ImageFirstPoint) then
      write('the First Mesh Point ')
   else if (ImageLastPoint) then
      write('the Last Mesh Point ')
   else write('None of the Mesh Points ');
   LF(2); Indent; write('Options and Parameter Settings —');
   LF(1); NL; write('Maximum Number of Iterations = ', ItMax:3);
   LF(1); NL;
   if ReduceTimeOption then
      write('Calculation–Time Reduction Option Selected')
   else
      write('Calculation–Time Reduction Option Not Selected');
   NL;
   write('Increment Factor For Numerical Differentiation = ');
                                                               WR(FactorIncrement,10,3);
   LF(3)

end;  { PrintParameters }


procedure RunTimeDiagnostics;

{...  Purpose:  Report diagnostic information on the execution of
                      BandCore.

        Variables global to the routine from BandAid (via NonBandCalcs):
                                                                   NumberOfIterations,
                                                      CPUTimeUsed
                                                                              ...}

begin  { body of RunTimeDiagnostics }

   LF(8); write(' Run–Time Diagnostics — ');
   LF(2); write(' Number of Iterations = ',NumberOfIterations:2);
   LF(1);
   write(' Computation Time'); write(' = ',CPUTimeUsed:6,' MilliSeconds ');
   LF(3)

end;  { RunTimeDiagnostics }


procedure PrintProfiles( function x( Node : integer ) : RealNumber;
                                        FinalResult : ValueArray );

{...  Purpose:  List values of the dependent and independent variables
                      at each mesh point.

        Variables global to the routine from BandAid —
                                        ImageFirstPoint, ImageLastPoint, XDist, N
                                                                              ...}

var      j : integer;
         OtherResult : ValueArray;


procedure PrintReal( number : RealNumber );
begin  WR(number,10,5)  end;


        { calculate the dimensional distance, concentration, and potential
            and store them in the array OtherResult }

function dist : RealNumber;
begin  dist := x(j) * v / a / kmR  end;

function conc : RealNumber;
begin       conc := FinalResult[1,j] * cRf                    end;
```

```
function pot : RealNumber;
begin
   pot := ( FinalResult[2,j] - ln( -n * F *kmR * cRf / sR / ioRref ) )
                                          * R * T / F / alphaCR
end; { pot }

function yfunction( y : RealNumber ) : RealNumber;
begin  yfunction := y  end;

function distfunc( y : RealNumber ) : RealNumber;
begin
            if (v = 0) then
                        distfunc := y / a
            else
                        distfunc := y * v / a / kmR
end;


begin  { body of PrintProfiles }

for j := 1 to jmax do
   begin
      OtherResult[1,j] := conc;
      OtherResult[2,j] := pot
   end;

writeln('       DIMENSIONLESS CONCENTRATION AND POTENTIAL PROFILES');
writeln('       ============= ============= === ========= ========');
writeln;  writeln;

writeln('     j           Y                   Theta                Eta');
writeln;

ListPrint(FinalResult, yfunction, 2, jmax, ImageFirstPoint,
                                                    ImageLastPoint, yMin, yMax);
LF(4);
writeln('       DIMENSIONAL CONCENTRATION AND POTENTIAL PROFILES');
writeln('       =========== ============= === ========= ========');
writeln; writeln;
writeln(' Node          x, cm            conc, mol/cc             eta, V');
writeln;
ListPrint(OtherResult, distfunc, 2, jmax, ImageFirstPoint,
                                                     ImageLastPoint, yMin, yMax)

end;  { PrintProfiles }


procedure PrintBarProfiles( FinalResult : ValueArray);

var      j : integer;

function ThetaBar( j : integer ) : RealNumber;
begin  ThetaBar := FinalResult[1,j]  end;

function EtaBar( j : integer ) : RealNumber;
begin  EtaBar := FinalResult[2,j]  end;


procedure PrintHeading;
begin
   write( BarProfiles, 'Node        ',
                         'ThetaBar                      ',
                         'EtaBar');
   writeln( BarProfiles ); writeln( BarProfiles )
end;  { PrintHeading }

procedure PrintLine;
begin
   write( BarProfiles, j :4, ' ' : 5 );
   write( BarProfiles, ThetaBar( j ) : 22, ' ' : 5 );
   write( BarProfiles, EtaBar( j ) : 22 );
   writeln( BarProfiles )
end;  { PrintLine }

begin   { body of PrintBarProfiles }
   rewrite( BarProfiles );
   PrintHeading;
   for j := 1 to jMax do PrintLine
end;    { PrintBarProfiles }


procedure PrintTimeSummary;

{... Purpose:  Report a summary of the runtime diagnostics for the
                         execution of BandAid.

        Variables global to the routine:  from BandAid — ClockInitial
                                                                     ...}

var     PrintTime, PrintDate : DateType;
        ComputTime : integer;
        RoundedComputTime : integer;

begin  { body of PrintTimeSummary }

   SetTimeString(PrintTime); SetDateString(PrintDate);
   LF(8); write('Summary of RunTime Diagnostics — '); LF(3);
   ComputTime := SystemClock - ClockInitial;
   TB(10);
   write('Total Computation Time = ',ComputTime:8,' Milli-Seconds');
   TB(5);
   RoundedComputTime := round(ComputTime /60000);
   write('( ',RoundedComputTime:4,' Minutes)');
   LF(7); TB(25); write('BandAid — Version 2');
   LF(1); TB(35); write('(Stop Time = ',PrintTime);
   write(', Date = ',PrintDate,')');
   LF(5)

end;  {PrintTimeSummary}


{... ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

                    End of Printout Routines

    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ ...}


end.  { module SteadyIO }
```

APPENDIX O

Listing of Program *Impedance*

```
[GFLOATING,INHERIT('AidMod.pen', 'ImpedDcl.pen', 'ImpedIo.pen' )]

{... ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

        Program Title : Impedance
        Written By :     Paul Shain

        Date written : May 16, 1984
        Date modified: May 22, 1990

        Purpose: This program calculates the concentration and poten-
                 tial distributions in a flow-through porous electrode
                 with an alternating applied current according to the
                 model of Trainham and Newman (Electrochim. Acta, 26,
                 455 (1981)). The approach used is that used by
                 Tribollet and Newman (J. Electrochem. Soc., 131, 2780
                 (1984)).

        Variable Number 1 == real dimensionless concentration
        Variable Number 2 == imaginary dimensionless concentration
        Variable Number 3 == real dimensionless potential
        Variable Number 4 == imaginary dimensionless potential

        Parameters are those for reduction of a reactant in a redox
        system. A side reaction and axial diffusion and dispersion are
        included.

        Double-layer capacitance is included.


    +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ ...}


program Impedance( input, output );


{... The constant, type, and variable declarations are declared in the
     file, 'ImpedDcl.pas.' This program was written in separate pieces
     to remove from view parts which do not contribute to an under-
     standing of how the program works. A list of the files that make
     up the program follows.

       AidMod    -  Contains constant, type, and var declarations required
                    by BandAid, and the BandAid procedure. Is inherited
                    by the other files.

       ImpedDcl -  Contains all declarations needed by the program Imped.
                    Is inherited by ImpedIO and Imped.

       ImpedIO   -  Contains I/O routines used by Imped. Is inherited by
                    Imped.

       Imped     -  Contains the equations to be solved by BandAid and
                    their boundary conditions.

       Inheriting is a nonstandard feature available in VAX Pascal. A
       "module" containing declarations and procedures can be compiled to
       produce an object file of filetype "obj" and an environment file
       of filetype "pen." Other files can "inherit" environments to use
       the declarations or procedures contained therein. After a program
```

```
      is compiled, its object file is linked to the object files of the
      environments it has inherited to produce an executable file.

                                                                   ... }


function Equation( i, j : integer;
                   y, h : RealNumber;
                   var NewResult : ValueArray;
                   function cInterp( k : integer;
                                     y : RealNumber;
                                     var Result : ValueArray) :
                                                             RealNumber;
                   function dnFdyn( n : integer;
                                    function F( y : RealNumber ) :
                                                             RealNumber;
                                    y : RealNumber;
                                    Approx : DiffApprox ) : RealNumber )
                                                           : RealNumber;

   function Var1 ( y : RealNumber ) : RealNumber;
   begin Var1 := cInterp(1,y,NewResult) end;

   function Var2 ( y : RealNumber ) : RealNumber;
   begin Var2 := cInterp(2,y,NewResult) end;

   function Var3 ( y : RealNumber ) : RealNumber;
   begin Var3 := cInterp(3,y,NewResult) end;

   function Var4 ( y : RealNumber ) : RealNumber;
   begin Var4 := cInterp(4,y,NewResult) end;

   function T1 : RealNumber;
   begin  T1 := Var1(y)  end;

   function T2 : RealNumber;
   begin  T2 := Var2(y)  end;

   function E1 : RealNumber;
   begin  E1 := Var3(y)  end;

   function E2 : RealNumber;
   begin  E2 := Var4(y)  end;

   function TBar : RealNumber;
   begin  TBar := 1.0  end;

   function EBar : RealNumber;
   begin  EBar := ln( -n*F*kmR*cRf/sR/ioRref )  end;

   function dT1dy ( y : RealNumber ) : RealNumber;
   begin  dT1dy := dnFdyn(1,Var1,y,Cendiff)  end;

   function dT2dy ( y : RealNumber ) : RealNumber;
   begin  dT2dy := dnFdyn(1,Var2,y,Cendiff)  end;

   function dE1dy ( y : RealNumber ) : RealNumber;
   begin  dE1dy := dnFdyn(1,Var3,y,CenDiff)  end;

   function dE2dy ( y : RealNumber ) : RealNumber;
```

```
    begin  dE2dy := dnFdyn(1,Var4,y,CenDiff)  end;

    function d2T1dy2( y : RealNumber ) : RealNumber;
    begin  d2T1dy2 := dnFdyn(2,Var1,y,Cendiff)  end;

    function d2T2dy2( y : RealNumber ) : RealNumber;
    begin  d2T2dy2 := dnFdyn(2,Var2,y,Cendiff)  end;

    function d2E1dy2 ( y : RealNumber ) : RealNumber;
    begin  d2E1dy2 := dnFdyn(2,Var3,y,CenDiff)  end;

    function d2E2dy2 ( y : RealNumber ) : RealNumber;
    begin  d2E2dy2 := dnFdyn(2,Var4,y,CenDiff)  end;


function UpStrmBC( i : integer; y : RealNumber ) : RealNumber;

begin

    case i of

        1:  UpStrmBC := T1;    { = 0 }

        2:  UpStrmBC := T2;    { = 0 }

        3:  UpStrmBC := dE1dy(y);    { = 0 }

        4:  case Configuration of

                UD:  UpStrmBC := dE2dy(y) + P5*DeltaIStar;  { = 0 }
                DU:  UpStrmBC := dE2dy(y) + P6*DeltaIStar;  { = 0 }
                UU:  UpStrmBC := dE2dy(y) - P2*DeltaIStar;  { = 0 }
                DD:  UpStrmBC := dE2dy(y)    { = 0 }

            end    { Configuration cases }

    end    { i cases }

end;  { UpStrmBC }


function DiffEQ( i : integer; y : RealNumber ) : RealNumber;

var     Denom, Beta, SideReactionTerm : RealNumber;
        g, F1, AlphaC, AlphaS : RealNumber;

begin  { body of DiffEq }

    F1 := 1 + (1 - TBar) / ThetaPf;
    Beta := 1 + (AlphaaR / AlphacR);
    AlphaC := AlphacS / AlphacR;
    AlphaS := ( AlphaaS + AlphacS ) / AlphacR;
    g := P1 * P7 / ThetaPf;
    Denom := 1 + exp( EBar ) + g * exp( Beta*EBar ) ;

    case i of

        1: DiffEQ:= sqr(Denom)*(-T2*Omega - DPrime*d2T1dy2(y) )
                        + E1 *
```

```
                              (
                                 ( exp(EBar) + g*Beta*exp(Beta*EBar) ) *
                                                   ( F1*P1*exp(Beta*EBar) - TBar )
                                 - P1 * F1 * Denom * Beta * exp(Beta*EBar)
                              )
                         - T1 * Denom * ( 1 + P1 * exp(Beta*EBar) / ThetaPf );

        2: DiffEQ := sqr(Denom)*(T1*Omega - DPrime*d2T2dy2(y))
                         + E2 *
                              (
                                 ( exp(EBar) + g*Beta*exp(Beta*EBar) ) *
                                                   ( F1*P1*exp(Beta*EBar) - TBar )
                                 - P1 * F1 * Denom * Beta* exp(Beta*EBar)
                              )
                         - T2 * Denom * ( 1 + P1 * exp(Beta*EBar) / ThetaPf );

        3: DiffEQ := sqr(Denom) * ( d2E1dy2(y) + P2 * Cdl * Omega * E2 )

                    - P2 * E1 * (
                             -Denom * P1 * F1 * Beta * exp(Beta*EBar)
                             + ( exp(EBar) + g*Beta*exp(Beta*EBar) )
                                          *( F1*P1*exp(Beta*EBar) - TBar )
                             -sqr(Denom)*P3*exp(-alphaC*EBar) *
                                ( alphaC + P4*exp(alphaS*EBar)*(alphaS-alphaC) )
                                   )

                    - Denom * T1 * P2 * ( 1 + P1*exp(beta*ebar)/ThetaPf );


        4: DiffEQ := sqr(Denom) * ( d2E2dy2(y) - P2 * Cdl * Omega * E1 )

                    - P2 * E2 * (
                             -Denom * P1 * F1 * Beta * exp(Beta*EBar)
                            + ( exp(EBar) + g*Beta*exp(Beta*EBar) )
                                          *( F1*P1*exp(Beta*EBar) - TBar )
                            -sqr(Denom)*P3*exp(-alphaC*EBar) *
                                ( alphaC + P4*exp(alphaS*EBar)*(alphaS-alphaC) )
                                   )

                    - Denom * T2 * P2 * ( 1 + P1*exp(beta*ebar)/ThetaPf )


        end  { i cases }

end;  { DiffEQ }


function DnStrmBC( i : integer; y : RealNumber ) : RealNumber;

begin

    case i of

        1:  DnStrmBC := T1;  { = 0 }

        2:  DnStrmBC := T2;  { = 0 }

        3:  DnStrmBC := dE1dy(y);  { = 0 }

        4:  case Configuration of
```

```
            UD:  DnStrmBC := dE2dy(y) - P6*DeltaIStar;  { = 0 }
            DU:  DnStrmBC := dE2dy(y) - P5*DeltaIStar;  { = 0 }
            UU:  DnStrmBC := dE2dy(y);                  { = 0 }
            DD:  DnStrmBC := dE2dy(y) + P2*DeltaIStar   { = 0 }

         end  { Configuration cases }

   end  { i cases }

end;  { DnStrmBC }


begin  { body of Equation }

   if ( j = 1 ) then

      case i of

         1:  Equation := UpStrmBC(1,yMin);
         2:  Equation := UpStrmBC(2,yMin);
         3:  Equation := UpStrmBC(3,yMin);
         4:  Equation := UpStrmBC(4,yMin)

      end  { i cases }

   else if ( j = jMax ) then

      case i of

         1:  Equation := DnStrmBC(1,yMax);
         2:  Equation := DnStrmBC(2,yMax);
         3:  Equation := DnStrmBC(3,yMax);
         4:  Equation := DnStrmBC(4,yMax)

      end   { i cases }

   else case i of

      1:  Equation := DiffEQ(1,y);
      2:  Equation := DiffEq(2,y);
      3:  Equation := DiffEQ(3,y);
      4:  Equation := DiffEQ(4,y)

   end  { i cases }

end;  { Equation }


function Converged( function y( j : integer ) : RealNumber;
                    h : RealNumber;
                    var NewResult, Deviation : ValueArray;
                    var Residual : ValueArray;
                    function cInterp( k : integer;
                                      y : RealNumber;
                                      var Result : ValueArray ) :
                                                                  RealNumber;
                    function dnFdyn(  n : integer;
                                      function F( y : RealNumber ) :
```

```
                                                                  RealNumber;
                                        y : RealNumber;
                                        Approx : DiffApprox ) : RealNumber
                                                                     ) :
                                                                     boolean;

var      k, j : integer;

begin

   Converged := true;

   for k := 1 to nEqns do
      for j := 1 to jMax do
                                  begin
                                    if ( NewResult[k,j] = 0 ) then
                                               if ( abs(Deviation[k,j]) > Tolerance ) then
                                                          Converged := false;
                                    if ( NewResult[k,j] <> 0 ) then
               if (abs(Deviation[k,j] /NewResult[k,j]) > Tolerance) then
                                                  Converged := false
                                  end

end;  { Converged }


procedure NonBandCalcs( LastIteration : boolean;
                         Iteration, CPUTime : integer;
                         function y( j : integer ) : RealNumber;
                         h : RealNumber;
                         var NewResult, Deviation : ValueArray;
                         var Residual : ValueArray;
                         function cInterp( k : integer;
                                            y : RealNumber;
                                           var Result : ValueArray ) :
                                                                RealNumber;
                         function dnFdyn( n : integer;
                                        function F( y : RealNumber ) :
                                                                RealNumber;
                                           y : RealNumber;
                                           Approx : DiffApprox ) :
                                                              RealNumber );

var      j : integer;

begin  { body of NonBandCalcs }

   if ( LastIteration ) then
      begin
         NumberOfIterations := Iteration;  { rename the parameters    }
         MeshSize := h;                       { passed from BandAid for }
         CPUTimeUsed := CPUTime;               { use in the i/o routines  }
{*       PrintParameters;
         RunTimeDiagnostics;
         PrintProfiles( y, NewResult, Numberofiterations );
         PrintTimeSummary       *}
      end

end;  { NonBandCalcs }
```

```
procedure SetParameters;


procedure Error;

begin
   writeln; write('Configuration ',CounterElectrode,CurrentCollector,
                                             ' is not UD, DU, UU, or DD. ');
   writeln; write('Check the data file and try again. ');
   writeln; writeln;
   halt
end;   { Error }



begin   { body of SetParameters }

kmR := D / rPore;
kmP := kmR;
Da := 0;
DPrime := e * D * a / kmR;

P1 := (-sR*ioRref)/(n*F*kmR*CRf) ** (1+(AlphaaR/AlphacR));

P2 := AlphacR*n*F*F*kmR*CRf/(sR*a*R*T) *(1/kappa + 1/sigma);

P3 := -sR*ioSref/(n*F*kmR*CRf) * exp( AlphacS*F/(R*T)*DeltaU )
                     * ( -n*F*kmR*CRf/(sR*ioRref) ) ** (AlphacS/AlphacR);

P4 := (- sR*ioRref/(n*F*kmR*CRf)) ** ( (AlphaaS+AlphacS)/AlphacR )
                              * exp( - F*(AlphaaS+AlphacS)*DeltaU/(R*T) );

P5 := -sigma*P2/(sigma + kappa);
P6 := kappa/sigma * P5;
P7 := kmR / kmP;
ThetaPf := CPf/CRf;
AlphaL := a*L;

{*******
reset ( pFactors );
readln( pFactors, P1 );
readln( pFactors, P2 );
readln( pFactors, P3 );
readln( pFactors, P4 );
readln( pFactors, P5 );
readln( pFactors, P6 );
readln( pFactors, P7 );
readln( pFactors, thetapf );
readln( pFactors, Dprime );
readln( pFactors, alphaL );
readln( pFactors, istar );
readln( pFactors, cdl );
 ********}

   case CounterElectrode of

      'U':  case CurrentCollector of

            'U':     Configuration := UU;
            'D':     Configuration := UD;
            otherwise  Error

            end;  { CurrentCollector cases }

      'D':  case CurrentCollector of

            'U':    Configuration := DU;
            'D':    Configuration := DD;
            otherwise  Error

            end;  { CurrentCollector cases }

      otherwise    Error

      end;  { CounterCollector cases }

      yMin := 0;
      yMax := AlphaL

end;  { SetParameters }


begin { body of Impedance }

ClockInitial := SystemClock;
SetTimeString(BeginTime); SetDateString(BeginDate);
ReadParameters;
SetParameters;
PrintTitle;
EchoParameters;

OmegaUnits := 1.0e+1;

write('Omega,Hz  Re(Z),Ohm.cm2  -Im(Z),Ohm.cm2  ');
writeln('Iterations');

while ( OmegaUnits < 1000001.0 ) do
 begin
   omega := OmegaUnits / (e * a * kmR );
   BandShell(  nEqns, jMax, ItMax, yMin, yMax, FactorIncrement,
               AbsoluteIncrement, ImageFirstPoint, ImageLastPoint,
               ReduceTimeOption, Guess, FinalResult, Deviation,
               Residual, Equation, Converged, NonBandCalcs );

           if (CurrentCollector = 'D') then WhichEnd := jmax
           else if (CurrentCollector = 'U') then WhichEnd := 1;

   ZReal := -( (R * T) / (alphaCR * F) ) * FinalResult[4,WhichEnd] / DeltaI;
   ZImag :=  ( (R * T) / (alphaCR * F) ) * FinalResult[3,WhichEnd] / DeltaI;

   write(OmegaUnits:20, ' ');
   writeln( ' ', ZReal:20, ' ', -ZImag:20, ' ',
                                          NumberofIterations:4);
   OmegaUnits := 10.0 * OmegaUnits
 end

end.  { Impedance }
```

APPENDIX P

Listing of Program *ImpedanceDcl*

```
[GFLOATING, INHERIT('AidMod.pen'), ENVIRONMENT('ImpedDcl.pen')]

module ImpedanceDcl( input, output, BarProfiles, pFactors );


const           {... BandAid Parameters ...}

    nEqns = 4;
    ImageFirstPoint = true;
    ImageLastPoint = true;
    FactorIncrement = 1e–6;
    AbsoluteIncrement = 1e–6;
    ReduceTimeOption = true;

                {... Physical Constants ...}

    F = 96487.0;            { Faraday's constant, coulombs/equivalent }
    R = 8.314;              { ideal gas constant, joules/mole–K }


type    alfa = packed array[1..11]of char;


var             {... BandAid Parameters ...}

    Guess, FinalResult, OtherResult, Deviation, Residual : ValueArray;
    j, jMax, ItMax : integer;
    Tolerance : RealNumber;

                    {... Physical Parameters ...}

    v, e, CRf, CPf, AlphaaR, AlphacR, AlphaaS, AlphacS : RealNumber;
    L,a, D, Da, kmR, rPore, sR, ioRref, ioSRef, T, kmP : RealNumber;
    i, kappa, sigma, DeltaU, Cdl : RealNumber;
    n : integer;

                    {... Dimensionless Parameters ...}

    AlphaL, DPrime, ThetaPf : RealNumber;
    P1, P2, P3, P4, P5, P6, P7, P8 : RealNumber;
    yMin, yMax : RealNumber;
    ZReal, ZImag : RealNumber;

                    {... Other Parameters ...}

    Configuration : (UD, DU, UU, DD);
    CounterElectrode, CurrentCollector : char;
    Theta1Guess, Theta2Guess, Eta1Guess, Eta2Guess  : RealNumber;
    DeltaI, DeltaIStar, OmegaUnits, Omega  : RealNumber;
    iStar : RealNumber;
    BarProfiles : text;
    pFactors : text;
    BarValues : ValueArray;
    WhichEnd : integer;
    MeshSize : RealNumber;
    CPUTimeUsed, ClockInitial : integer;
    NumberOfIterations : integer;
    BeginDate, BeginTime : alfa;

end.
```

APPENDIX Q

Listing of Program *ImpedanceIO*

```
[GFLOATING, INHERIT('ImpedDcl.pen', 'AidMod.pen'), ENVIRONMENT('ImpedIO.pen')]

module ImpedanceIO;


procedure SetTimeString( var TimeString : alfa );
begin  time(TimeString)  end;


procedure SetDateString( var DateString : alfa );
begin  date(DateString)  end;



{... ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

        Output Routines:
                The following routines print the input, output and
                runtime diagnostic messages for the BandShell calling
                program Imped.pas.

   ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ ...}


procedure ReadParameters;

var     ch1, ch2 : char;
        Node, int : integer;

begin
   Find('*');
   RI(jMax);            RI(ItMax);          RR(Tolerance);
   RI(n);               RR(L);              RR(T);
   RR(D);               RR(e);
   RR(a);               RR(v);              RR(kappa);
   RR(sigma);           RR(Cdl);            RR(sR);
   RR(CRf);             RR(CPf);

   { Convert CRf and CPf from moles per liter to moles per cc }

   CRf := CRf / 1000;   CPf := CPf / 1000;

   RR(kmR);             RR(kmP);            RR(rPore);
   RR(AlphaaR);
   RR(AlphacR);         RR(AlphaaS);        RR(AlphacS);
   RR(ioRref);          RR(ioSref);         RR(DeltaU);
   RR(i);               RR(DeltaI);         RR(OmegaUnits);

          if (v = 0) then
          DeltaIStar := (sR / n * F * cRf * kmR) * DeltaI
          else
          DeltaIStar := (sR / n * F * cRf * v) * DeltaI ;
   Omega := OmegaUnits  / ( e  *  a * kmR );

   RR(Theta1Guess);   RR(Eta1Guess);
   RR(Theta2Guess);   RR(Eta2Guess);

   Find('='); repeat  read(ch1)  until not(ch1 = ' ');
   read(ch2);
   CounterElectrode := ch1;
   CurrentCollector := ch2;
   readln;

{* reset( BarProfiles );
   readln( BarProfiles ); readln( BarProfiles );
   for j := 1 to jMax do
      readln(BarProfiles, int, BarValues[1,j], BarValues[2,j])    *}

end;  { ReadParameters }


procedure PrintTitle;

begin

   LF(5); TB(25); write('Flow-Thru Porous Electrode Redox Program');
   LF(2); TB(25); write('                 written  by');
   LF(2); TB(25); write('                  Paul Shain');
   LF(2); TB(29); write('(Date Written : 11 January 1984)');
   LF(2); TB(29); write('(      Revised : 22 May 1990)');
   LF(2); TB(40); write('Program begun at ', BeginTime );
   LF(2); TB(40); write('                     on ', BeginDate );
   LF(3);

end;  { PrintTitle }


procedure EchoParameters;

procedure PrintReal( number : RealNumber );
begin    WR(number,10,5)    end;

begin

   LF(5); TB(15); write('Input Parameters — ');
   LF(2); TB(30); write('—— Dimensional Values ——');
   LF(2); TB(20); write('Number of Electrons Transferred in',
                                                     ' Main Reaction = ');
                                PrintReal(n);
   LF(2); TB(20); write('Electrode Length (L) =');
                                PrintReal(L); write(' cm');
   LF(1); TB(20); write('Temperature (T) = ');
                                PrintReal(T); write(' K');
   LF(1); TB(20); write('Reactant Diffusivity (D) = ');
                                write(D:10,' sq cm/s');
   LF(1); TB(20); write('Dispersion coefficient (Da) = ');
                                write(Da:10,' sq cm/s');
   LF(2); TB(20); write('Electrode Void Fraction (epsilon) = ');
                                PrintReal(e);
   LF(1); TB(20); write('Electrode Surface-Area/Volume (a) = ');
                                PrintReal(a); write(' sq cm/cu cm');
   LF(1); TB(20); write('Fluid Superficial Velocity (v) = ');
                                PrintReal(v); write(' cm/s');
   LF(1); TB(20); write('Effective Solution Conductivity (kappa) = ');
                                PrintReal(kappa); write(' mho/cm');
   LF(1); TB(20); write('Electrode conductivity (sigma) = ');
                                PrintReal(sigma);  write(' mho/cm');
   LF(1); TB(20); write('Double-layer capacity (Cdl) = ');
                                PrintReal(Cdl);  write(' F/sq cm');
```

```
        {convert cdl to dimensionless form}
Cdl := a * Cdl * R * T * sR / ( alphacR * e * n * sqr(F) * cRf );

LF(2); TB(20); write('Reactant Stoichiometric Coefficient (sR) = ');
                                   PrintReal(sR);
LF(1); TB(20); write('Reactant feed concentration (cRf) = ');
                                   PrintReal(CRf);  write(' mol/cc');
LF(1); TB(20); write('Reactant mass-transfer coefficient (kmR) = ');
                                   PrintReal(kmR); write(' cm/s');
LF(1); TB(20); write('Reactant mass-transfer coefficient (kmP) = ');
                                   PrintReal(kmP); write(' cm/s');
LF(1); TB(20); write('Pore radius (rPore) = ');
                                   PrintReal(rPore); write(' cm');
LF(1); TB(20); write('Product feed concentration (cPf) = ');
                                   PrintReal(CPf);  write(' mol/cc');
LF(1); TB(20); write('Product mass-transfer coefficient (kmP) = ');
                                   PrintReal(kmP); write(' cm/s');
LF(2); TB(20); write('Main reaction');
LF(1); TB(25); write('anodic alpha = '); PrintReal(alphaaR);
LF(1); TB(25); write('cathodic alpha = '); PrintReal(alphacR);
LF(1); TB(20); write('Side reaction');
LF(1); TB(25); write('anodic alpha = '); PrintReal(alphaaS);
LF(1); TB(25); write('cathodic alpha = '); PrintReal(alphacS);

LF(2); TB(20); write('Exchange current densities ' );
LF(1); TB(25); write('main reaction (ioRref) = ');
                                   PrintReal(ioRref); write(' amps/sq cm');
LF(1); TB(25); write('side reaction (ioSref) = ');
                                   PrintReal(ioSref); write(' amps/sq cm');
LF(1); TB(20); write('Delta U = ');
                                   PrintReal(DeltaU);  write(' volts');
LF(2); TB(20); write('Current density = ');
                                   PrintReal(i);  write(' amps/sq cm');
LF(2); TB(20); write('Current variation = ');
                                   PrintReal(DeltaI);  write(' amps/sq cm');
LF(3); TB(20); write('Electrode Configuration = ');

  case Configuration of

     UD:   begin
             write('UD — Upstream Counterelectrode,');
             LF(1); TB(46);
             write('        Downstream Current Collector')
           end;

     DU:   begin
             write('DU — Downstream Counterelectrode,');
             LF(1); TB(20);
             write('                Upstream Current Collector')
           end;

     UU:   begin
             write('UU — Upstream Counterelectrode,');
             LF(1); TB(20);
             write('        Upstream Current Collector')
           end;

     DD:   begin
             write('DD — Downstream Counterelectrode,');
             LF(1); TB(20);
             write('        Downstream Current Collector')
```

```
                end

        end;  { Configuration cases }

LF(5);  TB(30);  write('—— Dimensionless Groups — ');
LF(2);  TB(20);  write('P1 = ');  PrintReal(P1);
LF(1);  TB(20);  write('P2 = ');  PrintReal(P2);
LF(1);  TB(20);  write('P3 = ');  PrintReal(P3);
LF(1);  TB(20);  write('P4 = ');  PrintReal(P4);
LF(1);  TB(20);  write('P5 = ');  PrintReal(P5);
LF(1);  TB(20);  write('P6 = ');  PrintReal(P6);
LF(1);  TB(20);  write('P7 = ');  PrintReal(P7);
LF(2);  TB(20);  write('ThetaPf = '); PrintReal(ThetaPf);
LF(1);  TB(20);  write('AlphaL = ');  PrintReal(AlphaL);
LF(2);  TB(20);  write('DPrime = ');  PrintReal(DPrime);

LF(2);  TB(20);  write('DeltaIStar = ');  PrintReal(DeltaIStar);  LF(5);
LF(2);  TB(20);  write('Capacity = ');  PrintReal(Cdl);  LF(5);

end;  { EchoParameters }



procedure Indent;  { Indent }
begin  TB(5)  end;  { Indent }



procedure NL;  { NewLine }
begin  LF(1);  Indent;  Indent  end;  { NewLine }



procedure PrintParameters;

{... Purpose:  List all of the characteristic information used in the
               execution of BandAid.

     Variables global to routine:
        from BandAid — nEqns,
                       yMin, yMax,
                       MeshSize  (via NonBandCalcs),
                       ImageFirstPoint, ImageLastPoint,
                       ItMax,
                       FactorIncrement,
                       ReduceTimeOption
                                                                      ...}
var     PrintTime, PrintDate : alfa;

begin  { body of PrintParameters }

   SetTimeString(PrintTime);   SetDateString(PrintDate);

   LF(11);  TB(25);
   write(' BandAid — Version 3  (October 20, 1984)');
   writeln;  TB(35);
   write('(Start  Time = ',PrintTime, ', Date = ',PrintDate,')');

   LF(4);  write(' Procedure Specifications — ');
   LF(1);  Indent;  write('Number of Equations and Unknowns = ',nEqns:3);
   LF(2);  Indent;  write('X-Direction specifications —');
```

```
    LF(1); NL; write('Minimum Distance (XMin) = '); WR(yMin,10,5);
    NL; write('Maximum Distance (XMax) = '); WR(yMax,10,5);
    NL; write('Number of Mesh Points = ',jMax:4);
    NL; write('Mesh Size = '); WR(MeshSize,10,5);
    NL; write('ImagePoint at ');
    if (ImageFirstPoint) and (ImageLastPoint) then
       write('the First and Last Mesh Points ')
    else if (ImageFirstPoint) then
       write('the First Mesh Point ')
    else if (ImageLastPoint) then
       write('the Last Mesh Point ')
    else write('None of the Mesh Points ');

    LF(2); Indent; write('Options and Parameter Settings —');
    LF(1); NL; write('Maximum Number of Iterations = ', ItMax:3);
    LF(1); NL;
    if ReduceTimeOption then
       write('Calculation-Time Reduction Option Selected')
    else write('Calculation-Time Reduction Option Not Selected');
    NL; write('Increment Factor For Numerical Differentiation = ');
    WR(FactorIncrement,10,3);
    LF(3)

end;  { PrintParameters }


procedure RunTimeDiagnostics;

{... Purpose:  Report diagnostic information on the execution of
               BandCore.

        Variables global to the routine:
           from BandAid (via NonBandCalcs):
              NumberOfIterations,
              CPUTimeUsed
                                                                  ...}

begin  { body of RunTimeDiagnostics }

    LF(8); write(' Run-Time Diagnostics — ');
    LF(2); write(' Number of Iterations = ',NumberOfIterations:2);
    LF(1); write(' Computation Time');
    write(' = ',CPUTimeUsed:6,' MilliSeconds ');
    LF(3)

end;  { RunTimeDiagnostics }


procedure PrintProfiles( function y( j : integer ) : RealNumber;
                         FinalResult : ValueArray;
                                                                              Iterations : integer );

{...    Purpose:  List values of the dependent and independent variables
                  at each mesh point.

          Variables global to the routine:
             from BandAid — ImageFirstPoint, ImageLastPoint, XDist, N
                                                                  ...}

var      j : integer;
```

```
procedure PrintReal( number : RealNumber );
begin  WR(number,10,5)  end;


function dist : RealNumber;
begin  dist := y(j) * v / a / kmR   end;


function conc : RealNumber;
begin  conc := FinalResult[1,j] * cRf  end;

function pot : RealNumber;
begin  pot := FinalResult[3,j] * R * T / F / alphaCR  end;

function imconc : RealNumber;
begin  imconc := FinalResult[2,j] * cRf  end;

function impot : RealNumber;
begin  impot := FinalResult[4,j] * R * T / F / alphaCR  end;

function yfunction(y : RealNumber) : RealNumber;
begin        yfunction := y          end;

function distfunc(y : RealNumber) : RealNumber;
begin
          if (v = 0) then
                   distfunc := y / a
          else
                   distfunc := y * v / a / kmR
end;


begin  { body of PrintProfiles }

for j := 1 to jmax do
   begin
      OtherResult[1,j] := conc;
      OtherResult[2,j] := imconc;
      OtherResult[3,j] := pot;
      OtherResult[4,j] := impot
   end;

TB(28); writeln('DIMENSIONLESS CONCENTRATION AND POTENTIAL PROFILES');
TB(28); writeln('============= ============= === ========= ========');
LF(2);

write('Node'); TB(12); write('Y'); TB(14); write('Theta'); TB(13);
write('ThetaIm'); TB(14); write('Eta'); TB(15); write('EtaIm');
writeln;

ListPrint(FinalResult, yfunction, 4, jmax, ImageFirstPoint,
                                             ImageLastPoint, yMin, yMax);
LF(4);
{
TB(28); writeln('DIMENSIONAL CONCENTRATION AND POTENTIAL PROFILES');
TB(28); writeln('=========== ============= === ========= ========');
LF(2);

write('Node'); TB(10); write('x,cm'); TB(13); write('conc'); TB(15);
write('imconc'); TB(14); write('pot'); TB(15); write('impot'); writeln;
LF(2);
ListPrint(OtherResult, distfunc, 4, jmax, ImageFirstPoint,
                                            ImageLastPoint, yMin, yMax);
}
LF(4);

end;


procedure PrintTimeSummary;

{... Purpose:  Report a summary of the runtime diagnostics for the
                  execution of BandAid.

        Variables global to the routine:
           from BandAid — ClockInitial
                                                                        ...}
var     PrintTime, PrintDate : alfa;
        ComputTime : integer;
        RoundedComputTime : integer;

begin  { body of PrintTimeSummary }

   SetTimeString(PrintTime);
   SetDateString(PrintDate);
   LF(8); write('Summary of RunTime Diagnostics — '); LF(3);
   ComputTime := SystemClock - ClockInitial;
   TB(10);
   write('Total Computation Time = ',ComputTime:8,' Milli-Seconds');
   TB(5);
   RoundedComputTime := round(ComputTime /60000);
   write('( ',RoundedComputTime:4,' Minutes)');

   LF(7); TB(25); write('BandAid — Version 3');
   LF(1); TB(35); write('(Stop Time = ',PrintTime);
   write(', Date = ',PrintDate,')');
   LF(5)

end;  {PrintTimeSummary}


{... +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

                  End of Printout Routines

    +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ ...}


end.     { module ImpedanceIO }
```

CHAPTER 9

Porous Electrode Experiments and Simulations

This chapter begins with a discussion of the apparatus for measuring the frequency response of a flow-through porous electrode. Results of one such experiment are shown.

The responses of various transmission lines, electric circuit networks analagous to porous electrodes, are discussed to show how porous electrodes are expected to behave.

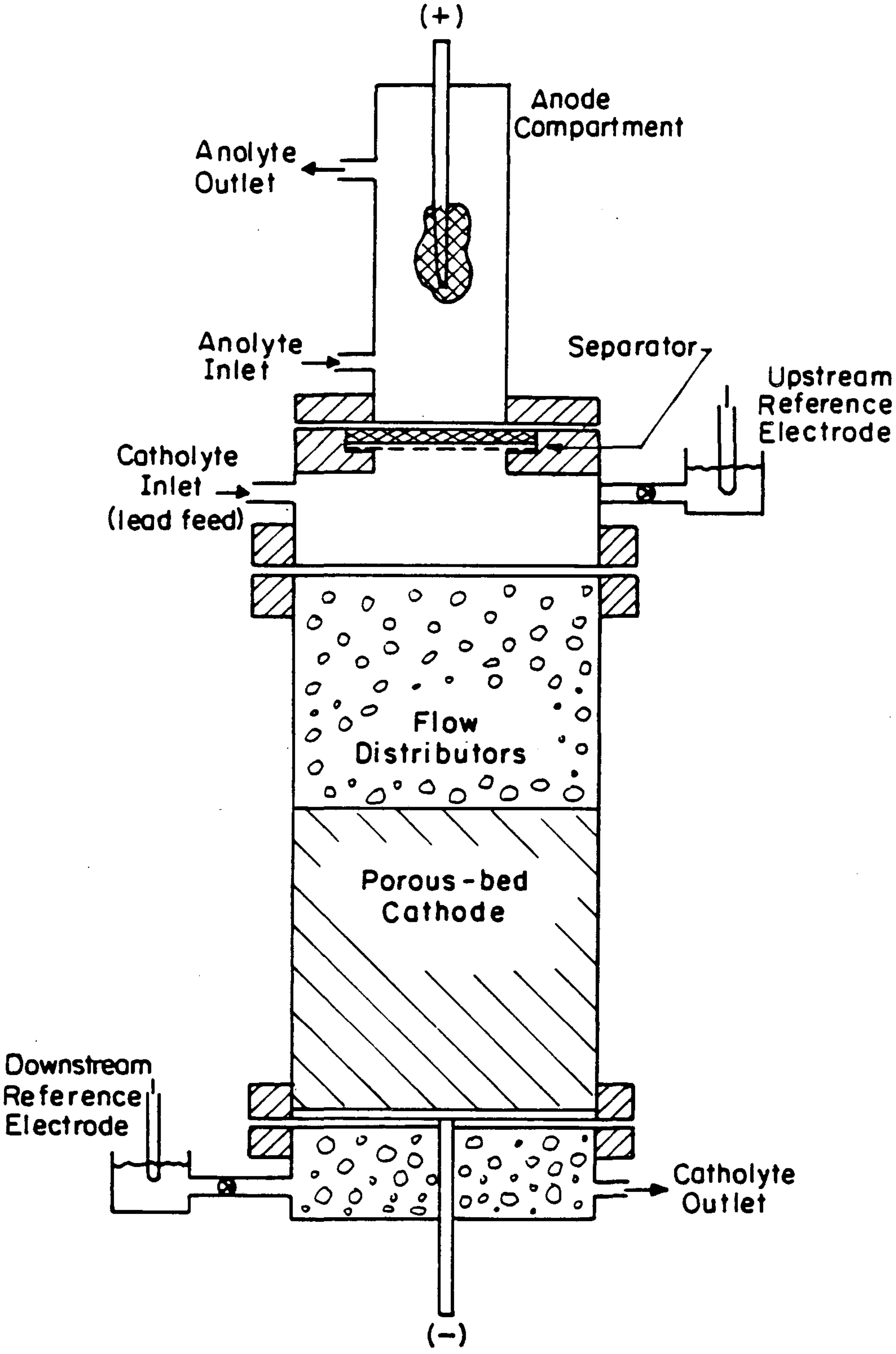
Results of the model presented in the previous chapter are then presented and compared to the experimental results. The effect of changing parameters on the results of the model are also shown.

9.1. Experimental apparatus

The flow-through porous electrode apparatus was the same one used by Matlosz[1, 2] and (except for the working electrode material) by Trost[3] and Sisler.[4] A photograph of the apparatus was included in Trost's dissertation.[5]

Figure 9-1 is a schematic diagram of the electrode. The working electrode was a reticulated vitreous carbon (RVC) cylinder (5 inches long, 2 inches in diameter) in a plastic tube. The RVC was manufactured by ERG, Inc. (Oakland, California). The counterelectrode was a platinum/rhodium screen welded to a current collector rod. Both current collectors were made of tantalum. Electrolyte was pumped through the working and counter electrode compartments (which were separated by a Nafion membrane) and rotameters were used to measure the flow rate. A calomel reference electrode was placed downstream from the working electrode for electrical control of the system.

Several pieces of electronic equipment served to control the apparatus. A Stonehart BC1200



XBL 839-6461

Figure 9-1. Schematic diagram of porous electrode apparatus.

Potentiostat and a Princeton Applied Research 175 Universal Programmer were used to apply a specified current to the electrode by controlling its potential relative to the reference electrode. A device fabricated by the Lawrence Berkeley Laboratory Electronics Shop was used to subtract approximately any direct-current bias, *e.g.* the open-circuit potential, for more accurate measurement of the potential variation by the frequency response analyzer.[6] The main piece of controlling equipment was the Solartron 1254 Frequency Response Analyzer. This device adds a perturbation to the current set by the programmer and potentiostat and calculates the real and imaginary parts of the impedance from the potential response at frequencies from 10 μHz to 65.535 kHz. A Central Point Software Laser128 computer (an "Apple clone") was used to change the settings on the Solartron via its keyboard and to capture and store experimental data for later manipulation. A Nicolet 206 digital oscilloscope was used to monitor the input and output signals (current perturbation and voltage response) to check that they were sine waves.

9.2. Experimental results

The solution used as both anolyte and catholyte was 1 N HCl, 0.25 M $FeCl_3$, 0.25 M $FeCl_2$. The flow rate was zero. The measurements were made around the open-circuit potential. The perturbation was set at 10 mV rms. This voltage was divided by 1000; thus the voltage perturbation was a sine wave of amplitude $10\sqrt{2}$ μV—small enough so that the system's response was linear. This was shown by a lack of response at harmonic frequencies.

Figure 9-2 is a Nyquist plot of the experimental results. The experiment was repeated four times with consistent results. It is not clear why the results cross the imaginary axis. An ohmic potential drop or contact resistance between electrode and current collector would shift the results in the opposite direction (*i.e.*, in the positive direction). However, results of the model have this feature if the ohmic potential drop in the solution is important (P_5 is large or κ is small). In section 9.4 these results and those of de Levie are compared to the predictions of the model developed in Chapter 8.

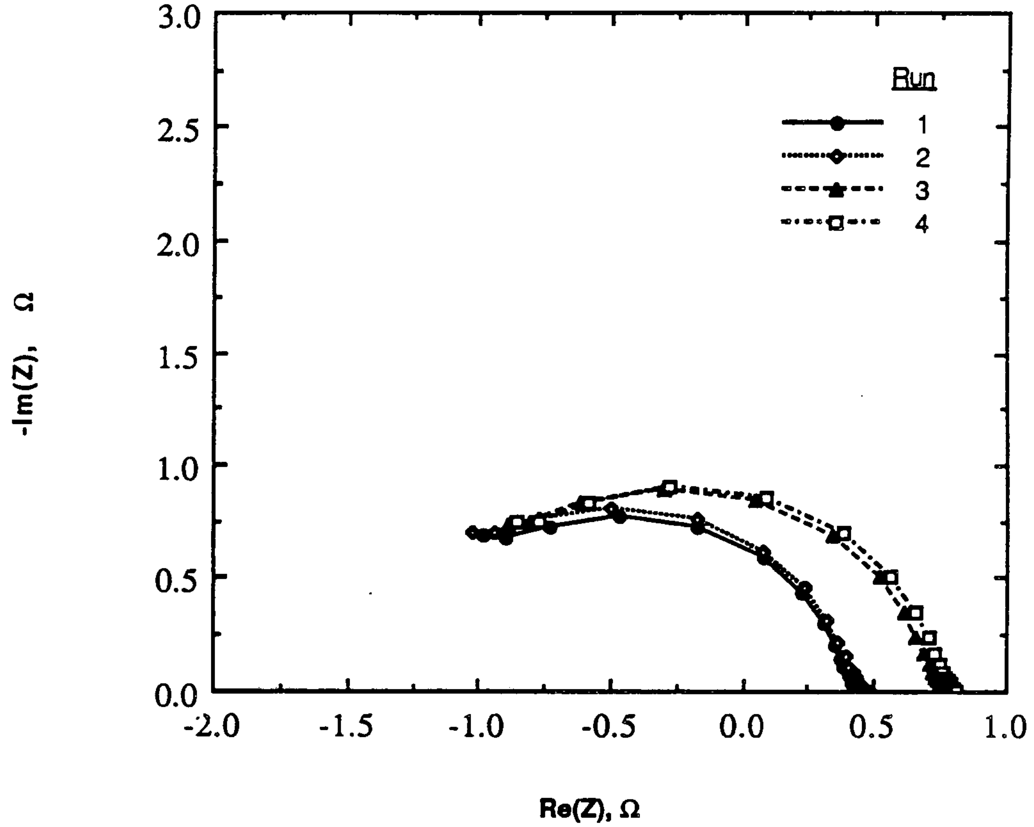


Figure 9-2. Nyquist plot of the impedance of the experimental system.

9.3. Transmission lines

It is often convenient to have a simple approximation to a complicated model. Such an approximation can be used to make certain that the more complicated model behaves correctly. Here we use an electrical circuit analogous to the porous electrode. The "equivalent circuit" that comes closest to resembling a porous electrode is a finite transmission line with leakage and capacitance. Both have two parallel conduction paths separated by resistance to charge transfer (leakage), capacitance, and possibly induction. The impedance of finite and infinite transmission lines can be calculated.[7] For a finite the transmission line analogous to the electrode of interest, the impedance is

$$Z = \tanh\left\{ d\left[(R + j\omega L)(G + j\omega C)\right]^{1/2}\right\}\left[\frac{R + j\omega L}{G + j\omega C}\right]^{1/2} , \qquad (1)$$

where Z is the impedance of the transmission line; G, the conductance (leakage) per length; R, the resistance per length; C, the capacitance per length; d, the length; and L, the inductance per length. Nyquist plots of the impedance for finite transmission lines are shown in Figures 9-3 through 9-7. The frequency range examined is 1 μHz to 1 MHz. For the base case shown in Figure 9-3, the parameter values were estimated to approximate the experimental conditions: d = 12.7 cm, R = 0.225 Ω/cm, C = 0.2 F/cm, G = 134.1 S/cm, and L = 0.

The parameters d, C, G, and R are analogous to the packed-bed parameters d, C_{dl}, i_o, and some combination of $\frac{1}{\kappa}$ and $\frac{1}{\sigma}$. There is no obvious analog to the inductance L. Knowing how the values of these parameters affect the impedance will help in fitting experimental results with the model.

The response of a transmission line as a function of length is shown in Figure 9-4. Because the impedance is proportional to the hyperbolic tangent of a factor times the length, its derivative with respect to length is proportional to the square of the hyperbolic secant of the factor times the length. This quantity is positive; therefore the impedance increases with increasing length.

Figure 9-5 shows the effect of changing the capacitance. If the capacitance is zero, the impedance is constant and real. All the curves follow the same path, but over the same frequency range

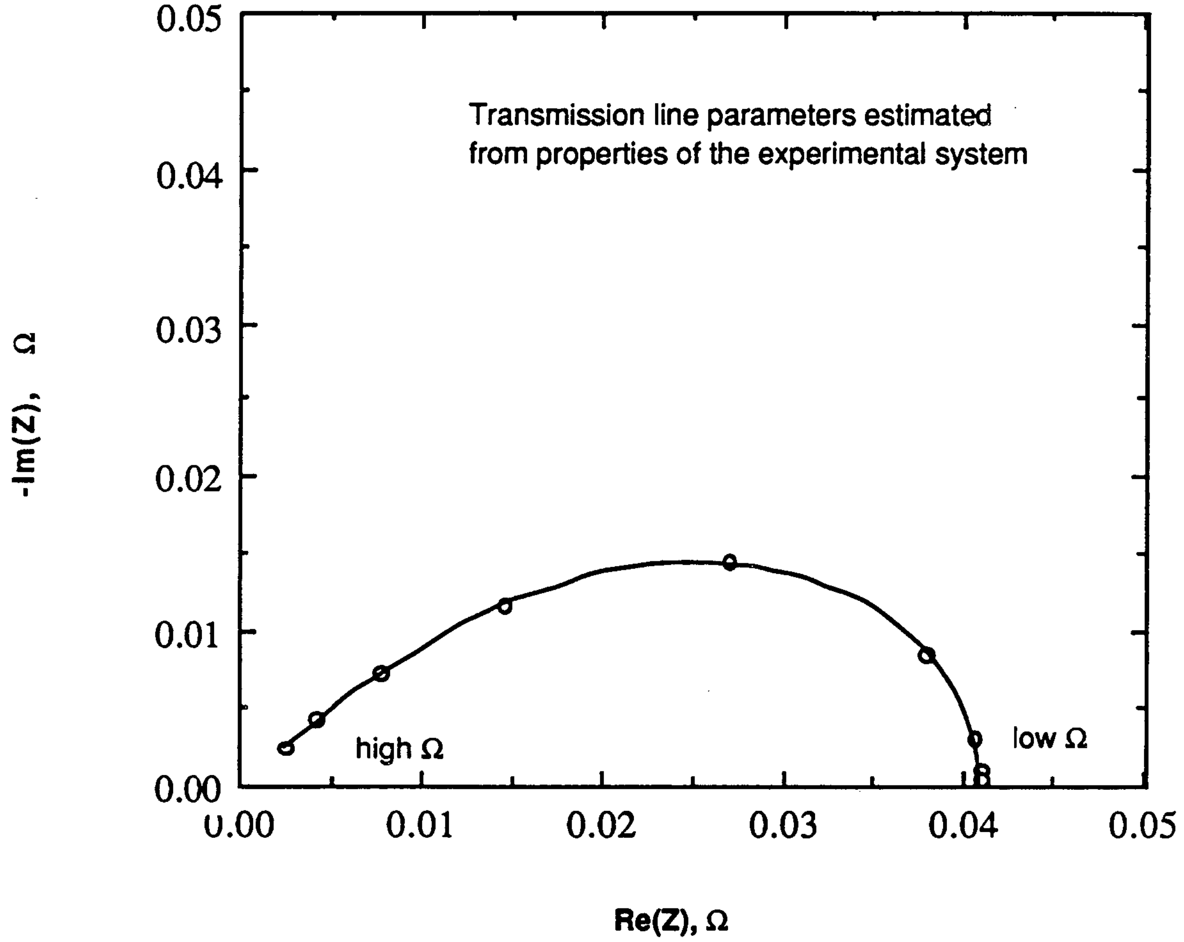


Figure 9-3. Nyquist plot of the impedance of a finite transmission line.

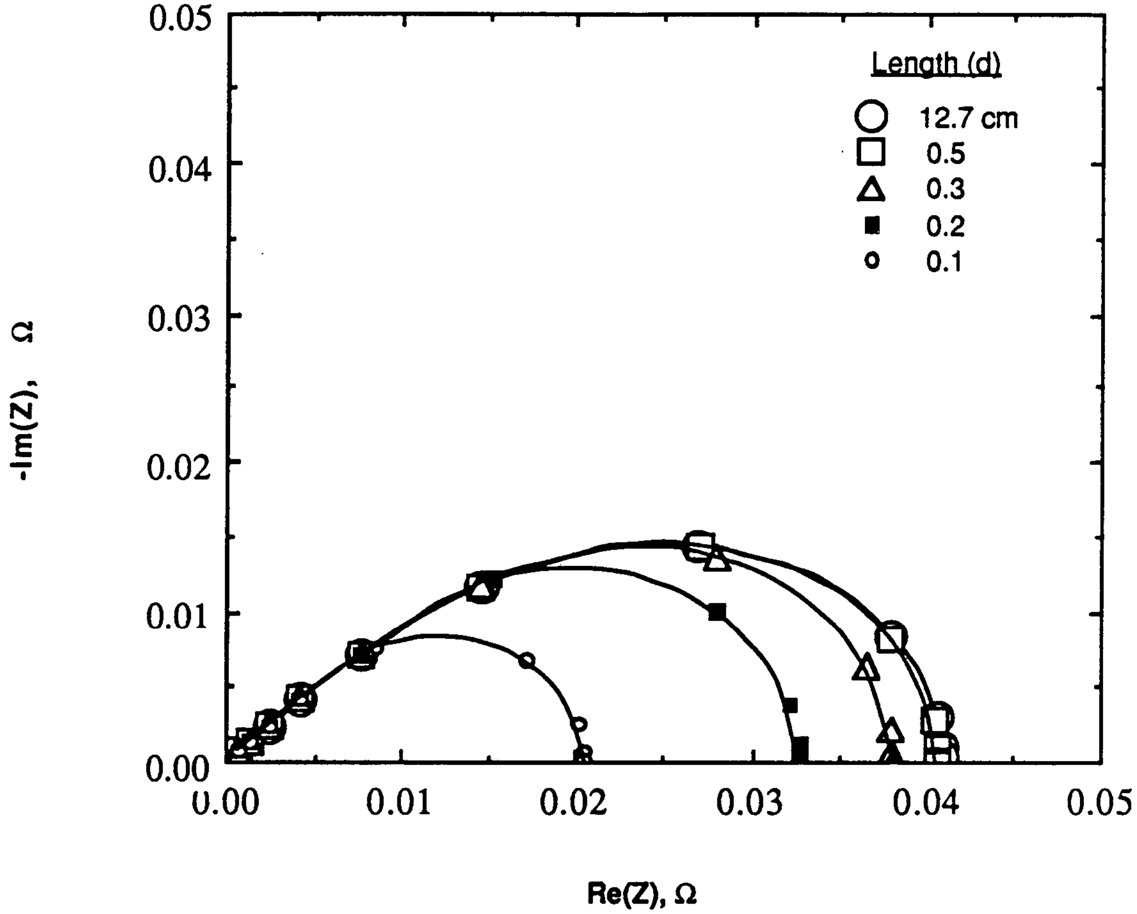


Figure 9-4. Nyquist plot of the impedance of a finite transmission lines of different lengths.

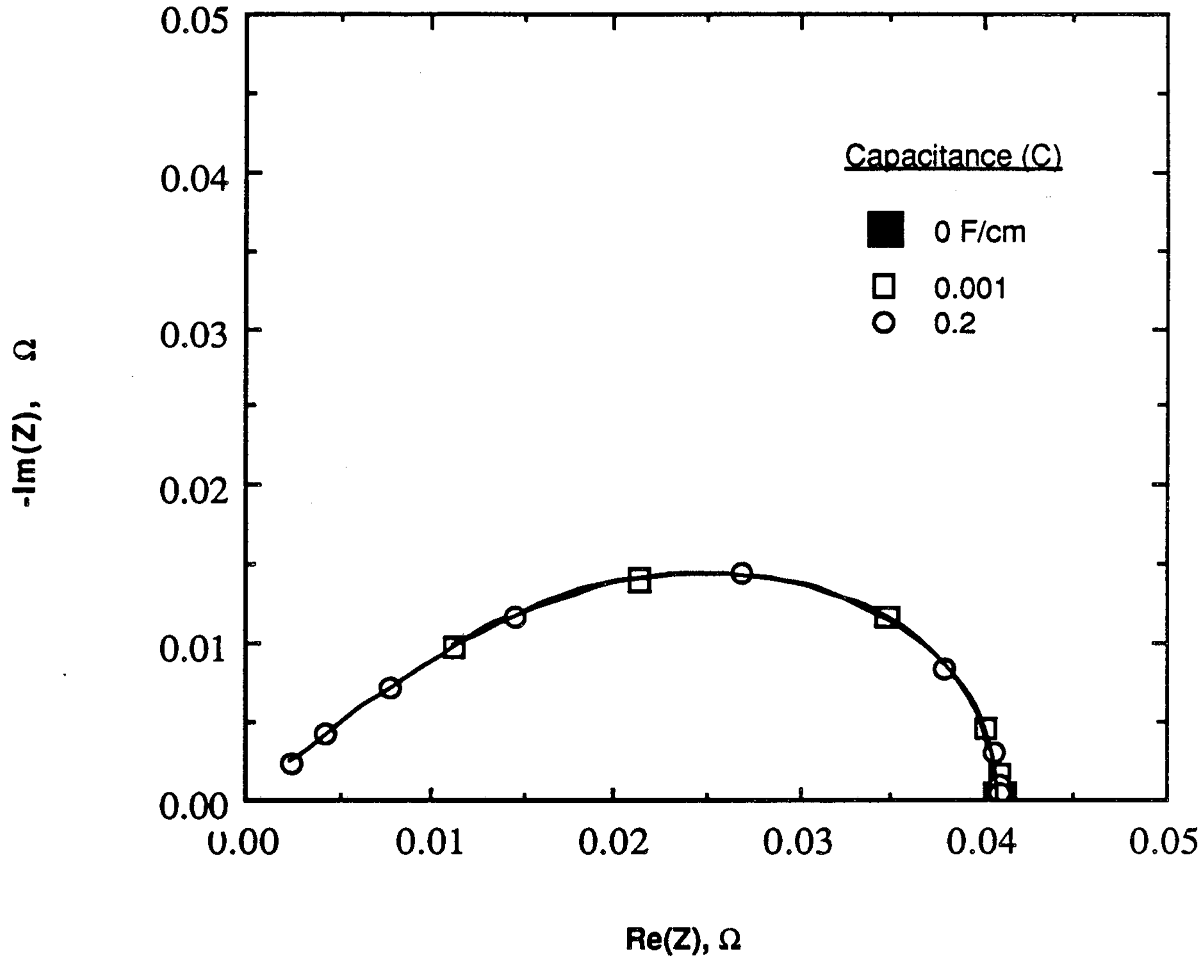


Figure 9-5. Nyquist plot of the impedance of transmission lines of different capacitances.

extend farther as the capacitance is increased. Different frequencies occur at a given point as the capacitance is changed.

Figure 9-6 shows the effect of changing the leakage in the transmission line. As expected, the impedance decreases as the leakage increases.

Figure 9-7 shows the effect of changing the conductivity of the transmission line. As expected, the impedance decreases as the resistance decreases.

Figure 9-8 shows the effect of changing the inductance of the transmission line. As expected, as the inductance increases, the curve extends farther into the negative complex half plane.

9.4. Model results

Possible sources of error in the experimental results include ground loops and shunt currents. The response of more than the electrode alone might have been measured because of the presence of other current paths. In addition the kinetic parameters used in the porous electrode model may not be the same for the reaction occurring on the porous electrode's surface as on that of the polished rotating disk. The value used for the mass transfer coefficient D/r_{pore} is also an approximation which may lead to error. Finally, the model does not predict, and the experimental results were not corrected for, the ohmic drop.

Calculations using the porous electrode model developed in Chapter 8 were made on a VAX 8650 computer. To solve the four equations at the 203 mesh points, with two iterations at each point, and a tolerance of 10^{-4} takes seven to twenty, but usually fifteen, minutes of CPU time for each frequency. A more efficient scheme should have been used to solve the equations. The data file used by the model is reproduced in Table 9-1.

To simulate the experiments, carried out at the open-circuit potential and without electrolyte flow, the steady-state values $\bar{\theta} = 1$ and $\bar{\eta} = 0$ were used. The curve labelled "0.25 M" in Figure 9-9 shows the model prediction of the impedance of the experimental system based on our best estimate for the physical properties of the experimental system. The measured impedance of the experimental system

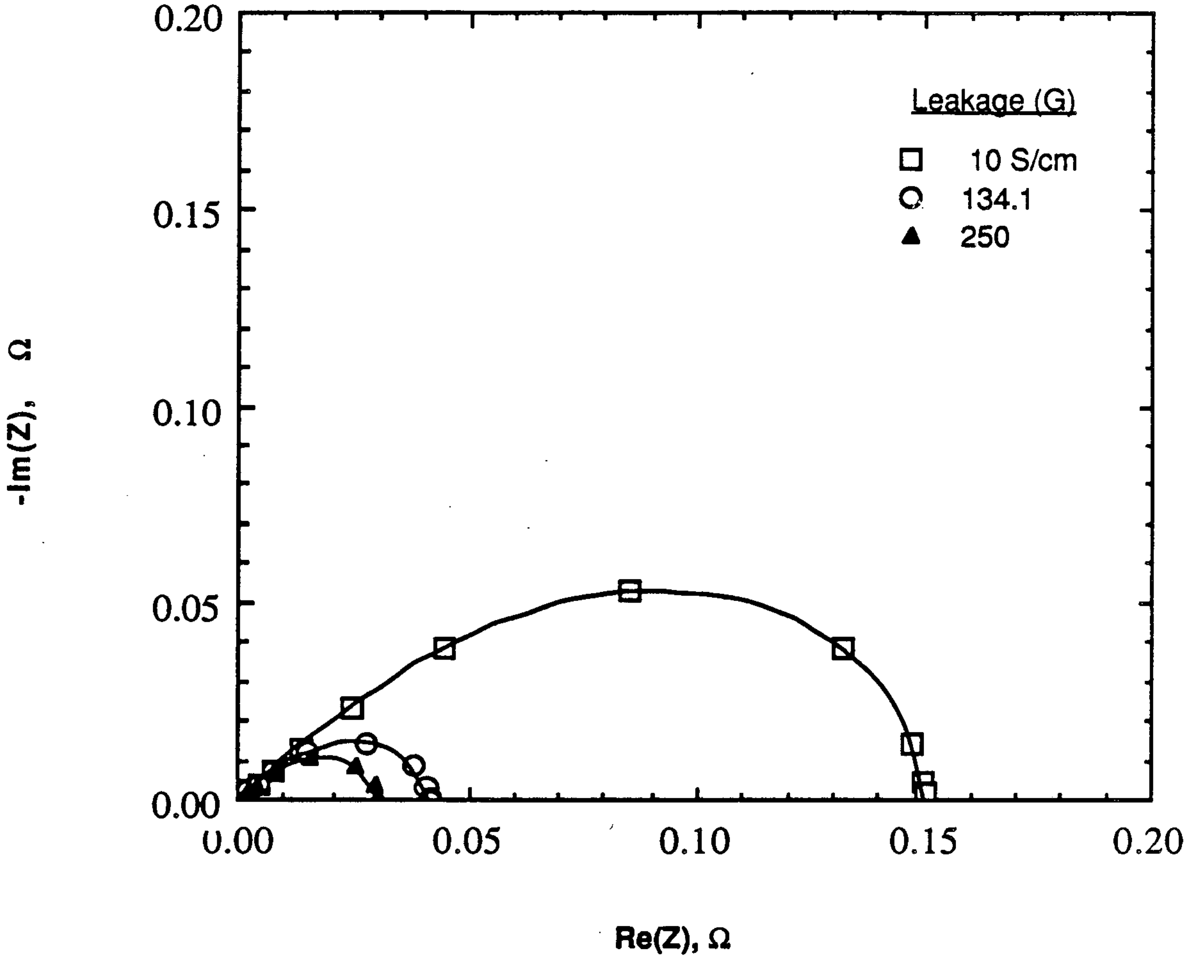


Figure 9-6. Nyquist plot of the impedance of transmission lines of different leakages.

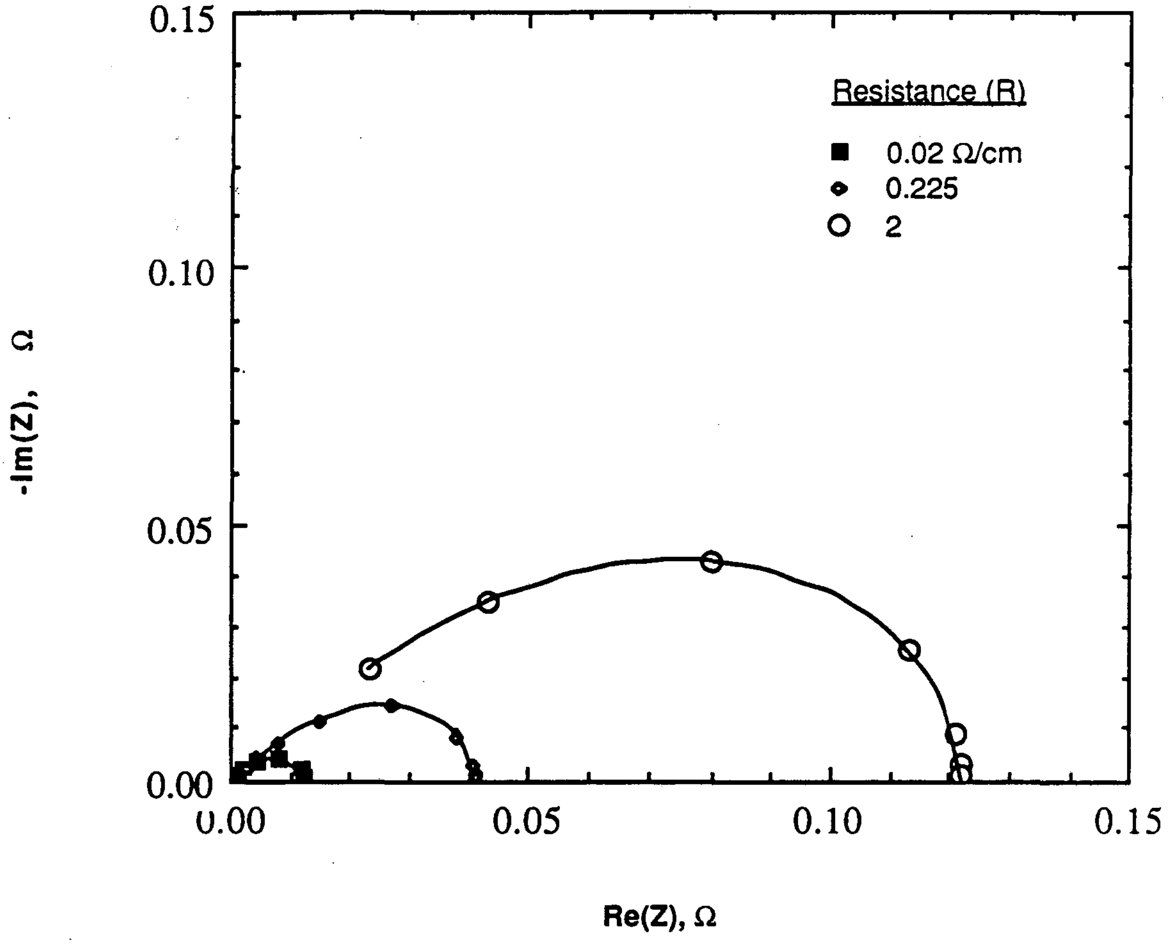


Figure 9-7. Nyquist plot of the impedance of transmission lines.

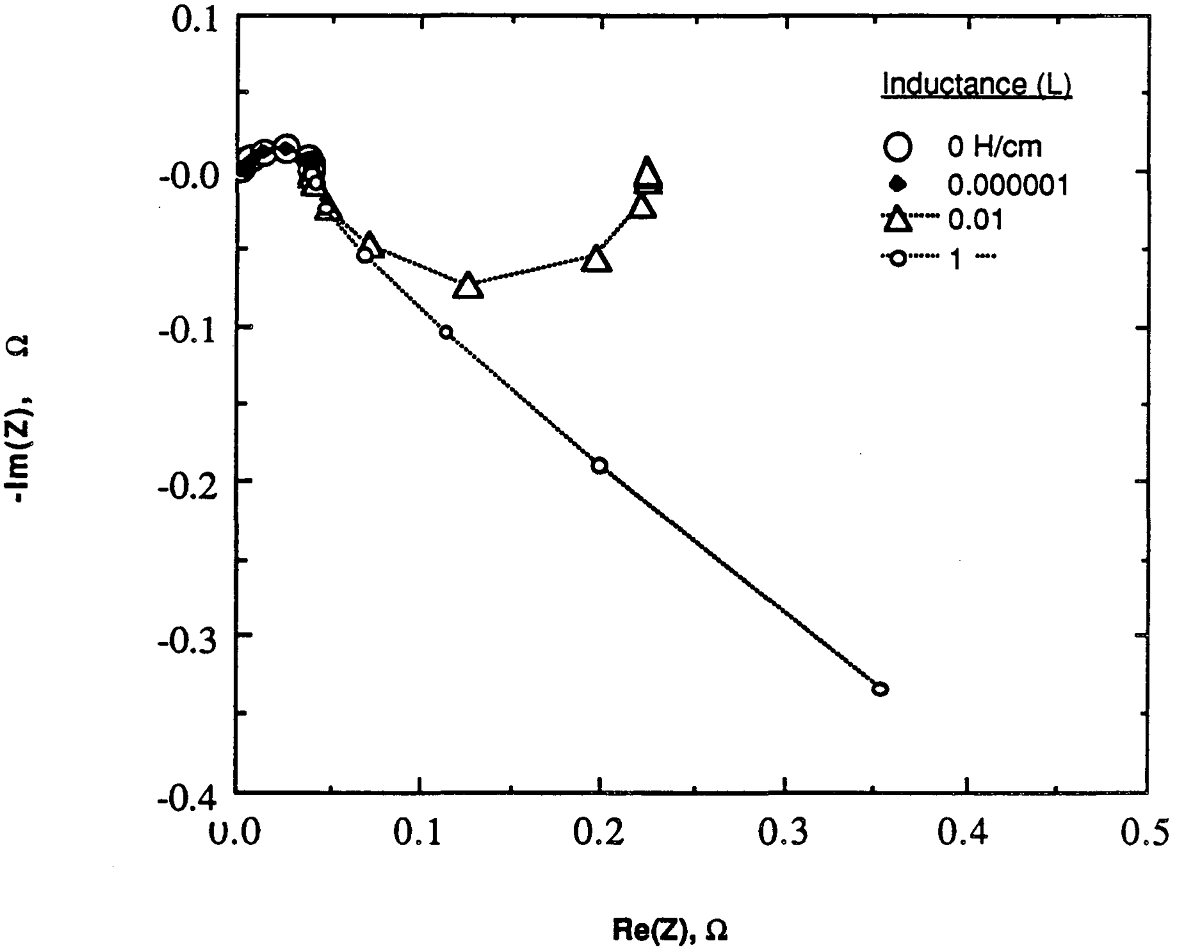


Figure 9-8. Nyquist plot of the impedance of transmission lines of different inductances.

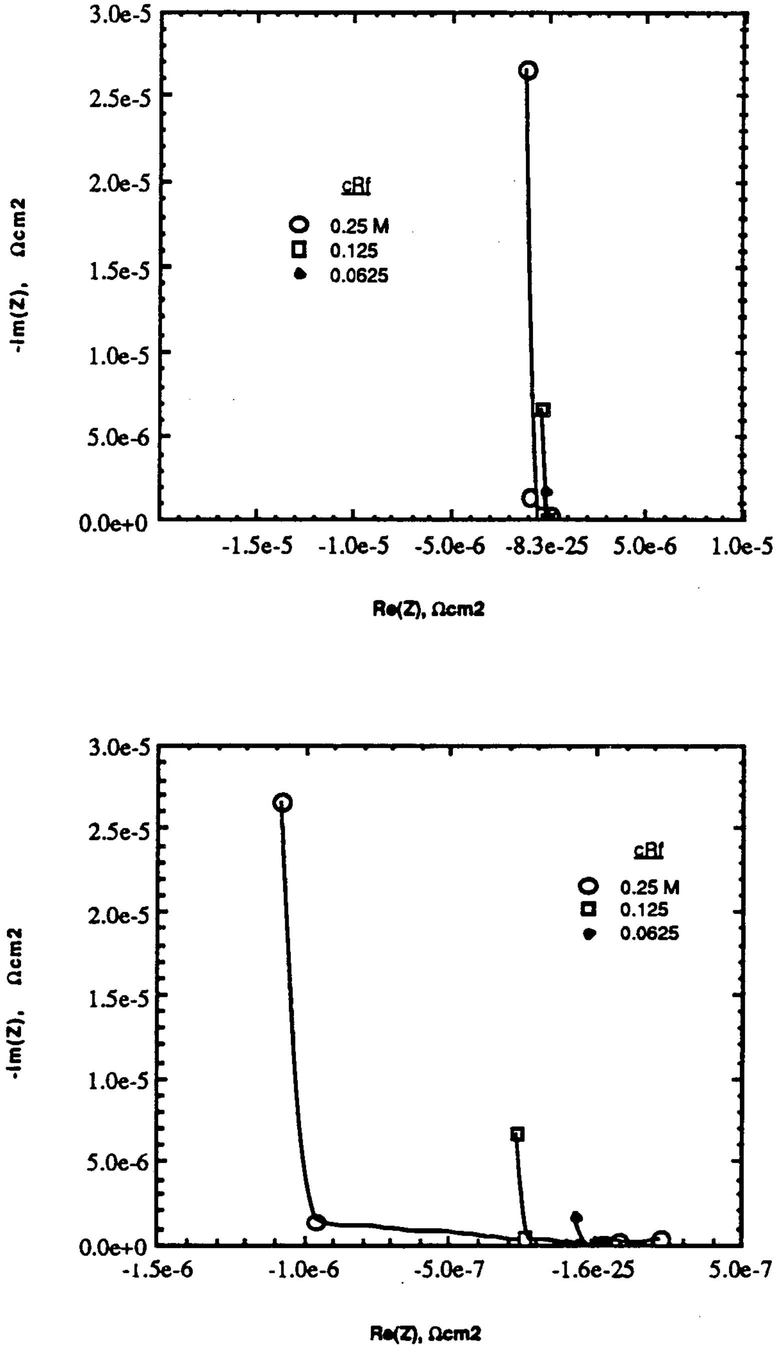


Figure 9-9. Model predictions for different electrolyte concentrations in a porous electrode. The maximum in a curve for an idealized porous electrode doubles when the concentration is reduced by a factor of four.

Number Of Mesh Points = 203
Iteration Limit = 10
Convergence Tolerance = 1e-4

Number of electrons in main reaction, n = 1
Electrode length = 1.27 cm
Temperature = 298.15 K
Reactant diffusivity = 6.2e-6 sq cm/s

Electrode void fraction = 0.97
Electrode surface area = 1000.0 sq cm/cu cm
Fluid velocity = 0.0e-4 cm/s
Solution conductivity = 0.25 mho/cm
Electrode conductivity = 1.73 mho/cm
Double-layer capacity = 10.0e-6 F/sq cm

Stoichiometric coefficient of the reactant, sR = -1.0
Reactant feed concentration = 0.25 M
Product feed concentration = 0.25 M
Pore radius = 1.00e-1 cm

Kinetic parameters:
Main reaction
anodic alpha = 0.74
cathodic alpha = 0.26
Side reaction
anodic alpha = 0.5
cathodic alpha = 0.5
Exchange current densities at the reference (i.e. feed) composition:
main reaction = 0.227e-3 amps/sq cm
side reaction = 6.708e-13 amps/sq cm

Delta U (Difference in the potentials of the side reaction and the primary reaction at the reference (feed) composition) = -0.6513 volts

Current = -0.0 amp/sq cm
Current oscillation amplitude = 0.14 amp/sq cm

Table 9-1. Parameters used by the model to simulate experimental results.

is not only much greater than the model's prediction, but is qualitatively different. The experimental result is a loop with a width of about 1.75 Ω or 35 Ωcm^2; and a height of 0.8 Ω or 16 Ωcm^2. Because the model predicts that the system behaves like a capacitor, it can be assumed that the double-layer capacity is less than the estimated 10 μF/cm^2 that was used in the model.

If the squares of the real and imaginary parts of the impedance of an idealized porous electrode are plotted (rather than the usual Nyquist plot), de Levie[8] showed that the resulting curve will be a semicircle. His experimental results agree with this, but his electrode was closer to the ideal than ours. Neither the squares of the porous electrode model's results shown below nor the square of the impedance of a transmission line are semicircles. However the two curves have the same shape.

The product of the radius of the semicircle in the squared impedance plane and the electrolyte concentration is a constant.[8] Therefore doubling the concentration should reduce the curve maximum in the impedance plane by a factor of $\sqrt{2}$. Figure 9-9 also shows the model's calculations of Nyquist plots for 0.25 M, 0.125 M, and 0.0625 M Fe^{3+}/Fe^{2+} solutions.

It is more practical to change the values of a few dimensionless groups than many individual variables to observe the effect of the changes on the model results. Table 9-2 contains the values of the dimensionless groups for the base case. These values were changed each time the program was run. For these simulations, the potential $\bar{\eta}'$ was assigned the value $-\dfrac{nFk_{mR}c_{Rf}}{sRi_{oR,ref}}$ instead of the natural logarithm of

P_1	1.67e+7
P_2	-6.93e-5
P_3	5.26e-14
P_4	7.26e+7
P_5	6.05e-5
P_6	8.75e-6
P_7	1.0
θ_{Pf}	1.0
D'	97.0
αL	1.27e+3
I^*	0.0
C_{dl}	-4.223e-5

Table 9-2. Values of the dimensionless parameters for the base case of the impedance program.

this value. This is equivalent to using a value of $\bar{\eta}$ of 0.46 V. The value assigned to the concentration $\bar{\theta}$ was unity which is the same as assuming that the measurements are sufficiently fast that they are finished before the reaction has proceeded appreciably.

Figures 9-9 through 9-17 show the effects of changing the values of the dimensionless groups listed in Table 9-2.

Figures 9-10 through 9-12 illustrate the effect of changing the importance of the faradaic processes. Figure 9-10 shows that change P_1, the importance of the backward term of the main reaction, by fourteen orders of magnitude causes little change in the curve under the assumed conditions. On the other hand, Figure 9-11 shows that decreasing P_4, the importance of the backward term in the side reaction, by fourteen orders of magnitude reduces the amount of current that will be passed for a given applied potential and thus increases the magnitude of the impedance. This same logic explains the difference between the two curves in Figure 9-12. Increasing P_3, the rate of the side reaction, decreases the impedance because charge transfer is easier.

The impedance should decrease with increasing bed length because there are more paths for the current to take between the two phases. Figure 9-13 shows that the model predicts this.

Figures 9-14 through 9-16 show the effects of changing the importance of the ohmic terms. Figure 9-14 shows that if the ohmic drop is very important, *i.e.*, if P_2 is very large, the electrode acts like a resistor and the Nyquist curve lies at a point on the real axis. If the potential drop or resistance in the matrix P_6 is very large, then the impedance of the electrode is very large as seen in Figure 9-15. The most interesting result, especially when compared with the experimental results in Figure 9-2, is that shown in Figure 9-16. Here the curve crosses the imaginary axis, and the impedance is large at most frequencies for the larger value of P_5 the ohmic drop or resistance in the solution phase.

In figure 9-17 the curve for the large value of the double layer capacity is nearly vertical because the bed is acting as a capacitor. It is interesting to note that if the wrong sign is used for the

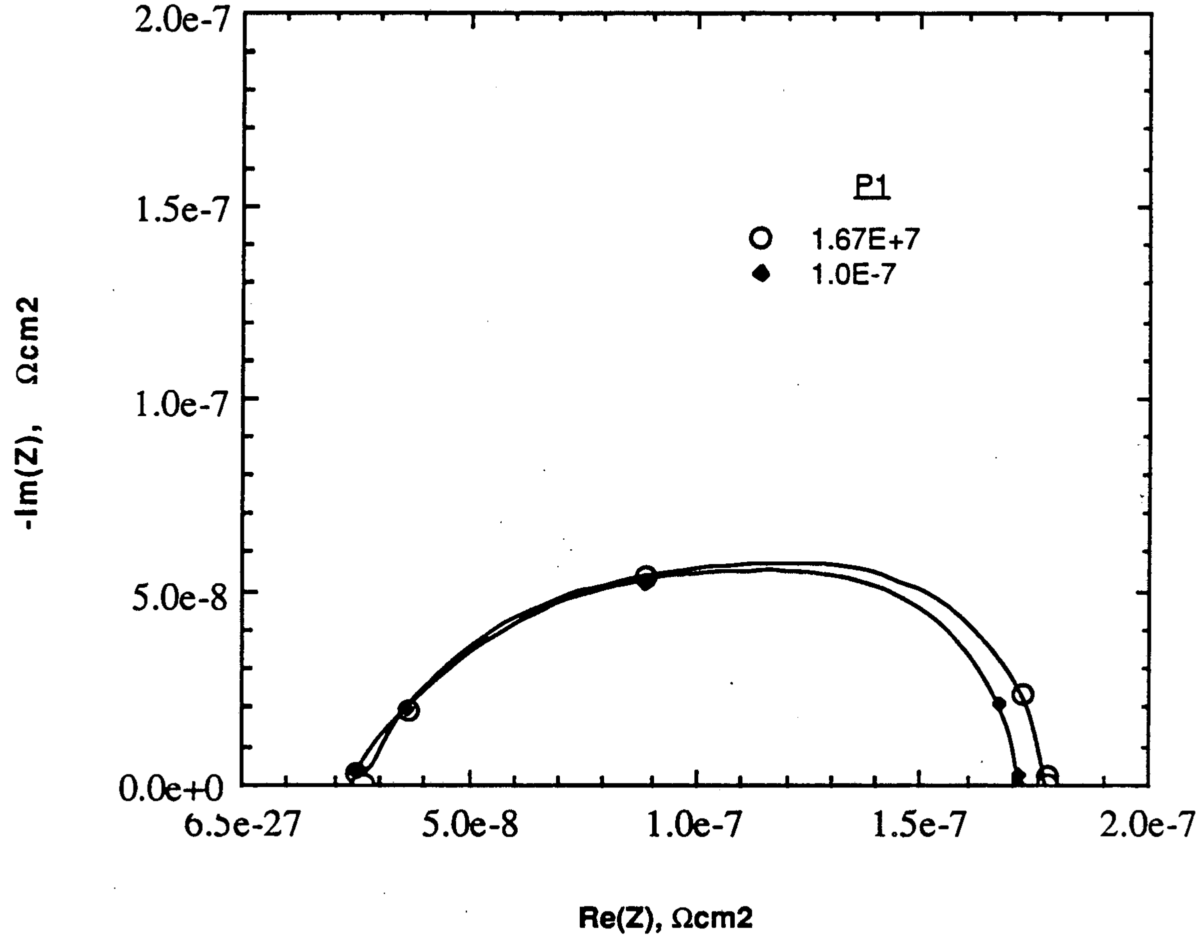


Figure 9-10. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. P_1 characterizes the importance of the main reaction's rate.

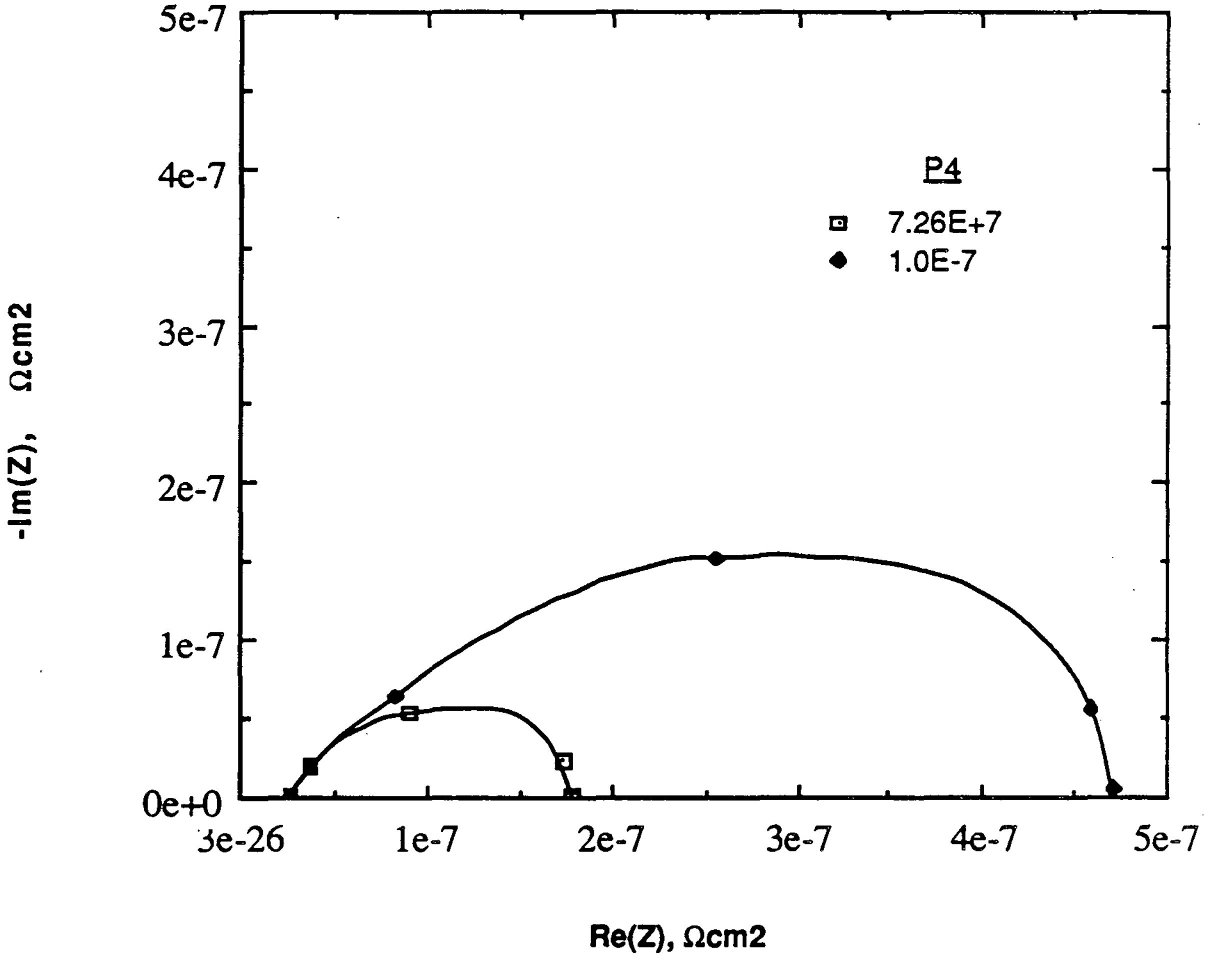


Figure 9-11. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. P_4 characterizes the importance of the cathodic term of the side reaction.

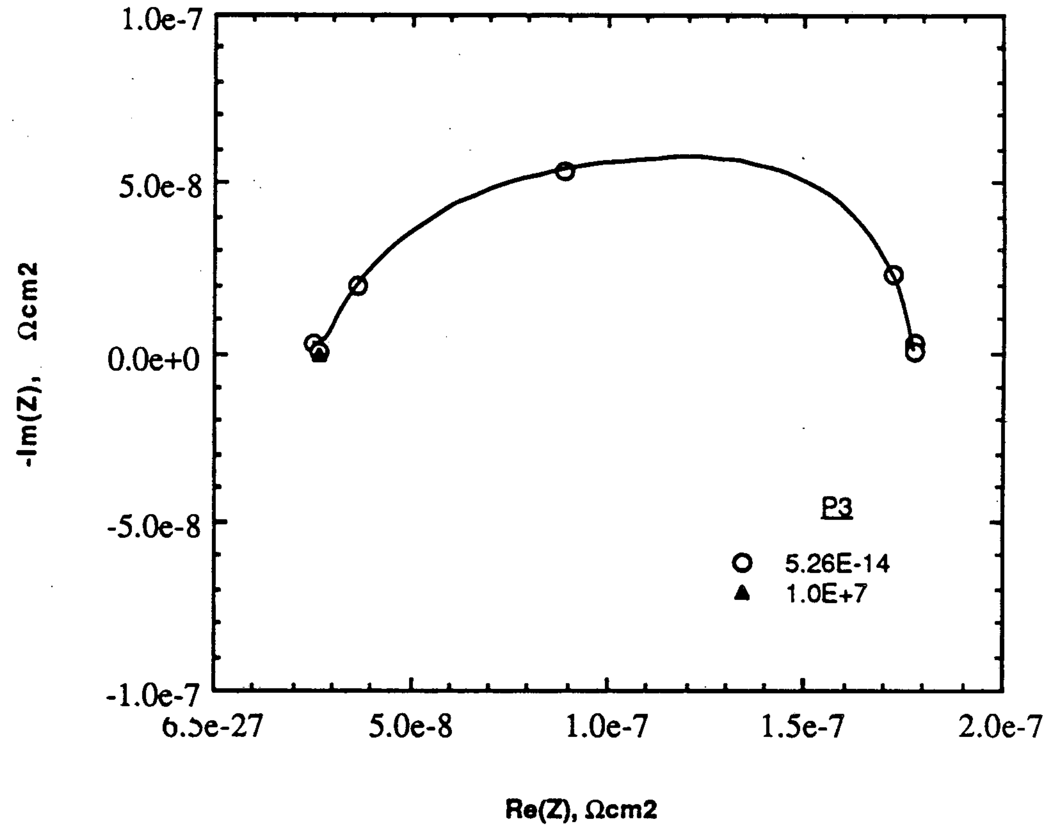


Figure 9-12. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. P_3 characterizes the importance of the side reaction.

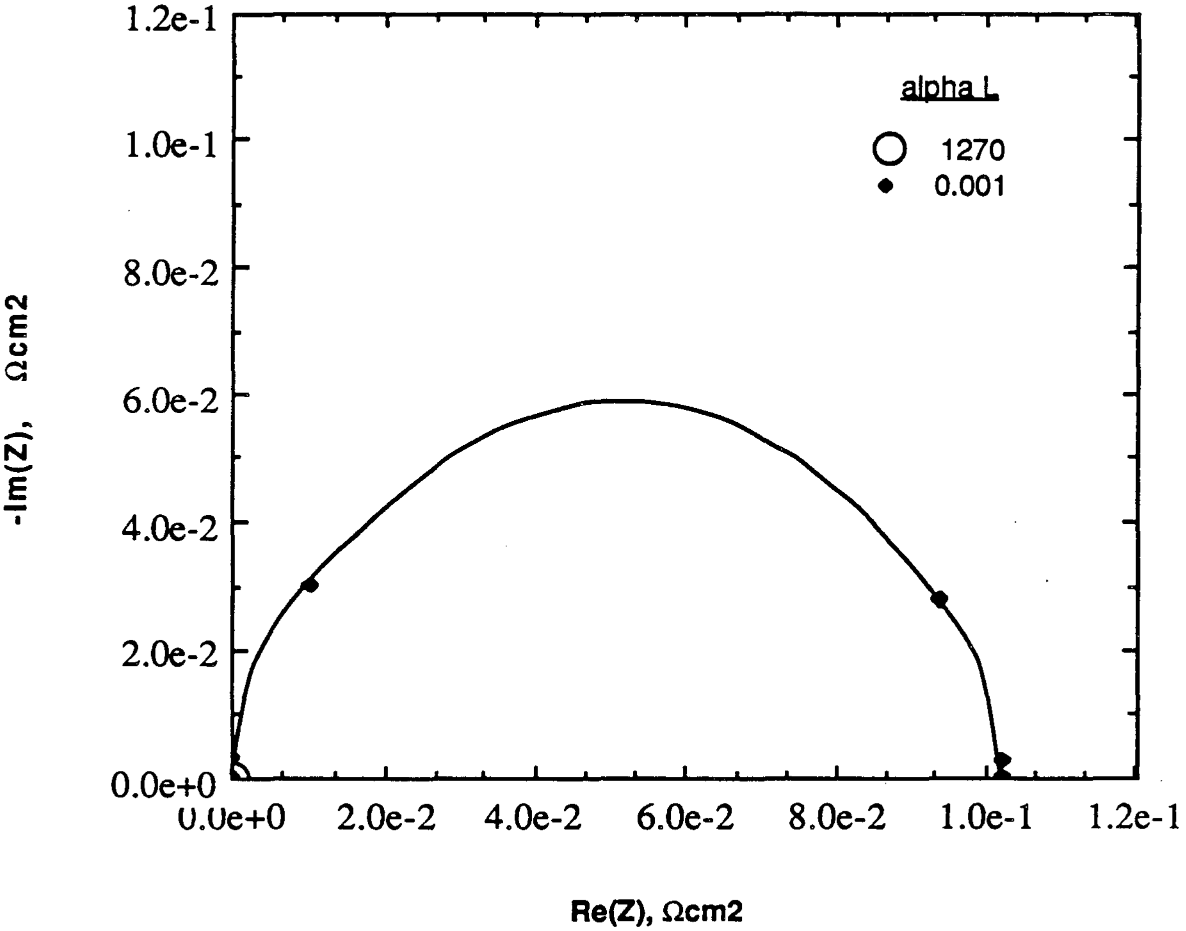


Figure 9-13. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. αL is the dimensionless bed length.

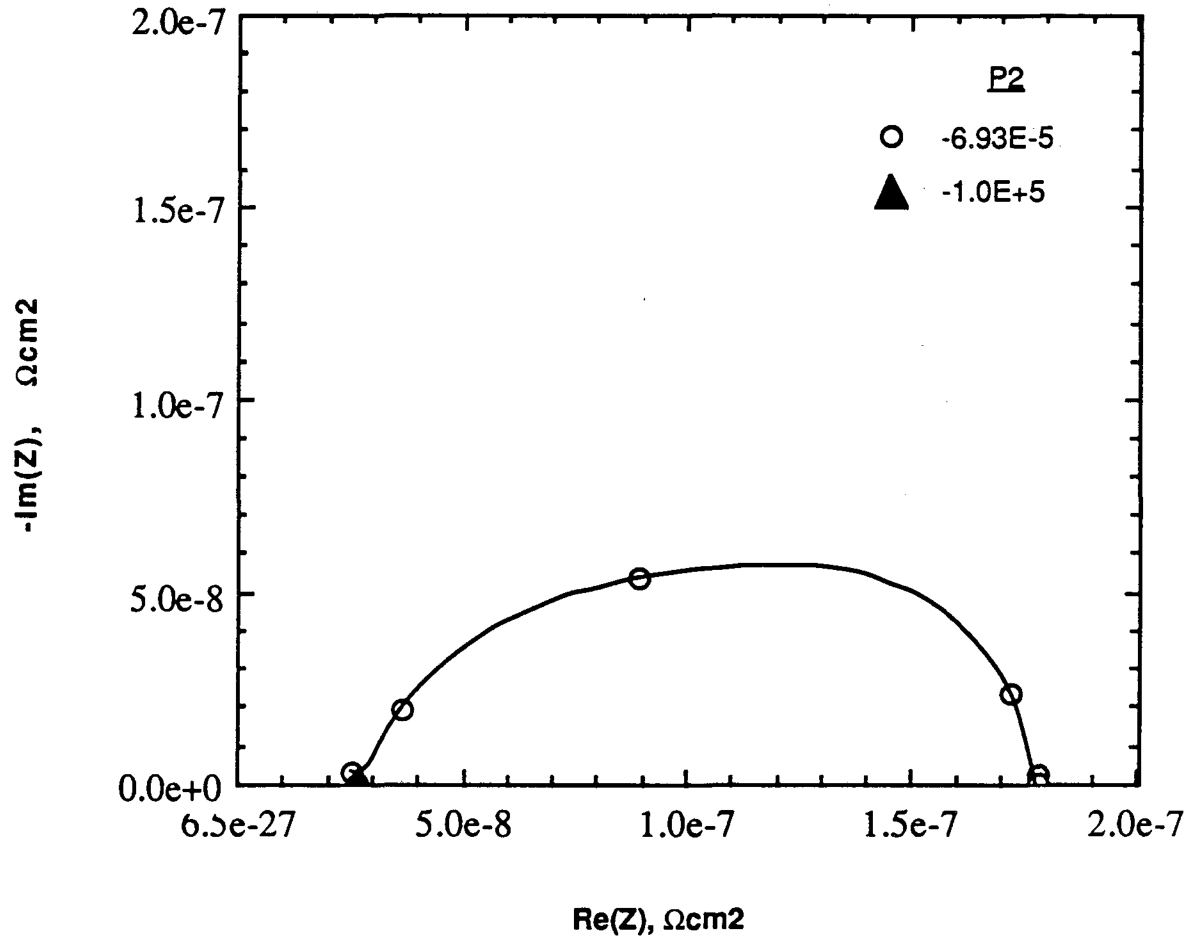


Figure 9-14. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. P_2 characterizes the importance of the ohmic potential drop.

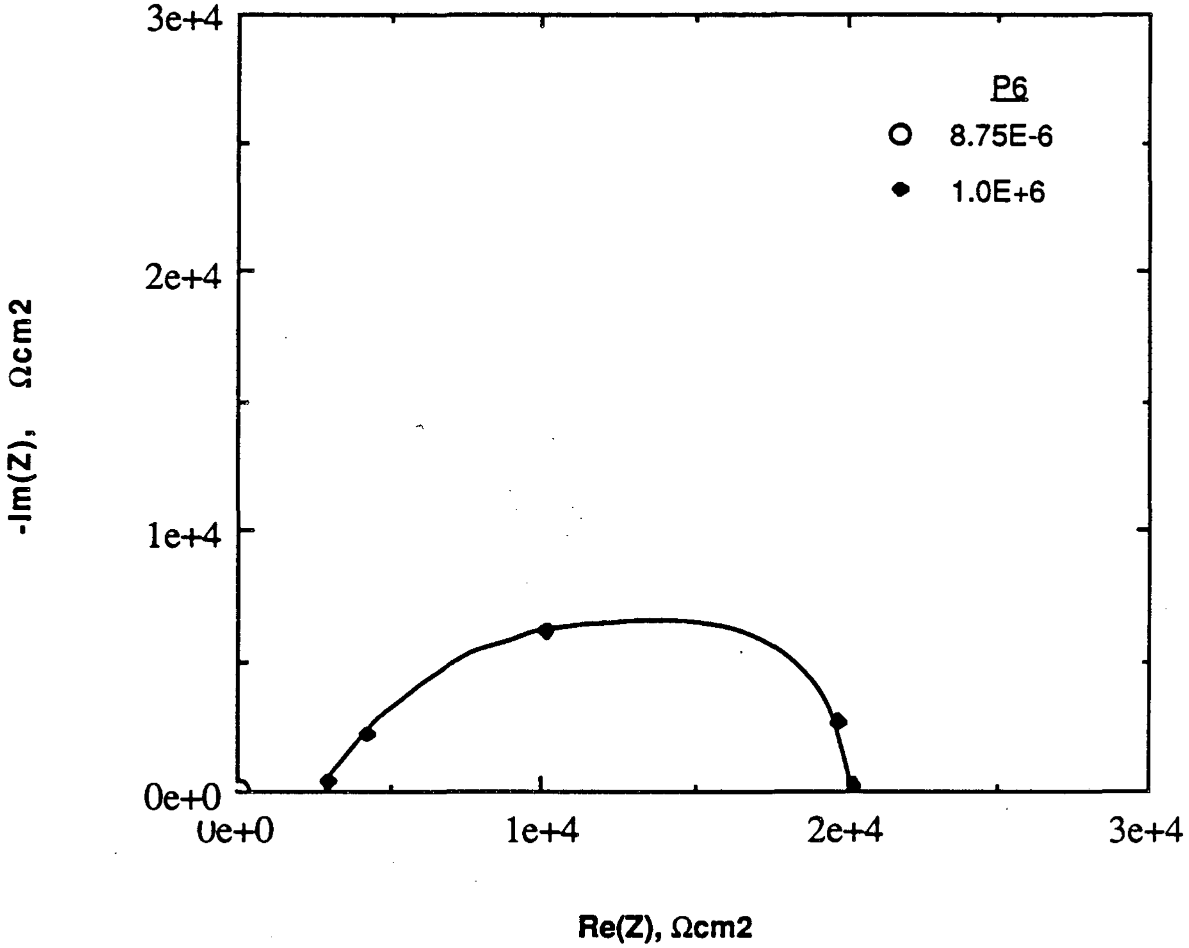


Figure 9-15. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. P_6 characterizes the importance of the ohmic potential drop in the matrix phase.

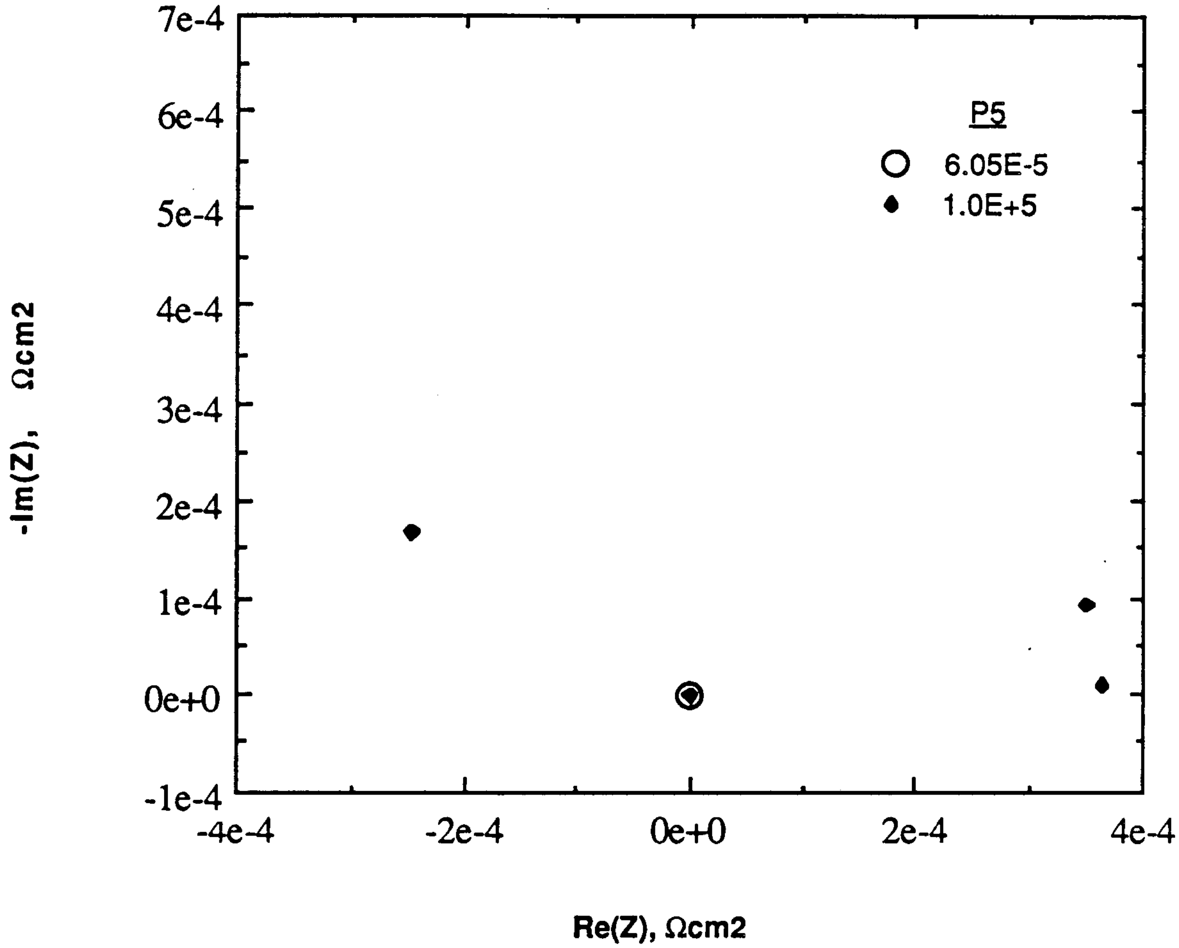


Figure 9-16. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. P_5 characterizes the importance of the ohmic potential drop in the solution phase.

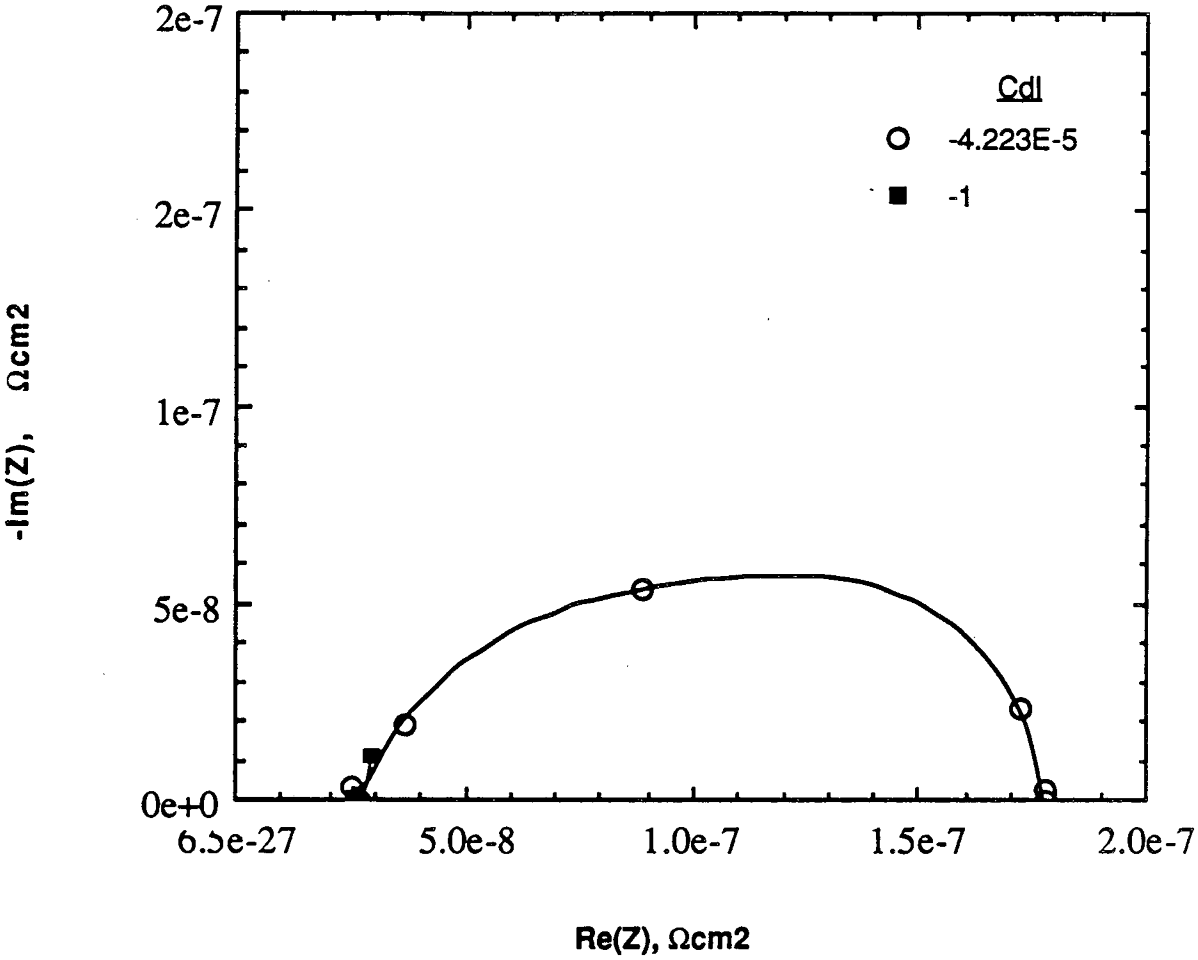


Figure 9-17. Model predictions for the impedance of a porous electrode. All other parameters are as listed in Table 9-2. C_{dl} is the dimensionless double-layer capacity.

double-layer capacity, the loop is inverted, *i.e.*, it appears below rather than above the real axis. This is what intuition would suggest because the out-of-phase or imaginary part of the impedance is usually proportional to C^{-1} so changing the sign of C changes the sign of Im(Z).

The dimensionless groups are related to properties of the experimental system and can be changed in the experimental system by changing the system appropriately. The dimensionless group P_1 is proportional to the experimental variable $i_{oR,ref}^{1+\alpha_{aR}/\alpha_{cR}}$; P_2, $\frac{1}{\kappa}+\frac{1}{\sigma}$; P_3, $i_{oS,ref}\exp(\alpha_{cS}F\Delta U/RT)$; P_4, $\exp[-F(\alpha_{aS}+\alpha_{cS})\Delta U/RT]$; P_5, $1/\kappa$; P_6, $1/\sigma$; P_7, k_{mR}/k_{mP}; θ_{Pf}, c_{Pf}/c_{Rf}; D', $(D_R+D_a)ak_{mR}\varepsilon/v^2$; αL, $ak_{mR}L/v$; I^*, i; and C_{dl}, C.

9.5. Conclusions

The model works well in general, but not for the particular case of zero velocity. Its results change reasonably as the input parameters are changed.

Before the model is used to look at concentration and potential profiles and how they are affected by experimental conditions, it would be wise to make the programs more time efficient. Some changes that would reduce the amount of CPU time used would be to use single precision, to reduce the number of function calls, to linearize the equations, and to use directly Newman's BAND program rather than BandAid to solve the equations.

References

1. Michael John Matlosz, *Experimental Methods and Software Tools for the Analysis of* Electrochemical Systems, Ph.D. dissertation, University of California, Berkeley, pp. 7-52 (1988) LBL-19375.

2. Michael Matlosz and John Newman, "Experimental Investigation of a Porous Carbon Electrode for the Removal of Mercury from Contaminated Brine," *J. Electrochem. Soc.*, 133, 1850 (1986).

3. G. G. Trost, *Applications of Porous Electrodes to Metal-Ion Removal and the Design of Battery Systems*, Ph.D. Dissertation, University of California, Berkeley (1983) LBL-16852.

4. G. Sisler, *Adsorption of Dilute-Aqueous Zinc Ions in the Electrical Double* Layer of a Porous-Carbon Electrode, M.S. Thesis, University of California, Berkeley (1987) LBL-24406.

5. Trost, *op. cit.*, pp. 50-60.

6. C. Haili, *The Corrosion of Iron Rotating Hemispheres in 1 M Sulfuric Acid:* An Electrochemical Impedance Study, M.S. Thesis, University of California, Berkeley, p. 184 (1987) LBL-23776.

7. David K. Cheng, *Analysis of Linear Systems*, pp. 348-380, Addison-Wesley, Reading, Mass. (1959).

8. R. de Levie, "On Porous Electrodes in Electrolyte Solutions—IV," *Electrochim. Acta*, 9, p. 1240 (1964).

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
INFORMATION RESOURCES DEPARTMENT
BERKELEY, CALIFORNIA 94720