Copyright © 1988, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

MACROMODELLING FOR THE SIMULATION OF LARGE SCALE ANALOG INTEGRATED CIRCUITS

by

Giorgio Casinovi

Memorandum No. UCB/ERL M88/55

16 August 1988

COVER IMAGE

MACROMODELLING FOR THE SIMULATION OF LARGE SCALE ANALOG INTEGRATED CIRCUITS

by

Giorgio Casinovi

Memorandum No. UCB/ERL M88/55

16 August 1988

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

TITLE PAGE

MACROMODELLING FOR THE SIMULATION OF LARGE SCALE ANALOG INTEGRATED CIRCUITS

by

Giorgio Casinovi

Memorandum No. UCB/ERL M88/55

16 August 1988

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Macromodelling for the Simulation of Large Scale Analog Integrated Circuits

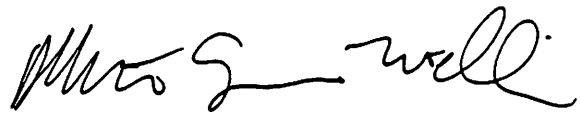
Giorgio Casinovi

Ph.D.

Department of Electrical Engineering
and Computer Science

Abstract

Circuit simulation is one of the most valuable tools available to the circuit designer, providing him with accurate information about the electrical behavior of a particular circuit without the need of building it. Unfortunately the CPU time required to perform each simulation grows very quickly with the circuit size, making it problematic to simulate circuits containing more than a few hundred nodes. One of the methods suggested to overcome this hurdle is the use of macromodels for functional blocks such as operational amplifiers or logic gates. All the methods to generate macromodels that have appeared in the literature so far make very stringent assumptions about either the type of circuits to be modeled or the shape of the waveforms involved. The models generated in this way are therefore unsuitable for circuit simulation. This thesis describes a general purpose algorithm that can be used to generate models for an arbitrary circuit without any restrictions on the waveforms. Experimental results indicate that the algorithm requires reasonable amounts of CPU time and that the models are accurate enough to be used for standard circuit simulation.



Alberto Sangiovanni-Vincentelli
Thesis Committee Chairman

Acknowledgements

I would like to thank my research adviser, Professor Alberto L. Sangiovanni-Vincentelli, for suggesting the topic of this dissertation; his outstanding work in the area of Computer-Aided Design of Integrated Circuits will always be an example for me to look up to. I am likewise grateful to the other members of my thesis committee, Professors Charles A. Desoer and Beresford N. Parlett, for their valuable comments and suggestions. I am indebted to Professor Henry Helson of the Department of Mathematics for suggesting the outline of the proof of Theorem 4.1. I also appreciate the efforts of Professors Donald O. Pederson and Richard Newton in creating a stimulating and pleasant research environment.

While I was visiting the Massachusetts Institute of Technology I held several helpful discussions with Professors Sanjoy K. Mitter and Dimitri P. Bertsekas and with Tom Richardson. I would also like to thank Messrs. Hermann K. Gummel, Ajoy K. Bose, Hao N. Nham and Chin F. Chen of AT&T Bell Laboratories, where I started to work on this research project.

Financial support received from JSEP under Grant F49620-87-C-0041 is gratefully acknowledged.

The computer implementation of the algorithm would not have been possible without the contribution of several fellow students : Wayne Christopher and Tom Quarles, who wrote the user front-end, and Ken Kundert, who wrote the sparse matrix package. The front-end to simulator interface is due to Tom Quarles, Ken Kundert, Don Webber, Nick Weiner and Tom Laidig. Don Webber's program RELAX3 was the model for

some of the code that I wrote. Many other of my officemates helped me with general advice and support.

I am very grateful to Don Webber for his careful proofreading of a preliminary draft of this dissertation. Finally, I would like to express my appreciation for the work of Brad Krebs and Kurt J. Pires as system administrators of our computers.

Contents

1 The Macromodeling Problem **1**

1.1 Introduction 1

1.2 Previous Work In Circuit Modeling 3

1.3 Previous Work In Model Order Reduction 7

1.4 Our Approach 10

2 Mathematical Preliminaries **15**

2.1 Topological Spaces 15

2.2 Derivatives in Banach Spaces 18

2.2.1 Banach Spaces 18

2.2.2 Duals of Banach Spaces 19

2.2.3 The Frechet Derivative 20

2.2.4 The Gateaux Derivative 21

2.3 Function Spaces 22

2.4 Differential Equations 25

2.5 Optimal Control 29

2.5.1 Problem Formulation 29

2.5.2 The Lagrange Multipliers 30

2.5.3 Pontryagin's Minimum Principle 33

3 The Circuit Equations **35**

3.1 Preliminary Assumptions 35

3.2 Charge Formulation of the Node Equations 36

3.3 Numerical Integration of the Circuit Equations 42

3.3.1 General Properties of Numerical Integration Methods 42

3.3.2 Linear Multistep Methods 43

3.3.3 Backward Differentiation Formulae 49

3.4 The Adjoint Network 51

4 Theoretical Framework **59**

4.1 Circuits as Dynamical Systems 60

4.2 Macromodeling as a Min-Max Problem 62

4.3 The Existence of an Optimal Model 65

4.4 Computation of Derivatives 72

4.5 Application to Circuit Equations 79

5 Saddle Point Algorithm **84**

5.1 A Conceptual Algorithm 85

5.2 A Modification of the Steepest Descent Algorithm 88

5.3 A Variable Metric Algorithm 89

5.4 Broyden's Method 93

6 Computer Implementation **97**

6.1 Input Parametrization 97

6.2 Experimental Results 100

6.2.1 Distributed RC Line 100

6.3 Conclusions 112

Chapter 1

The Macromodeling Problem

1.1 Introduction

Circuit simulation is one of the most valuable tools available to the circuit designer, providing him with accurate information about the electrical behavior of a particular circuit without the need of building it. Unfortunately the CPU time required to perform each simulation grows very quickly with the circuit size, making it problematic to simulate circuits containing more than a few hundred nodes. Several methods have been suggested to overcome this hurdle, the most successful of which is the use of relaxation techniques [1]. However their effectiveness is mostly confined to circuits that do not contain tight feedback loops (typically digital circuits), and the presence of a capacitance from each node to ground is usually required to ensure convergence.

We will show that another solution to this problem can be found by exploiting the modular structure exhibited by most circuits. By "modular structure" we refer to a particular design style, in which several instances of a limited number of functional

blocks are connected together to build a circuit meeting the design specifications. This technique is widely used in the design of analog, digital and mixed integrated circuits, because it reduces dramatically the design time at every stage, from synthesis to layout.

When performing a simulation it is often the case that the designer is not interested in the waveforms that are internal to a block, but only in those relative to nodes or branches connecting two or more different blocks. In other words, it is the input-output behavior of each block that matters, not its internal dynamics. In the case of electrical circuits, a block can always be described by a dynamical system whose order is related to the number of nodes contained in that block. If we manage to find another system of lower order but exhibiting the same input-output behavior, we can use it as a model for every instance of that block occurring in the circuit that we want to simulate. In this way, we can reduce the number of differential equations that must be solved to simulate the circuit, at the same time retaining all the relevant information.

In practice it is not possible to preserve the input-output relationship exactly, and we must settle for a reduced order model that approximates the original block within certain margins. This is perfectly acceptable, because the actual performance of an integrated circuit after fabrication cannot be predicted with total accuracy, due to the presence of many factors whose influence on the behavior of the physical chip cannot be controlled or predicted.

The approaches to the macromodeling problem that have been published so far are very varied, ranging from empirical methods suitable for a specific application to sophisticated mathematical techniques applicable to broad classes of systems. After reviewing briefly the existing literature on this topic, I will point out the advantages and

disadvantages of the methods proposed by others, and motivate my approach, which, from a mathematical point of view, can be described as an approximation problem in an abstract space whose elements are dynamical systems.

1.2 Previous Work In Circuit Modeling

The advantages of using simplified models of functional blocks for special purposes have been pointed out. Many timing verifiers model MOS transistors as switched resistors. The resistor value is determined empirically to take several factors into account, including the shape of the input waveform; this is the approach followed for instance in Crystal [2]. This drastic simplification reduces the problem of computing the delay through a circuit containing MOS transistors to the much easier task of estimating delays through an RC network. Rubinstein, Penfield and Horowitz [3] gave an algorithm that could be used to compute upper and lower bounds for the delays in an RC tree. A slightly different approach was used by Lin and Mead [4], who started from Elmore's definition of delay [5] : let $y(t)$ be the step response at some node of a linear network. Then the delay at that node is defined as :

$$T_D = \int_0^\infty t y'(t)\, dt$$

(see Fig. 1.1). This definition is very close to the intuitive notion of "delay" of a monotonically increasing (or decreasing) waveform. It has also the big advantage of being easy to compute: under fairly mild assumptions, it can be shown that $T_D = b_1 - a_1$, where a_1 and b_1 appear in the Laplace transform $\hat{y}(s)$ of $y(t)$:

$$\hat{y}(s) = \frac{1 + a_1 s + \ldots + a_m s^m}{s(1 + b_1 s + \ldots + b_n s^n)}$$

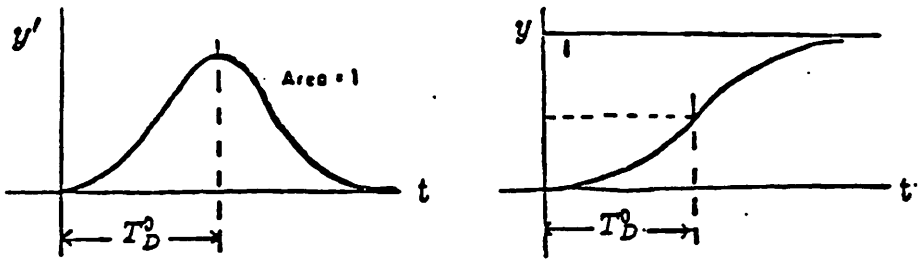


Figure 1.1: Elmore's definition of delay

Lin and Mead give a slightly more general definition of delay and an algorithm that can be used to compute the delay for any node in a general RC network.

Although very useful for the particular purpose of timing analysis (Crystal's accuracy is reportedly within 10% of SPICE results), this approach has its obvious limitations. Matson [6] notes that the delay introduced by an MOS gate varies substantially with the slope of the input waveform. For instance, an inverter's delay can be computed fairly accurately by representing each transistor with a linear resistor if the input transition is fast enough (compared to the output transition) (see Fig. 1.2(a)). If the input varies slowly the inverter's behavior can be better described as a linear amplifier (Fig. 1.2(b)). Using a detailed case analysis, the author shows how those two basic models can be combined to give an estimate of the inverter's delay which is accurate for all input and output waveforms, which he approximates with time-shifted ramps with exponential tails. He also shows how that approach can be extended to approximate the delays of general MOS gates.

The idea of approximating waveforms with exponentials is also used in WASIM, an

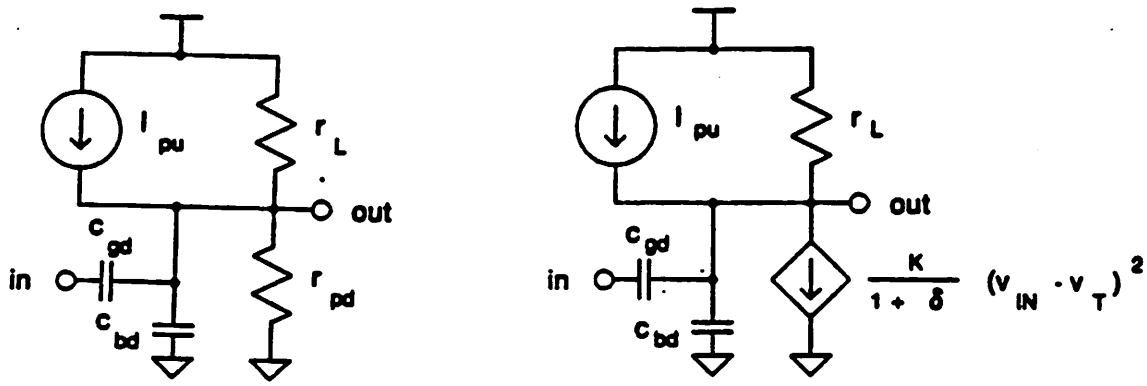


Figure 1.2: Matson's models to compute an inverter delay

event-driven, waveform-based simulator [7]. In this case the approximation is chosen to be a sum of exponentials of the form $\sum_i e^{-R_i}$, where

$$R_i = ((t - t_{s_i})/\tau_i)^{m_i}$$

The authors note that, with the use of nonlinear parameter fitting, the waveforms actually found in digital integrated circuits can be approximated with good accuracy. Under these assumptions, the behavior of a cell can be represented as a function of the type

$$\mathbf{R}_{out} = \mathcal{F}(\mathbf{R}_{in}, \mathbf{L}, \mathbf{P})$$

where $\mathbf{R}_{in}$ and $\mathbf{R}_{out}$ are vectors of parameters for the input and output waveforms respectively, $\mathbf{L}$ is a vector representing the cell's loading and $\mathbf{P}$ is another vector containing relevant cell parameters such as device sizes. The function $\mathcal{F}$ is represented using a table : a circuit simulator is used to compute the output waveforms correspond-

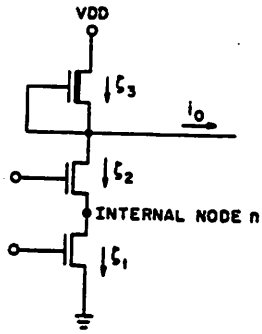


Figure 1.3: Schematic of a NAND gate

ing to a certain number of values for $\mathbf{R}_{in}$, $\mathbf{L}$ and $\mathbf{P}$. The corresponding values for $\mathbf{R}_{out}$ are obtained by nonlinear parameter fitting on the corresponding output waveforms. Polynomial interpolation is used to approximate the function for other values of its arguments.

In all the examples of macromodeling that we have presented so far, the actual input and output waveform shapes have been either completely neglected or parametrized by approximating them with exponential functions. This approach is obviously inadequate to represent the behavior of a cell when more detailed information about its dynamical behavior is needed (e.g. in circuit simulation). An approach to macromodeling that does not impose any *a-priori* limitations on the shape of the waveforms has been presented in [8]. There the authors build a static macromodel for a NAND gate in the following manner : let voltages v_1, v_2 and v_3 be applied to the input and output nodes respectively (Fig. 1.3). The output voltage of the unloaded NAND gate is a function of v_1 and v_2 : $v_{out} = f(v_1, v_2)$. If $v_3 = v_{out}$, no current flows from the voltage source v_3 into the output node; in all other cases, there is a *residual current* i_0 which is a

function of v_1, v_2 and v_3 :

$$i_0 = g(v_1, v_2, v_3)$$

The values of the function g corresponding to certain values of v_1, v_2 and v_3 can be found by simulating the gate in the appropriate conditions and monitoring the corresponding value of i_0. Piecewise cubic splines are then used to represent the function for other voltage values. If the NAND gate is a part of a bigger circuit, its behavior at the output node can be represented by the function g, eliminating the need to solve an additional equation corresponding to the internal node. It can be seen that no assumptions are made concerning the actual waveform shapes.

1.3 Previous Work In Model Order Reduction

The topic of order reduction for the realizations of a dynamical system has also been dealt with extensively in the system theory literature. Moore [9] notes that, when the system is linear, the problem of approximating it with a lower order model is strongly related to its Kalman decomposition [10]. Recall that for any linear system *controllable* and *unobservable* subspaces of the space state [11] can be defined, and that there exists a realization

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x}$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{A_{11}} & \mathbf{0} & \mathbf{A_{13}} & \mathbf{0} \\ \mathbf{A_{21}} & \mathbf{A_{22}} & \mathbf{A_{23}} & \mathbf{A_{24}} \\ \mathbf{0} & \mathbf{0} & \mathbf{A_{33}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A_{43}} & \mathbf{A_{44}} \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} \mathbf{B_1} \\ \mathbf{B_2} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}$$

$$\mathbf{C} = \begin{pmatrix} \mathbf{C_1} & \mathbf{0} & \mathbf{C_3} & \mathbf{0} \end{pmatrix}$$

such that

- the subsystem $\{\mathbf{A_{11}}, \mathbf{B_1}, \mathbf{C_1}\}$ is completely controllable and observable;
- the subsystem $\{\mathbf{A_{22}}, \mathbf{B_2}, \mathbf{0}\}$ is controllable but not observable;
- the subsystem $\{\mathbf{A_{33}}, \mathbf{0}, \mathbf{C_3}\}$ is observable but not controllable;
- the subsystem $\{\mathbf{A_{44}}, \mathbf{0}, \mathbf{0}\}$ is neither controllable nor observable.

This decomposition is very elegant from a theoretical point of view; unfortunately it is "structurally unstable", in the sense that arbitrary small perturbations of the system can change the dimensions of its controllable and unobservable parts. This implies, for instance, that we can perturb slightly a model whose controllable part is a proper subspace, and make that subspace not proper. Strictly speaking, this means that the impulse responses of the two systems are different, but Moore remarks that from a

practical point of view they have almost the same behavior. He goes on to show that, if $\mathbf{F}(t)$ is a matrix-valued function of time, its *Gramian*

$$\mathbf{W}^2 = \int_{t_1}^{t_2} \mathbf{F}(t)\mathbf{F}^T(t)\,dt$$

has the following properties :

- $\mathbf{F}(t) = \sum_{i=1}^{n} \mathbf{v}_i \mathbf{f}_i^T(t)$ where $\mathbf{v}_i$ is the *i-th* unit eigenvector of $\mathbf{W}^2$ corresponding to the eigenvalue σ_i^2, and $\mathbf{f}_i(t)$ is defined as $\mathbf{f}_i(t) = \mathbf{v}_i^T\mathbf{F}(t)$.

- Let $\mathbf{F}_k(t) = \sum_{i=1}^{k} \mathbf{v}_i \mathbf{f}_i^T(t)$. Then $\int_{t_1}^{t_2} \|\mathbf{F}(t) - \mathbf{F}_k(t)\|_F^2\,dt = \sum_{i=k+1}^{n} \sigma_i^2$, where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix and the eigenvalues are supposed to be ordered so that $\sigma_1^2 \geq \ldots \geq \sigma_n^2$. Moreover $\mathbf{F}_k(t)$ is optimal in the following sense : if $\mathbf{G}(t)$ is any piecewise continuous matrix-valued function such that dim span$\{v : v \in \text{im}\mathbf{G}(t)\,,\, t \in [t_1, t_2]\} = k$, then $\int_{t_1}^{t_2} \|\mathbf{F}(t) - \mathbf{F}_k(t)\|_F^2\,dt \leq \int_{t_1}^{t_2} \|\mathbf{F}(t) - \mathbf{G}(t)\|_F^2\,dt$.

If $\mathbf{F}(t)$ is the impulse response of the system, the above inequalities give bounds on the error resulting from the reduction of the order of the model. Moore also notes that the $\mathbf{v}_i$'s and σ_i's can be effectively computed using a very stable algorithm [12] for the *singular value decomposition* of a matrix [11,13]. This makes Moore's approach interesting both from a theoretical and practical point of view.

Other authors have approached the problem directly from an input-output point of view. This approach involves the *Hankel norm* of a transfer function $\mathbf{F}(t)$, defined in the following way. Let $\mathbf{u}(t) \in L^2[-\infty, 0]$, and let $\mathbf{y}(t)$ be

$$\mathbf{y}(t) = \int_0^{\infty} \mathbf{F}(t-\tau)\mathbf{u}(-\tau)\,d\tau$$

If $\mathbf{F}(t)$ is stable, $\mathbf{y}(t) \in L^2[0, +\infty]$; the Hankel norm of $\mathbf{F}(t)$ is defined as

$$\|\mathbf{F}(t)\|_H = \sup_{\|\mathbf{u}\|_2 \leq 1} \|\mathbf{y}(t)\|_2$$

In other words, the Hankel norm of $\mathbf{F}(t)$ is its operator norm, when $\mathbf{F}(t)$ is considered a linear mapping from the space of past inputs to the space of future outputs. Its practical significance resides in the fact that it lies between the more traditional L^2 and L^∞ norms [14]. Therefore the problem of computing approximations in the Hankel norm can be viewed as a compromise between the easy, but sometimes unsatisfactory, least squares criterion and the computationally difficult min-max approximation. When $\mathbf{F}(t)$ is the transfer function of a linear time-invariant system a complete solution to this problem was published by Glover [15], who gave a characterization of all best approximations to $\mathbf{F}(t)$ of given degree. In the same paper, he gives also L^∞-norm bounds for the error of optimal Hankel-norm approximations, which is a more useful indication of their performance in practical applications.

1.4 Our Approach

All the approaches to macromodeling that we have discussed in the previous sections, although valuable in particular situations, cannot be used to generate general purpose models for circuit simulation for one or more of the following reasons :

- They are often based on a case-specific analysis, and they cannot be extended to systems other than those for which they were specifically developed.
- Some models, like those proposed by Matson, are too simple, and their use for circuit simulation purposes would result in gross inaccuracies in the computed

waveforms.

- Those models that try to take waveform shapes into account, like the table lookup approach used in [7], make particular assumptions about them (e.g. exponential-like behavior). There are no guarantees about how a given model would behave under different circumstances.

- Static models, like those used in [8], make no assumptions about waveform shapes (exactly because they are computed in static conditions), but nothing can be said a priori about their accuracy for time-domain analysis.

- Unfortunately those methods, such as principal component analysis [9] or Hankel-norm approximation, that do provide information about the model's accuracy from a dynamical point of view rely heavily on the linearity of the system to be approximated. There does not seem to be any clear extension of those algorithms to nonlinear systems.

Our goal was to come up with an algorithm to generate macromodels with the following characteristics :

- The macromodels had to be general and accurate enough to be used for standard circuit simulation.

- No particular conditions had to be imposed on the system to be approximated, other then those usually assumed for electrical networks.

- The algorithm could not be purely empirical in nature, but it had to have some theoretical grounds. At the same time, it had to be implementable on a computer.

For these reasons I chose to look at the macromodeling problem exclusively from an input-output point of view. In other words, both the original system and its model are regarded as maps from the space of input signals to the space of output signals, so that, from a mathematical point of view, the task of finding a macromodel for a given system becomes an approximation problem in an abstract space whose elements are maps between function spaces. The next step is to define a notion of distance in this space that gives an indication of how close two systems are *from the point of view of their input-output relationships.* In principle this involves comparing the output of the system and that of its model for every possible input. It is a somewhat surprising result that, because of the particular structure of the equations describing electrical circuits, only a much smaller class of inputs needs be considered, namely those that are piecewise constant (see Chapter 4). It is this result that gives practical significance to an approach that would otherwise be interesting only from a theoretical point of view.

The metric structure induced by the above definition of distance will be used to develop an iterative algorithm that, given a dynamical system and an approximation to it, will modify it to generate another approximation which is locally optimal. For the sake of simplicity, we will assume that the set of admissible reduced order models can be parametrized by a point in R^r (i.e. every admissible model is completely characterized by r real numbers $\alpha_1, \ldots, \alpha_r$). However it should be pointed out that the underlying mathematical theory can be extended almost *verbatim* to deal with arbitrary classes of admissible models.

From the above description it is clear that the macromodeling problem is recast into an optimization problem. This requires computing the time-domain derivatives of a

certain cost functional with respect to variations in the model parameters. In Chapter 4 it is shown that they can be computed in terms of an integral containing a certain Hamiltonian function. Similar formulas were derived by Hachtel and Rohrer [16] in a different context and using different mathematical techniques. In their paper, they consider the problem of minimizing a scalar performance function (for instance delay or switching time) of an electrical circuit by an appropriate choice of certain design parameters. From a mathematical viewpoint this translates into the minimization of a functional

$$c(\mathbf{p}) = \int_0^T h[\mathbf{x}(t,\mathbf{p}),\mathbf{p},t]\,dt$$

where $\mathbf{p}$ is the vector of the design parameters, and $\mathbf{x}$ is the vector of the state variables, which is assumed to satisfy the differential equation

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x},\mathbf{p},t)$$

The authors' choice for the mathematical setting of this problem is the calculus of variations, instead of optimal control theory. For this reason their starting point is the Lagrangian function

$$L(\mathbf{x},\mathbf{p},\boldsymbol{\lambda},\boldsymbol{\mu},t) = h(\mathbf{x},\mathbf{p},t) + \boldsymbol{\lambda}^T[\dot{\mathbf{x}} - \mathbf{F}(\mathbf{x},\mathbf{p},t)] + \boldsymbol{\mu}^T\dot{\mathbf{p}}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the adjoint variables. If we introduce the vector $\mathbf{z} = (\mathbf{x},\mathbf{p},\boldsymbol{\lambda},\boldsymbol{\mu})$, the theory of the calculus of variations requires that the following differential equations, known as Euler equations, be satisfied

$$\frac{\partial L}{\partial \mathbf{z}} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\mathbf{z}}}\right) = 0$$

which in this case become

$$\begin{aligned}
\frac{d\mathbf{x}}{dt} - \mathbf{F}(\mathbf{x},\mathbf{p},t) &= 0 \\
\frac{d\mathbf{p}}{dt} &= 0 \\
\frac{\partial h}{\partial \mathbf{x}} - \boldsymbol{\lambda}^T \frac{\partial F}{\partial \mathbf{x}} - \frac{d\boldsymbol{\lambda}^T}{dt} &= 0 \\
\frac{\partial h}{\partial \mathbf{p}} - \boldsymbol{\lambda}^T \frac{\partial F}{\partial \mathbf{p}} - \frac{d\boldsymbol{\mu}^T}{dt} &= 0
\end{aligned}$$

and after some algebra the authors derive the following expression for the derivative of the performance function :

$$\frac{\partial c}{\partial \mathbf{p}} = -\boldsymbol{\lambda}^T(0)\frac{\partial \mathbf{x}(0)}{\partial \mathbf{p}} + \int_0^T \left(-\boldsymbol{\lambda}^T \frac{\partial \mathbf{F}}{\partial \mathbf{p}} + \frac{\partial h}{\partial \mathbf{p}}\right) dt$$

which is identical to the expression that we will derive in Chapter 4 (apart from a sign change, deriving from a different definition of the Lagrangian function). Nevertheless we believe that our approach along the lines of optimal control theory lends itself better to a unified treatment of variational calculus, because it brings to light the fact that the formulas involved are substantially the same, whether we are varying circuit parameters or functions of the time. As a matter of fact, the computer implementation of our algorithm exploits the fact that one single backward integration can be used to compute derivatives with respect to both model parameters and input waveform, with a significant saving of CPU time.

Chapter 2

Mathematical Preliminaries

This chapter contains miscellaneous mathematical results which will be used in the following chapters. Almost all the theorems presented here are well known, and are therefore stated without proofs. The proofs as well as more in-depth treatment of each topic can be found in the references given in each section.

2.1 Topological Spaces

A topological space is a pair $(X, \mathcal{T})$ where X is a set and $\mathcal{T}$ is a collection of subsets of X satisfying the following axioms :

(A1) X and $\emptyset$ (the empty set) belong to $\mathcal{T}$.

(A2) If $\{V_\alpha\}$ is an arbitrary collection of elements of $\mathcal{T}$, the intersection $\bigcap_\alpha V_\alpha$ belongs to $\mathcal{T}$.

(A3) If V_1 and V_2 belong to $\mathcal{T}$, the union $V_1 \bigcup V_2$ belongs to $\mathcal{T}$.

The elements of $\mathcal{T}$ are called *open sets*; by definition, a subset of X is *closed* if its complement is open.

In many cases the topology $\mathcal{T}$ is defined in terms of a so-called *basis*. A basis for a topology on X is a collection $\mathcal{B}$ of subsets of X (called *basis elements*) satisfying the following axioms :

(A4) For each $x \in X$ there is at least one element $B \in \mathcal{B}$ such that $x \in B$.

(A5) If B_1 and B_2 belong to $\mathcal{B}$, for every $x \in B_1 \bigcap B_2$ there exists $B_3 \in \mathcal{B}$ such that $x \in B_3 \subseteq B_1 \bigcap B_2$.

Definition 2.1 *If $\mathcal{B}$ is a basis for a topology on X, the topology $\mathcal{T}$ generated by $\mathcal{B}$ is the collection of subsets of* X *which are the union of elements of $\mathcal{B}$.*

In other words, a set $V \subseteq X$ belongs to $\mathcal{T}$ if and only if it is the union of elements of $\mathcal{B}$. For instance, the topology of the real numbers R is usually defined in terms of a basis, the elements of which are the open intervals (a, b).

In a topological space there are certain sets which are of particular interest because of their special properties. Those sets are called *compact* and are defined in the following way.

Definition 2.2 *Let A be a subset of the topological space X. An* **open covering** *of A is a collection $\{V_\alpha\}$ of open sets such that $A \subseteq \bigcup_\alpha V_\alpha$.*

Definition 2.3 *Let A be a subset of the topological space X. A is* **compact** *if every open covering of A contains a finite subcollection that is also a covering for A.*

It can be shown [17, p.174] that a subset of R^n is compact if and only if it is closed and bounded. A useful property of compactness is that it is preserved by cartesian products.

Definition 2.4 *Let X, Y be two sets. Their* **cartesian product,** *denoted by $X \times Y$, is the set of ordered pairs (x, y), where $x \in X$ and $y \in Y$.*

Theorem 2.1 ([17, p. 167]) *The cartesian product of two compact sets is compact.*

Let f be a function from the real numbers R into R. The classical definition of continuity for f formalizes the intuitive notion that $f(x)$ is "close" to $f(y)$ whenever x is "close" to y. It is possible to generalize the definition of continuity to functions between topological spaces in the following way.

Definition 2.5 *Let X, Y be two topological spaces. A function $f : X \to Y$ is* **continuous** *if the inverse image under f of every open set of Y is open in X.*

It can be shown (but we will not do it here) that this definition is equivalent to the classical one when $X = Y = \mathrm{R}$.

Continuous functions have many interesting properties. One of them, which we will make use of, is stated in the following theorem.

Theorem 2.2 ([17, p. 175]) *Let $f : X \to \mathrm{R}$ be a continuous function from the topological space X into the real numbers, and let K be a compact subset of X. Then f has a maximum and a minimum on K, i.e. there exist points $x_m, x_M \in K$ such that*

$$f(x_m) \leq f(x) \leq f(x_M) \forall x \in K$$

Another useful property of continuous functions on compact subsets of R^n is *uniform continuity.*

Definition 2.6 *Let X be a subset of R^n. A function $\mathbf{f} : X \to \mathrm{R}^m$ is* **uniformly continuous** *on X if for every $\epsilon > 0$ there exists a $\delta_\epsilon > 0$ such that $\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})\| < \epsilon$ whenever $\mathbf{x}, \mathbf{y} \in X$ and $\|\mathbf{x} - \mathbf{y}\| < \delta_\epsilon$.*

Theorem 2.3 ([17, p. 180]) *Let X be a compact subset of R^n, and let $\mathbf{f} : X \to \mathrm{R}^m$ be a continuous function. Then f is uniformly continuous on X.*

2.2 Derivatives in Banach Spaces

If $f(x)$ is a real function of a real variable x, its derivative $f'(x_0)$ at x_0 can be described informally as the best linear approximation to $f(x)$ in a neighborhood of x_0. The idea of "best linear approximation" can be generalized to a much wider class of functions, and it is a very useful tool whenever one has to deal with "small variations". Because I will need it in the sequel, this generalized derivative will now be defined. A rigorous and complete analysis of this topic can be found in [18].

2.2.1 Banach Spaces

Let X be a vector space over the reals. A *norm* on X is a real valued function $\|\cdot\| : X \to \mathrm{R}$ which satisfies the following conditions :

$$\|\mathbf{x}\| \geq 0, \|\mathbf{x}\| = 0 \Leftrightarrow \mathbf{x} = \mathbf{0}$$

$$\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$$

$$\|a\mathbf{x}\| = |a|\|\mathbf{x}\|$$

A vector space on which a norm has been defined is called a *normed vector space*, or simply a *normed space*. The most common example of normed spaces is $\mathbf{R}^n$ with the *Euclidean norm*

$$\|\mathbf{x}\| = \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}}$$

Examples of Banach spaces whose elements are functions will be given in Section 2.3. In a normed space we can also define a notion of distance between two points $\mathbf{x}$ and $\mathbf{y}$, given by $\|\mathbf{y}-\mathbf{x}\|$. A *Cauchy sequence* in a normed space X is a sequence of points $\{\mathbf{x}_n\}$ such that for every $\epsilon > 0$ we can find an index N_ϵ such that we have $\|\mathbf{x}_n - \mathbf{x}_m\| < \epsilon$ whenever $n, m > \epsilon$. A sequence $\{\mathbf{x}_n\}$ is said to converge to $\hat{\mathbf{x}} \in X$ if for every $\epsilon > 0$ we can find an index N_ϵ such that $\|\mathbf{x} - \hat{\mathbf{x}}\| < \epsilon$ whenever $n > N_\epsilon$; in this case we write $\hat{\mathbf{x}} = \lim_{n\to\infty} \mathbf{x}_n$. A normed vector space X is said to be *complete* if every Cauchy sequence has a limit in X. A complete normed vector space is called a *Banach space*. $\mathbf{R}^n$ is a Banach space.

2.2.2 Duals of Banach Spaces

Let X be a Banach space over the reals. A function $\phi : X \to \mathbf{R}$ is *linear* if $\phi(a\mathbf{x}+b\mathbf{y}) = a\phi(\mathbf{x}) + b\phi(\mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in X$ and $a, b \in \mathbf{R}$. A continuous linear function from a real Banach space into $\mathbf{R}$ is often called a *functional*. The set of all functionals over X form another Banach space with the norm

$$\|\phi\| = \sup_{\|\mathbf{x}\|\leq 1} |\phi(\mathbf{x})|$$

This Banach space is called the *dual* of X and is denoted by X^*.

Beside the topology defined by the norm, X^* is often given another topology, called

the *weak-* topology*, which is defined as the topology induced by the basis whose elements are sets of the form

$$B(\phi_0, \mathbf{x}_1, \ldots, \mathbf{x}_n, \varepsilon_1, \ldots, \varepsilon_n) = \{\phi \in X^* : |\phi(\mathbf{x}_i) - \phi_0(\mathbf{x}_i)| < \varepsilon_i, i = 1, \ldots, n\} \quad (2.1)$$

where $\phi_0 \in X^*, \mathbf{x}_1, \ldots, \mathbf{x}_n \in X$ and $\varepsilon_1, \ldots, \varepsilon_n$ are positive real numbers.

One of the most important features of the weak-* topology is stated in the following theorem [19, p. 174]:

Theorem 2.4 (Alaoglu) *The closed unit ball* $S = \{\phi \in X^* : \|\phi\| \leq 1\}$ *of* X^* *is compact in the weak-* topology.*

2.2.3 The Frechet Derivative

Let X, Y be two Banach spaces, $\mathbf{f}$ a function from X into Y and $\mathbf{x}_0$ a point of X. The *Frechet derivative of* $\mathbf{f}$ *at* $\mathbf{x}_0$ is a linear operator $\mathbf{F} : X \to Y$ such that

$$\lim_{\Delta\mathbf{x} \to \mathbf{0}} \frac{\|\mathbf{f}(\mathbf{x}_0 + \Delta\mathbf{x}) - \mathbf{f}(\mathbf{x}_0) - \mathbf{F}\Delta\mathbf{x}\|}{\|\Delta\mathbf{x}\|} = 0 \quad (2.2)$$

As in the case of real-valued functions, the Frechet derivative of $\mathbf{f}$ may or may not exist. However, if it exists, it is unique [18]. Just to give a concrete example, let $X = \mathrm{R}^n, Y = \mathrm{R}^m, \mathbf{f} = (f_1, \ldots, f_m)^T$. Assume that all the partial derivatives $\frac{\partial f_i}{\partial x_j}$ exist and are continuous in a neighborhood of $\mathbf{x}_0$. Then the Frechet derivative $\mathbf{F}$ of $\mathbf{f}$ at $\mathbf{x}_0$ is the *Jacobian matrix*

$$\mathbf{F} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$$

where all the partial derivatives are intended to be evaluated at $\mathbf{x}_0$. To check that $\mathbf{F}$ does indeed satisfy eqn. (2.2), remember that under the stated assumptions for each $\epsilon > 0$ we can find positive numbers $\delta_1, \ldots, \delta_n$ such that

$$\frac{|f_i(\mathbf{x}_0 + \Delta\mathbf{x}) - f_i(\mathbf{x}_0) - \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j} \Delta x_j|}{\|\Delta\mathbf{x}\|} < \frac{\epsilon}{\sqrt{n}}$$

whenever $\|\Delta\mathbf{x}\| < \delta_i$. If $\|\Delta\mathbf{x}\| < \delta = \min(\delta_1, \ldots, \delta_n)$ then

$$\begin{aligned} &\|\mathbf{f}(\mathbf{x}_0 + \Delta\mathbf{x}) - \mathbf{f}(\mathbf{x}_0) - \mathbf{F}\Delta\mathbf{x}\| = \\ &= \left(\sum_{i=1}^{n} (f_i(\mathbf{x}_0 + \Delta\mathbf{x}) - f_i(\mathbf{x}_0) - \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j} \Delta x_j)^2 \right)^{\frac{1}{2}} < \\ &< \epsilon \|\Delta\mathbf{x}\| \end{aligned}$$

which proves (2.2). The Frechet derivative of a vector-valued function is often denoted by $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$, and this is the notation that we will use from now on.

2.2.4 The Gateaux Derivative

There is another derivative which is often used in optimization, and it is defined in the following way. Let $\mathbf{x}_0 \in X$; the *Gateaux* (or *weak*) *derivative of* $\mathbf{f}$ *at* $\mathbf{x}_0$ is a linear operator $\mathbf{G}(\mathbf{x}_0)$ such that the equality

$$\mathbf{G}(\mathbf{x}_0)\Delta\mathbf{x} = \lim_{t \to 0} \frac{\mathbf{f}(\mathbf{x}_0 + t\Delta\mathbf{x}) - \mathbf{f}(\mathbf{x}_0)}{t} \tag{2.3}$$

is satisfied for all $\Delta\mathbf{x}$, provided that the above limits exist. If $\mathbf{f}$ has a Frechet derivative $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_0)$ then it also has a Gateaux derivative $\mathbf{G}(\mathbf{x}_0) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_0)$. However a function can have a Gateaux derivative but not a Frechet derivative ([18, p. 67], example E 3.1-6).

2.3 Function Spaces

In this section, I give the definitions of some Banach spaces whose elements are functions. In addition I state some properties of these spaces which will be needed in the following chapters.

Let $[0,T]$ be a closed bounded interval of the real line. $C([0,T],\mathrm{R}^n)$ denotes the set of all continuous functions from $[0,T]$ into R^n, and it is a Banach space under the norm

$$\|\mathbf{x}\| = \sup_{t\in[0,T]} \|\mathbf{x}(t)\|$$

Convergence in $C([0,T],\mathrm{R}^n)$ is called *uniform convergence* : a sequence $\{\mathbf{x}_n\}$ converges to $\mathbf{x}$ if for every $\varepsilon > 0$ we can find an index N_ε such that

$$\sup_{t\in[0,T]} \|\mathbf{x}_n(t) - \mathbf{x}(t)\| < \varepsilon$$

whenever $n > N_\varepsilon$. This is a stronger requirement than *pointwise convergence*, which is defined as

$$\lim_{n\to\infty} \mathbf{x}_n(t) = \mathbf{x}(t) \qquad \forall t \in [0,T]$$

However if additional restrictions are imposed on the sequence $\{\mathbf{x}_n\}$, pointwise convergence becomes equivalent to uniform convergence.

Definition 2.7 *A collection* $\{\mathbf{x}_\alpha(t)\}$ *of functions on* $[0,T]$ *is called* **equicontinuous** *if, for every* $\varepsilon > 0$, *there exists a* $\delta_\varepsilon > 0$ *such that the inequality* $\|\mathbf{x}_\alpha(t_1) - \mathbf{x}_\alpha(t_2)\| < \varepsilon$ *is satisfied for every* $|t_1 - t_2| < \delta_\varepsilon$ *and every* α.

Theorem 2.5 *Let* $\{\mathbf{x}_n\}$ *be a sequence in* $C([0,T],\mathrm{R}^n)$. *If the sequence* $\{\mathbf{x}_n\}$ *is equicontinuous and converges pointwise to* $\mathbf{x}$, *then it converges to* $\mathbf{x}$ *uniformly.*

Proof. First we prove that if we add the limit function $\mathbf{x}$ to the sequence $\{\mathbf{x}_n\}$ we still get an equicontinuous collection. Under the stated assumptions, given $\varepsilon > 0$ there exists $\delta_\varepsilon > 0$ such that $\|\mathbf{x}_n(t_1) - \mathbf{x}_n(t_2)\| < \varepsilon/3$ if $|t_1 - t_2| < \delta_\varepsilon$. If we take the limit of this inequality for $n \to \infty$, we obtain $\|\mathbf{x}(t_1) - \mathbf{x}(t_2)\| \leq \varepsilon/3$, which proves our claim. Now choose $M+1$ points $0 = t_0 < t_1 < \ldots < t_M = T$ such that $|t_i - t_{i-1}| < \delta_\varepsilon, i = 1, \ldots, M$. Because the sequence converges to $\mathbf{x}$ pointwise, for every t_i we can find an index N_i such that $\|\mathbf{x}_n(t_i) - \mathbf{x}(t_i)\| < \varepsilon/3$ if $n > N_i$. Let $N = \max(N_0, \ldots, N_M)$. Let t be any point in $[0, T]$: there exists a point t_i such that $|t - t_i| < \delta_\varepsilon$. Then if $n > N$ we have

$$\begin{aligned} \|\mathbf{x}_n(t) - \mathbf{x}(t)\| &\leq \|\mathbf{x}_n(t) - \mathbf{x}_n(t_i)\| + \\ &+ \|\mathbf{x}_n(t_i) - \mathbf{x}(t_i)\| + \|\mathbf{x}(t_i) - \mathbf{x}(t)\| < \\ &< \frac{\varepsilon}{3} + \frac{\varepsilon}{3} + \frac{\varepsilon}{3} = \varepsilon \end{aligned}$$

which proves that $\{\mathbf{x}_n\}$ converges to $\mathbf{x}$ uniformly. □

Two other function spaces which will be mentioned are $L^1([0,T], \mathrm{R}^n)$ and $L^\infty([0,T], \mathrm{R}^n)$, which are the spaces of measurable functions[1] from $[0,T]$ into R^n which satisfy the following conditions

$$\|\mathbf{x}\| \triangleq \int_0^T \|\mathbf{x}(t)\|\, dt < +\infty \qquad \text{for } L^1([0,T], \mathrm{R}^n) \tag{2.4}$$

$$\|\mathbf{x}\| \triangleq \operatorname{ess\,sup}_{t \in [0,T]} \|\mathbf{x}(t)\| < +\infty \qquad \text{for } L^\infty([0,T], \mathrm{R}^n) \tag{2.5}$$

[1] For the definition of measurable function and related terminology see [20].

The quantities appearing on the left hand side of eqns. (2.4) and (2.5) define the norms on L^1 and L^∞ respectively. Under these norms both spaces are Banach spaces, and $L^\infty([0,T],\mathrm{R}^n)$ is the dual of $L^1([0,T],\mathrm{R}^n)$ [19, p. 145], in the sense that every functional ϕ on $L^1([0,T],\mathrm{R}^n)$ can be uniquely identified with a function $\mathbf{u} \in L^\infty([0,T],\mathrm{R}^n)$ satisfying the following identity :

$$\phi(\mathbf{x}) = \int_0^T <\mathbf{x}(t),\mathbf{u}(t)>\, dt \qquad \forall \mathbf{x} \in L^1([0,T],\mathrm{R}^n)$$

Therefore $L^\infty([0,T],\mathrm{R}^n)$ has a weak-* topology; sequences that converge in this topology are characterized by the following theorem.

Theorem 2.6 *Let* $\{\mathbf{u}_n\}$ *be a sequence in* $L^\infty([0,T],\mathrm{R}^n)$. $\{\mathbf{u}_n\}$ *converges to* $\mathbf{u} \in L^\infty([0,T],\mathrm{R}^n)$ *in the weak-* topology if and only if*

$$\lim_{n\to\infty} \int_0^T <\mathbf{x}(t),\mathbf{u}_n(t)>\, dt = \int_0^T <\mathbf{x}(t),\mathbf{u}(t)\, dt> \qquad \forall \mathbf{x} \in L^1([0,T],\mathrm{R}^n) \quad (2.6)$$

Proof. We can assume $\mathbf{u} = 0$ (otherwise replace $\mathbf{u}_n$ with $\mathbf{u}_n - \mathbf{u}$). For every $\mathbf{x} \in L^1$ and every $\varepsilon > 0$ the set $B(\mathbf{0},\mathbf{x},\varepsilon)$ (eqn. (2.1)) is an open neighborhood of $\mathbf{0}$ in the weak-* topology. Therefore if $\mathbf{u}_n \to \mathbf{0}$ there exists an index N_ε such that $n > N_\varepsilon$ implies $\mathbf{u}_n \in B(\mathbf{0},\mathbf{x},\varepsilon)$, which is equivalent to

$$\left| \int_0^T <\mathbf{x}(t),\mathbf{u}_n(t)>\, dt \right| < \varepsilon$$

To prove the converse let $B(\mathbf{0},\mathbf{x}_1,\ldots,\mathbf{x}_p,\varepsilon_1,\ldots,\varepsilon_p)$ be a weak-* open neighborhood of $\mathbf{0}$. Because eqn. (2.6) holds, we can find indices $N_1,\ldots,N_p$ such that $n > N_i$ implies

$$\left| \int_0^T <\mathbf{x}_i(t),\mathbf{u}_n(t)>\, dt \right| < \varepsilon_i, \qquad i = 1,\ldots,p$$

Therefore $n > \max(N_1, \ldots, N_p)$ implies $\mathbf{x} \in B(\mathbf{0}, \mathbf{x}_1, \ldots, \mathbf{x}_p, \epsilon_1, \ldots, \epsilon_p)$.

□

The following theorem can be used to establish the equicontinuity of a collection of functions that are differentiable almost everywhere.

Theorem 2.7 *Let* $\{\mathbf{x}_\alpha\} \in C([0,T], \mathrm{R}^n)$ *be a collection of functions that are differentiable almost everywhere in* $[0,T]$. *Suppose that* $\dot{\mathbf{x}}_\alpha \in L^\infty([0,T], \mathrm{R}^n)$ *for every* α *and that there exists a constant* $M > 0$ *such that* $\|\dot{\mathbf{x}}_\alpha\| \leq M \ \forall \alpha$. *Then the collection* $\{\mathbf{x}_\alpha\}$ *is equicontinuous on* $[0,T]$.

Proof. For every $t_1, t_2 \in [0,T]$ we can write

$$\|\mathbf{x}_\alpha(t_1) - \mathbf{x}_\alpha(t_2)\| = \left\| \int_{t_1}^{t_2} \dot{\mathbf{x}}_\alpha(t)\, dt \right\| \leq M|t_1 - t_2|$$

Therefore given $\epsilon > 0$ we will have $\|\mathbf{x}_\alpha(t_1) - \mathbf{x}_\alpha(t_2)\| < \epsilon$ for every α if $|t_2 - t_1| < \epsilon/M$. □

2.4 Differential Equations

In this section we will prove existence and uniqueness theorems under assumptions applicable to our specific problem. Most textbooks study the differential equation

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), t]$$

assuming that $\mathbf{f}$ is continuous in both $\mathbf{x}$ and t. However we are only interested in a special type of differential equation, namely one of the form

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t)] + \mathbf{B}\mathbf{u}(t) \tag{2.7}$$

where $\mathbf{B}$ is an $n \times m$ matrix and $\mathbf{u} \in L^\infty([0,T],\mathrm{R}^m)$. We will see that under certain assumptions we can still prove the existence and uniqueness of a solution of eqn. (2.7) satisfying the initial condition $\mathbf{x}(0) = \mathbf{x}_0$, i.e. of a function $\mathbf{x} \in C([0,T],\mathrm{R}^n)$ which is differentiable a.e. in $[0,T]$ and which satisfies eqn. (2.7) a.e. in [0,T].

First of all we note that any solution of eqn. (2.7) such that $\mathbf{x}(0) = \mathbf{x}_0$ is a solution of the integral equation

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_0^t \mathbf{f}[\mathbf{x}(\tau)]\,d\tau + \mathbf{B}\int_0^t \mathbf{u}(\tau)\,d\tau \tag{2.8}$$

and viceversa. Therefore it is equivalent to study the properties of eqn. (2.8).

To prove our theorems we will need to use the following two lemmas, known as *Gronwall's inequality* and *generalized Gronwall's inequality* [21, p. 36].

Lemma 2.1 *If $b(t), \phi(t)$ are continuous functions with $b(t) \geq 0$ and if a is a real number such that the following inequality is satisfied*

$$\phi(t) \leq a + \int_{t_1}^t b(\tau)\phi(\tau)\,d\tau \qquad t \in [t_1, t_2]$$

then $\phi(t)$ satisfies

$$\phi(t) \leq a e^{\int_{t_1}^t b(\tau)\,d\tau} \qquad t \in [t_1, t_2]$$

Lemma 2.2 *If $a(t), \phi(t)$ are real-valued continuous functions on $[t_1, t_2]$, $b(t) \in L^1([t_1, t_2], \mathrm{R})$, $b(t) \geq 0$ and*

$$\phi(t) \leq a(t) + \int_{t_1}^t b(\tau)\phi(\tau)\,d\tau \qquad t \in [t_1, t_2]$$

then

$$\phi(t) \leq a(t) + \int_{t_1}^t a(\tau)b(\tau)e^{\int_\tau^t b(u)\,du}\,d\tau \qquad t \in [t_1, t_2]$$

The existence of a solution of eqn. (2.8) will be proven by taking the limit of sequence of functions. In order to show that the limit exists we will rely on the fact that the sequence satisfies the inequality defined in the following lemma.

Definition 2.8 *The function* $\mathbf{f} : \mathbf{R}^n \to \mathbf{R}^n$ *is* **Lipschitz continuous** *if there exists a number* $L > 0$ *such that*

$$\|\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\| \qquad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbf{R}^n$$

Lemma 2.3 *Let* $\mathbf{u} \in L^\infty([0,T], \mathbf{R}^m)$ *and let* $\mathbf{f}$ *be Lipschitz continuous on* $\mathbf{R}^n$. *Recursively define the following sequence of functions :*

$$\begin{aligned} \mathbf{x}_0(t) &\equiv \mathbf{x}_0 \\ \mathbf{x}_{k+1}(t) &= \mathbf{x}_0 + \int_0^t \mathbf{f}[\mathbf{x}_k(\tau)]\, d\tau + \mathbf{B}\int_0^t \mathbf{u}(\tau)\, d\tau \end{aligned}$$

Then there exists a constant $C > 0$ *such that the following inequality is satisfied :*

$$\|\mathbf{x}_{k+1}(t) - \mathbf{x}_k(t)\| \leq C\frac{(Lt)^k}{k!} \quad \forall t \in [0,T]$$

Proof. Define

$$C = \sup_{t\in[0,T]} \|\mathbf{x}_1(t) - \mathbf{x}_0\|$$

Then the inequality is satisfied for $k = 0$ by construction. Using induction on k we obtain

$$\begin{aligned} \|\mathbf{x}_{k+1}(t) - \mathbf{x}_k(t)\| &= \left\| \int_0^t (\mathbf{f}[\mathbf{x}_k(\tau)] - \mathbf{f}[\mathbf{x}_{k-1}(\tau)])\, d\tau \right\| \leq \\ &\leq \int_0^t L\|\mathbf{x}_k(\tau) - \mathbf{x}_{k-1}(\tau)\|\, d\tau \leq L\int_0^t C\frac{(L\tau)^{k-1}}{(k-1)!}\, d\tau = C\frac{(Lt)^k}{k!} \end{aligned}$$

which completes the proof. □

We are now ready to prove the existence theorem.

Theorem 2.8 *Let* $\mathbf{u} \in L^\infty([0,T],\mathrm{R}^m)$ *and let* $\mathbf{f}$ *be Lipschitz continuous on* R^n. *Then eqn. (2.8) has a solution on* $[0,T]$.

Proof. Let $\{\mathbf{x}_k\}$ be the sequence defined in lemma 2.3 : then $\{\mathbf{x}_k\}$ is a Cauchy sequence in $C([0,T],\mathrm{R}^n)$. To prove this, let $p > q$; from lemma 2.3 we have

$$\begin{aligned}\|\mathbf{x}_p(t) - \mathbf{x}_q(t)\| &\le \sum_{k=q}^{p-1} \|\mathbf{x}_{k+1}(t) - \mathbf{x}_k(t)\| \le \\ &\le \sum_{k=q}^{p-1} C\frac{(Lt)^k}{k!} \le C\frac{(Lt)^q}{q!}\sum_{k=q}^{\infty}\frac{(Lt)^{k-q}}{k(k-1)\dots(q+1)} \le \\ &\le C\frac{(Lt)^q}{q!}\sum_{k=q}^{\infty}\frac{(Lt)^{k-q}}{(k-q)!} = C\frac{(Lt)^q}{q!}e^{Lt}\end{aligned}$$

It follows that

$$\|\mathbf{x}_p - \mathbf{x}_q\| = \sup_{t\in[0,T]}\|\mathbf{x}_p(t) - \mathbf{x}_q(t)\| \le C\frac{(LT)^q}{q!}e^{LT}$$

Because $\lim_{q\to\infty}\frac{(LT)^q}{q!} = 0$, for every $\varepsilon > 0$ we can find an index N_ε such that $C\frac{(LT)^q}{q!}e^{LT} < \varepsilon$ whenever $q > N_\varepsilon$. But then we must also have $\|\mathbf{x}_p - \mathbf{x}_q\| < \varepsilon$ for $p, q > N_\varepsilon$, which shows that $\{\mathbf{x}_k\}$ is a Cauchy sequence.

Because $C([0,T],\mathrm{R}^n)$ is a Banach space, the sequence $\{\mathbf{x}_k\}$ must have a limit $\mathbf{x}$. Therefore we can write

$$\begin{aligned}\mathbf{x}(t) = \lim_{k\to\infty}\mathbf{x}_k(t) &= \lim_{k\to\infty}\left(\mathbf{x}_0 + \int_0^t \mathbf{f}[\mathbf{x}_{k-1}(\tau)]\,d\tau + \right. \\ &+ \left.\mathbf{B}\int_0^t \mathbf{u}(\tau)\,d\tau\right) = \\ &= \mathbf{x}_0 + \int_0^t \mathbf{f}[\mathbf{x}(\tau)]\,d\tau + \mathbf{B}\int_0^t \mathbf{u}(\tau)\,d\tau\end{aligned}$$

which means that $\mathbf{x}$ is a solution of eqn. (2.8). □

After establishing existence we prove the uniqueness of the solution.

Theorem 2.9 *Under the same assumptions as in theorem 2.8 the solution of eqn. (2.8) is unique.*

Proof. If both $\mathbf{x}_1$ and $\mathbf{x}_2$ solve eqn. (2.8) we can write

$$\|\mathbf{x}_1(t) - \mathbf{x}_2(t)\| = \left\| \int_0^t (\mathbf{f}[\mathbf{x}_1(\tau)] - \mathbf{f}[\mathbf{x}_2(\tau)])\, d\tau \right\| \leq$$
$$\leq \int_0^t L\|\mathbf{x}_1(\tau) - \mathbf{x}_2(\tau)\|\, d\tau$$

If we let $\phi(t) = \|\mathbf{x}_1(t) - \mathbf{x}_2(t)\|$, we see that the assumptions of lemma 2.1 are satisfied with $b(t) \equiv L$ and $a = 0$. Therefore we conclude that $\phi(t) \equiv 0$ on $[0, T]$. □

2.5 Optimal Control

This section deals with the classical theory of optimal control. We follow substantially the arguments given in [22].

2.5.1 Problem Formulation

The classical optimal control problem can be formulated in many equivalent ways; the one that best suits our purposes goes under the name of *Meyer's problem.* I will not have to deal with its most general form, and I will describe just one particular case (fixed terminal time, fixed initial conditions), which is the one that I will have to consider later on.

A dynamical system is given, described by the set of differential equations

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t)] \tag{2.9}$$

where $\mathbf{x}(t) \in \mathrm{R}^n, \mathbf{u}(t) \in \mathrm{R}^m$, a vector $\mathbf{x}_0 \in \mathrm{R}^n$, a positive number T, a real-valued function $\varphi : \mathrm{R}^n \rightarrow \mathrm{R}$ usually called *cost function* or *performance index*, and a set $\mathcal{U}$ of *admissible inputs*. Meyer's problem consists of finding that input $\mathbf{u}(t) \in \mathcal{U}$ that minimizes $\varphi[\mathbf{x}(T)]$ where $\mathbf{x}(t)$ satisfies eqn. (2.9) with the initial conditions $\mathbf{x}(0) = \mathbf{x}_0$. In formulas

$$\begin{aligned} &\min_{\mathbf{u}\in\mathcal{U}} \varphi[\mathbf{x}(T)] \\ &\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t)] \\ &\mathbf{x}(0) = \mathbf{x}_0 \end{aligned} \tag{2.10}$$

2.5.2 The Lagrange Multipliers

Of all numerical algorithms that can be used to solve Meyer's problem (or any other minimization problem for that matter), the most efficient ones require the knowledge of the first derivative of the cost function with respect to the independent variable, which in this case is the time function $\mathbf{u}(t)$. This can be done in the following manner.

Let $\boldsymbol{\lambda} : \mathrm{R}^n \rightarrow \mathrm{R}^n$ be a yet unspecified vector-valued function and define

$$H(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}) = \boldsymbol{\lambda}^T \mathbf{f}(\mathbf{x}, \mathbf{u}).$$

H is a real-valued function, called the *Hamiltonian*, and it plays an important role in both the theoretical characterization and the numerical solution of this and other optimal control problems. We can now write the following equation, which holds for

all functions λ of class C^1 :

$$\begin{aligned}\varphi[\mathbf{x}(T)] &= \varphi[\mathbf{x}(T)] + \int_0^T \boldsymbol{\lambda}^T(t)(\mathbf{f}[\mathbf{x}(t),\mathbf{u}(t)] - \dot{\mathbf{x}}(t))\,dt = \\ &= \varphi[\mathbf{x}(T)] + \int_0^T \boldsymbol{\lambda}^T\mathbf{f}\,dt - [\boldsymbol{\lambda}^T\mathbf{x}]_{t=0}^{t=T} + \int_0^T \dot{\boldsymbol{\lambda}}^T\mathbf{x}\,dt = \\ &= \varphi[\mathbf{x}(T)] + \int_0^T H[\mathbf{x}(t),\boldsymbol{\lambda}(t),\mathbf{u}(t)]\,dt - \\ &- [\boldsymbol{\lambda}^T\mathbf{x}]_{t=0}^{t=T} + \int_0^T \dot{\boldsymbol{\lambda}}^T\mathbf{x}\,dt\end{aligned}$$

Let us now consider a small perturbation $\delta\mathbf{u}(t)$ of the input $\mathbf{u}(t)$ and a small perturbation $\delta\mathbf{x}(0)$ of the initial condition $\mathbf{x}_0$, while keeping the function $\boldsymbol{\lambda}$ fixed. As a consequence, the solution of eqn. (2.9) changes by a small amount $\delta\mathbf{x}(t)$, and the corresponding variation in the cost $\delta\varphi$ is given by

$$\begin{aligned}\delta\varphi = &\left[\left(\frac{\partial\varphi}{\partial\mathbf{x}} - \boldsymbol{\lambda}^T\right)\delta\mathbf{x}\right]_{t=T} + [\boldsymbol{\lambda}^T\delta\mathbf{x}]_{t=0} + \\ &+ \int_0^T \left[\left(\frac{\partial H}{\partial\mathbf{x}} + \dot{\boldsymbol{\lambda}}\right)\delta\mathbf{x} + \frac{\partial H}{\partial\mathbf{u}}\delta\mathbf{u}\right]\,dt\end{aligned}$$

This expression can be simplified by requiring the function $\boldsymbol{\lambda}(t)$, so far left unspecified, to satisfy conditions which will cause a number of terms to cancel out. It is easily seen that, if $\boldsymbol{\lambda}(t)$ satisfies the differential equation

$$\dot{\boldsymbol{\lambda}}^T = -\frac{\partial H}{\partial\mathbf{x}} = -\boldsymbol{\lambda}^T\frac{\partial\mathbf{f}}{\partial\mathbf{x}} \tag{2.11}$$

and the "initial conditions"

$$\boldsymbol{\lambda}(T) = \left(\frac{\partial\varphi}{\partial\mathbf{x}}\right)_{t=T}$$

then the expression for $\delta\varphi$ reduces to

$$\delta\varphi = \boldsymbol{\lambda}^T(0)\delta\mathbf{x}(0) + \int_0^T \frac{\partial H}{\partial\mathbf{u}}\delta\mathbf{u}\,dt \tag{2.12}$$

In our particular case, the initial conditions are fixed, so eqn. (2.12) can be further simplified to

$$\delta\varphi = \int_0^T \frac{\partial H}{\partial \mathbf{u}} \delta \mathbf{u}\, dt \tag{2.13}$$

The components of $\boldsymbol{\lambda}$ are called the *Lagrange multipliers* of the problem, and eqn. (2.11) is often referred to as the *adjoint equation* for (2.9). From eqn. (2.12) we see that $\boldsymbol{\lambda}(0)$ is the gradient of φ with respect to variations in the initial conditions :

$$\boldsymbol{\lambda}^T(0) = \frac{\partial \varphi}{\partial \mathbf{x}(0)} \tag{2.14}$$

The same property holds for any $t' \in [0,T]$, as can be seen from the following argument. Let $\mathbf{x}(t)$ be the solution of eqn. (2.9) on the interval $[0,T]$ subject to the initial conditions $\mathbf{x}(0) = \mathbf{x}_0$, and fix $t' \in [0,T]$. Then $\mathbf{x}(t)$ also solves (2.9) on $[t',T]$ with the "initial conditions" $\mathbf{x}(t') = \mathbf{x}(t')$. Therefore the Lagrange multipliers for the two problems are the same, and applying eqn. (2.14) to the second case, where the initial time is not *0* but *t'*, we conclude that

$$\boldsymbol{\lambda}^T(t') = \frac{\partial \varphi}{\partial \mathbf{x}(t')} \tag{2.15}$$

This property of the Lagrange multipliers will be used later to give an intuitive explanation of an integral formula that gives the gradient of φ with respect to variations in parameters appearing in eqn. (2.9).

To summarize, the derivative of φ with respect to $\mathbf{u}$ can be computed from eqn. (2.12) after solving the following differential equations :

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t)] \tag{2.16}$$

$$\dot{\boldsymbol{\lambda}}^T = -\boldsymbol{\lambda}^T \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \tag{2.17}$$

$$\mathbf{x}(0) = \mathbf{x}_0 \quad , \quad \lambda^T(T) = \left(\frac{\partial \varphi}{\partial \mathbf{x}}\right)_{\mathbf{x}=\mathbf{x}(T)}$$

This is formally a two-point boundary value problem, but in this particular case its solution can be found by first integrating eqn. (2.16) forward in time, and then eqn. (2.17) *backward*. Eqns. (2.16, 2.17) can be put in the equivalent form

$$\begin{aligned} \dot{\mathbf{x}} &= \frac{\partial H}{\partial \lambda} \\ \dot{\lambda}^T &= -\frac{\partial H}{\partial \mathbf{x}} \end{aligned} \tag{2.18}$$

This latter form is commonly known as the *Hamiltonian formulation* of the equations for the optimal control problem. In Chapter 4 we will prove that eqns. (2.18) can also be used to compute the derivative of φ with respect to variations in parameters appearing in the differential equation (2.9).

2.5.3 Pontryagin's Minimum Principle

An interesting consequence of eqn. (2.13) is that, if $\mathbf{u}$ is a minimizer for φ, we must have $\int_0^T \frac{\partial H}{\partial \mathbf{u}} \delta \mathbf{u}\, dt \geq 0$ for all admissible $\delta \mathbf{u}$. This suggests that the following statement might be true

Proposition 2.1 *Let $\mathcal{U}(t) \subseteq \mathrm{R}^m$ be the set*

$$\mathcal{U}(t) = \{\mathbf{u}(t) : \mathbf{u} \in \mathcal{U}\}$$

If $\mathbf{u}_0(t)$ solves the optimal control Problem (2.10), then it also satisfies the inequality

$$H[\mathbf{x}(t), \lambda(t), \mathbf{u}_0(t)] \leq H[\mathbf{x}(t), \lambda(t), \mathbf{u}(t)]$$

for all $\mathbf{u} \in \mathcal{U}(t)$ and for almost all $t \in [0, T]$.

In other words, the Hamiltonian is minimized along the optimal trajectory. This proposition is indeed true, as was first proven by Pontryagin [23]; for this reason it is commonly known as *Pontryagin's minimum principle.* It has many applications, both theoretical and practical, in classical optimal control problems. Especially interesting is the case when the function $\mathbf{f}(\mathbf{x}, \mathbf{u})$ is linear in the inputs

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{g}(\mathbf{x}) + \mathbf{A}\mathbf{u}$$

and the set $\mathcal{U}$ is defined as

$$\mathcal{U} = \{\mathbf{u}(t) : |u_i(t)| \leq M_i, i = 1, \ldots, m\}.$$

Under these assumptions Pontryagin's minimum principle can be used to give a very stringent characterization of the optimal control.

Corollary 2.1 *Let* $\mathbf{f}$ *and* $\mathcal{U}$ *be given as above, and let* $\mathbf{u}_0(t) = (u_{0,1}(t), \ldots, u_{0,m}(t))$ *be the control that minimizes* $\varphi[\mathbf{x}(t)]$. *Denote the i-th column of* $\mathbf{A}$ *by* $\mathbf{a}_i$. *Then, for almost all* $t \in [0, T]$, *either* $u_{0,i}(t) = M_i$ *or* $u_{0,i}(t) = -M_i$, *unless* $\boldsymbol{\lambda}^T(t)\mathbf{a}_i \equiv 0$.

Proof. In this particular case the Hamiltonian is given by

$$H(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}) = \boldsymbol{\lambda}^T\mathbf{g}(\mathbf{x}) + \boldsymbol{\lambda}^T\mathbf{A}\mathbf{u} = \boldsymbol{\lambda}^T\mathbf{g}(\mathbf{x}) + \sum_{i=1}^{m} \boldsymbol{\lambda}^T\mathbf{a}_i u_i$$

The corollary is then an immediate consequence of Pontryagin's minimum principle. □

If $\boldsymbol{\lambda}^T(t)\mathbf{a}_i \equiv 0$, no conclusions about $u_{0,i}$ can be drawn from the above corollary; in this case, the control $\mathbf{u}_0$ is called *singular.*

Chapter 3

The Circuit Equations

3.1 Preliminary Assumptions

Throughout the rest of this dissertation I will restrict my attention to circuits satisfying the following assumptions :

- The only elements contained in the circuit are independent current sources, resistors, capacitors and voltage-controlled current sources.
- Branch currents and branch voltages are given the standard orientation (positive current flowing from the positive to the negative node) *except* in the case of independent current sources, for which the opposite convention is assumed.
- Resistors, capacitors and controlled sources are not required to be linear. However it is assumed that nonlinear resistors and controlled sources are voltage controlled, i.e. they can be described by branch equations of the form $i_d = f_d(v_d)$, where i_d is the current flowing through the device, f_d is a function of class C^1

which depends on the device and v_d is the controlling voltage. Similarly it is assumed that nonlinear capacitors can be described by equations of the type $q_d = g_d(v_d)$, where q_d is the charge on the capacitor, g_d is a function of class C^1 depending on the device and v_d is the voltage across the capacitor. Furthermore we assume that there exists a constant $c_{min} > 0$ such that $\frac{dq_d}{dv_d} \geq c_{min}$ for *all* voltages v_d and for *all* capacitors in the circuit.

- There are capacitors connecting the ground node to all other nodes of the circuit.

3.2 Charge Formulation of the Node Equations

To analyze the behavior of a circuit belonging to the class defined in the previous section we can use Node Analysis [24]. This means that we write Kirchhoff's Current Law at each node in the circuit other than the ground node. For our purposes it is convenient to give the resulting equations the following particular form.

Let $\mathcal{G}_C$ be the graph generated by the capacitive elements of the circuits (i.e. there is an edge connecting nodes i and j of $\mathcal{G}_C$ if and only if there is a capacitive element connecting nodes i and j of the circuit). Because of the assumptions made in the previous section, there is an edge of $\mathcal{G}_C$ connecting each node to ground. Give those edges the customary orientation (negative sign to the ground node); give arbitrary orientation to the remaining edges of $\mathcal{G}_C$. For a concrete example look at Fig. 3.1, which shows a very simple network and its capacitance graph.

For each node i in the circuit different from the ground node define the following

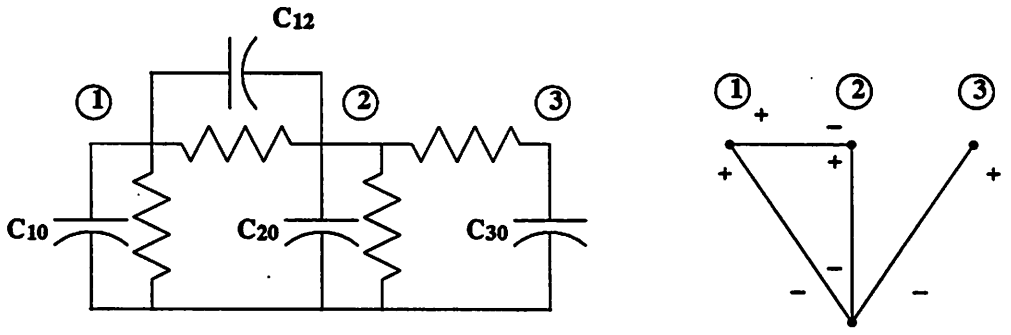


Figure 3.1: An RC network and its capacitance graph

quantity

$$q_i \triangleq \sum_j e_{ij} q_{ij} \tag{3.1}$$

where q_{ij} is the charge corresponding to the capacitive element connecting node i to node j, and the index j ranges over the set of nodes that are connected to node i by an edge of $\mathcal{G}_C$. In the above sum $e_{ij} = 1$ if the edge of $\mathcal{G}_C$ connecting i to j is oriented so that the positive sign corresponds to node i; otherwise $e_{ij} = -1$. Because of our assumptions the sum always contains the term $+q_{i0}$ corresponding to the capacitive element connecting node i to ground.

Let us look again at Fig. 3.1 for an example. The network contains three node in addition to the ground nodes, so we have three variables : q_1, q_2 and q_3. Using the above rules we obtain the following equations :

$$\begin{aligned} q_1 &= q_{10} + q_{12} \\ q_2 &= q_{20} - q_{12} \\ q_3 &= q_{30} \end{aligned}$$

With these conventions, it is easy to see that the quantity $\frac{dq_i}{dt}$ corresponds to the

current flowing out of node i through the capacitive elements. For convenience, we will call q_i the *charge at node i*, and the vector $\mathbf{q} = (q_1, \ldots, q_n)^T$ will be referred to as the *vector of the node charges.*

Kirchhoff's current law at node i generates an equation of the form

$$\sum_j e_{ij} i_{ij} = 0$$

where i_{ij} is the current flowing through an element connecting nodes i and j, and $e_{ij} = 1$ or $e_{ij} = -1$ according to the same convention used above. Let us split the sum in three separate terms, corresponding respectively to the capacitive elements, the resistive elements and the independent current sources. Then the above equation becomes

$$\sum_{cap.elts} \pm i_{ij} + \sum_{res.elts} \pm i_{ij} - \sum_{sources} \pm i_{ij} = 0$$

(the sign – before the last sum is due to the fact that independent sources have opposite orientation respect to the other elements). The first sum is equal to $\dot{q}_i$, as explained in the previous paragraph, while for each term in the second sum we have $i_{ij} = f_{ij}(v_i - v_j)$. Therefore the equation generated by Kirchhoff's current law at node i can be put in the following form

$$\dot{q}_i + f_i(v_1, \ldots, v_n) = i_i$$

where $f_i = \sum_{res.elts} \pm f_{ij}$ and $i_i = \sum_{sources} \pm i_{ij}$. Gathering all the equations for $i = 1, \ldots, n$ we obtain a set of differential equations of the form

$$\dot{\mathbf{q}} = -\mathbf{f}(\mathbf{v}) + \mathbf{i}$$

Similarly for each q_{ij} appearing in eqn. (3.1) we have $q_{ij} = g_{ij}(v_i - v_j)$, which means

that q_i has an expression of the form

$$q_i = g_i(v1, \ldots, v_n)$$

In this way we obtain a set of algebraic equations that can be written as :

$$\mathbf{q} = \mathbf{g}(\mathbf{v})$$

We conclude that the dynamical equations describing our circuit can be put in the following form :

$$\dot{\mathbf{q}} = -\mathbf{f}(\mathbf{v}) + \mathbf{i} \tag{3.2}$$

$$\mathbf{q} = \mathbf{g}(\mathbf{v}) \tag{3.3}$$

As is, eqn. (3.2) contains n differential equations in $2n$ unknowns (the vectors $\mathbf{q}$ and $\mathbf{v}$). We will now show that eqn. (3.3) can be inverted, i.e. that there exists a function $\mathbf{g}^{-1}$ of class C^1 such that $\mathbf{v} = \mathbf{g}^{-1}(\mathbf{q})$. This means that eqn. (3.2) can be put in *normal form* :

$$\dot{\mathbf{q}} + \mathbf{f}[\mathbf{g}^{-1}(\mathbf{q})] = \mathbf{i} \tag{3.4}$$

The existence of a normal form for the differential equations (3.2,3.3) is useful from a theoretical point of view, because then Theorems 2.8 and 2.9 can be applied to eqn. (3.4) (assuming that $\mathbf{f}^{-1}[\mathbf{g}(\mathbf{q})]$ is Lipschitz continuous). However from a computational point of view it is preferable to integrate eqns. (3.2,3.3) directly.

I begin by reporting a definition and a few theorems taken from [25, pp. 141 – 145], where all the proofs can be found.

Definition 3.1 *Let* $\mathbf{g} : D \to \mathrm{R}^n$ *be a* C^1 *function, where* D *is an open subset of* R^n.

Then $\mathbf{g}$ *is* **monotone** *in* D *if*

$$[\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y})]^T(\mathbf{x} - \mathbf{y}) \geq 0 \qquad \forall \mathbf{x}, \mathbf{y} \in D. \tag{3.5}$$

$\mathbf{f}$ *is* **strictly monotone** *on* D *if the above inequality is strict whenever* $\mathbf{x} \neq \mathbf{y}$. $\mathbf{f}$ *is* **uniformly monotone** *on* D *if there exists a* $\gamma > 0$ *such that*

$$[\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{y})]^T(\mathbf{x} - \mathbf{y}) \geq \gamma(\mathbf{x} - \mathbf{y})^T(\mathbf{x} - \mathbf{y}) \;\; \forall \mathbf{x}, \mathbf{y} \in D. \tag{3.6}$$

The following statements are obvious:

- A function which is strictly monotone on D is one-to-one on D (i.e. $\mathbf{g}(\mathbf{x}) \neq \mathbf{g}(\mathbf{y})$ whenever $\mathbf{x} \neq \mathbf{y}$).
- A uniformly monotone function is also strictly monotone.

Theorem 3.1 *Let* $\mathbf{g} : D \to \mathrm{R}^n$ *be of class* C^1 *on the open convex set* $D \subseteq \mathrm{R}^n$. *Then*

- $\mathbf{g}$ *is monotone on* D *if and only if* $\frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ *is positive semidefinite for all* $\mathbf{x} \in D$.
- *If* $\frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ *is positive definite for all* $\mathbf{x} \in D$ *then* $\mathbf{g}$ *is strictly monotone on* D.
- $\mathbf{g}$ *is uniformly monotone on* D *if and only if there is a* $\gamma > 0$ *such that*

$$\Delta \mathbf{x}^T \frac{\partial \mathbf{g}}{\partial \mathbf{x}} \Delta \mathbf{x} \geq \gamma \|\Delta \mathbf{x}\|^2 \qquad \forall \mathbf{x} \in D, \; \forall \Delta \mathbf{x} \in \mathrm{R}^n$$

Corollary 3.1 *If* $\mathbf{g} : D \to \mathrm{R}^n$ *is of class* C^1 *on the open convex set* $D \subseteq \mathrm{R}^n$ *and* $\frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ *is positive definite for all* $\mathbf{x} \in D$, *then* $\mathbf{g}$ *is one-to-one on* D.

Theorem 3.2 *If* $\mathbf{g} : \mathrm{R}^n \to \mathrm{R}^n$ *is of class* C^1 *an uniformly monotone on* R^n, *then* $\mathbf{g}$ *is a homeomorphism of* R^n *onto* R^n, *i.e.* $\mathbf{g}$ *has a* C^1 *inverse* $\mathbf{g}^{-1} : \mathrm{R}^n \to \mathrm{R}^n$.

Let us now apply these theorems to the function $\mathbf{g}(\mathbf{v})$ appearing in eqn. (3.3). The derivative $\mathbf{C}(\mathbf{v}) = \frac{\partial \mathbf{g}}{\partial \mathbf{v}}(\mathbf{v})$ is by definition the *capacitance matrix* of the circuit. A capacitive element connected between nodes i and j of the circuit contributes a quantity $+\frac{dq_{ij}}{dv_{ij}}$ to the elements c_{ii} and c_{jj} of $\mathbf{C}(\mathbf{v})$ and a quantity $-\frac{dq_{ij}}{dv_{ij}}$ to the elements c_{ij} and c_{ji}; a capacitive element between node i and ground contributes a quantity $+\frac{dq_{i0}}{dv_i}$ to the element c_{ii}. Because it is assumed that $\frac{dq}{dv} \geq c_{min}$ for all capacitive elements and that there is a capacitive element from each node to ground, all the diagonal elements of $\mathbf{C}(\mathbf{v})$ are positive, all the off-diagonal elements are negative (or zero), and the matrix $\mathbf{C}(\mathbf{v})$ is diagonally dominant and positive definite. Furthermore we can prove that

$$\Delta \mathbf{v}^T \mathbf{C}(\mathbf{v}) \Delta \mathbf{v} \geq c_{min} \|\Delta \mathbf{v}\|^2$$

for every vector $\Delta \mathbf{v} = (\Delta v_1, \ldots, \Delta v_n)^T$. This can be done by brute force algebra or by reasoning as follows. Let us fix $\mathbf{v}$ and let us replace every nonlinear capacitive element with a linear capacitance in the following way : if the branch equation of the nonlinear capacitance is $q_d = f_d(v_d)$, the value of the linear capacitance replacing it is $\frac{df_d}{dv_d}$. In this way way we obtain a linear capacitive network whose capacitance matrix is exactly $\mathbf{C}(\mathbf{v})$. If we now apply voltages $\Delta v_1, \ldots, \Delta v_n$ to the nodes of the network, the energy stored in all the capacitances is $\frac{1}{2} \Delta \mathbf{v}^T \mathbf{C}(\mathbf{v}) \Delta \mathbf{v}$. This is certainly larger than the energy stored on the grounded capacitances, which is equal to $\frac{1}{2} \sum_{i=1}^{n} c_{ii} \Delta v_i^2$. But $c_{ii} \geq c_{min}$, so that $\frac{1}{2} \sum_{i=1}^{n} c_{ii} \Delta v_i^2 \geq \frac{1}{2} c_{min} \|\Delta \mathbf{v}\|^2$, which proves our claim. Theorem 3.2 can now be applied to the function $\mathbf{g}(\mathbf{v})$ and we can state the following

Proposition 3.1 *The function* $\mathbf{q} = \mathbf{g}(\mathbf{v})$ *has an inverse* $\mathbf{v} = \mathbf{g}^{-1}(\mathbf{q})$ *of class* $\mathbf{C}^1$ *which is defined for all* $\mathbf{q} \in \mathbf{R}^n$.

As mentioned before, this means that eqns. (3.2,3.3) can be put in normal form (3.4), at least from a theoretical point of view. We will use this fact in Chapter 4.

3.3 Numerical Integration of the Circuit Equations

3.3.1 General Properties of Numerical Integration Methods

The issues involved in the numerical solution of a differential equation are countless, and the literature that deals with this topic is immense. Here I will only mention briefly those aspects that are relevant to my specific case, trying at the same time to make the reader aware of the reasons underlying my selection of a particular integration method. Throughout this section, I will assume that an approximation to a scalar function $x(t)$ satisfying the *initial value problem*

$$\begin{aligned} \dot{x} &= f(x,t) \\ x(0) &= x_0 \end{aligned} \tag{3.7}$$

has to be computed over a finite time interval $[0,T]$. The extension to vector-valued functions can be done without difficulty.

Almost all numerical integration methods generate a finite sequence $\hat{x}(t_i), i = 0,\ldots,N$ where $\hat{x}(t_i)$ approximates $x(t_i)$ with a certain accuracy. The quantity $x(t_i) - \hat{x}(t_i)$ is called the *global error* at time t_i, and in general it is not possible to reduce it to zero. However it is natural to expect it to tend to zero as the *mesh* $\{0 = t_0, t_1, \ldots, t_N = T\}$ thickens. This leads to the following definition [26, p. 172] :

Definition 3.2 *Let us define* $h = \max_{1\leq i\leq N}(t_i - t_{i-1})$. *An integration method is*

convergent *if*

$$\lim_{h \to 0} \max_{0 \le i \le N} \|x(t_i) - \hat{x}(t_i)\| = 0$$

The global error is the most natural way to describe the accuracy of an integration method; unfortunately it is practically impossible to estimate it in any useful way. For this reason another quantity, called the *local truncation error* [26, p. 59] (often abbreviated LTE), is used in its place.

Definition 3.3 *The* **local truncation error** *at time t_i is the quantity $x(t_i) - \hat{x}(t_i)$ computed under the condition that $x(t_j) = \hat{x}(t_j)$ for $j = 0, \ldots, i-1$.*

It is intuitive that the local truncation error at time t_i depends on the timestep $h_i = t_i - t_{i-1}$. Therefore one of the most widely used techniques to control the accuracy of an integration method is to estimate the local truncation error at each timepoint and to check that it does not exceed certain bounds. This consideration favors the choice of a method whose local truncation error can be estimated easily. We will see that this can be done trivially for a particular class of methods which are commonly referred to as Backward Differentiation Formulae (or BDF).

3.3.2 Linear Multistep Methods

The Backward Differentiation Formulae belong to a wider class of methods known as Linear Multistep Methods. The name arises from the fact that the sequence $\{\hat{x}(t_i)\}$ generated by one of these methods is required to satisfy a linear recurrence relation of the form

$$\sum_{j=0}^{p} a_j \hat{x}(t_{i-j}) = h_i \sum_{j=0}^{p} b_j f[\hat{x}(t_{i-j}), t_{i-j}] \tag{3.8}$$

where $a_0 = 1$. Assuming that $\hat{x}(t_{i-1}), \ldots, \hat{x}(t_{i-p})$ are available , eqn. (3.8) can be used to compute $\hat{x}(t_i)$. If $b_0 = 0$, this can be done trivially, because then

$$\hat{x}(t_i) = -\sum_{j=1}^{p} a_j \hat{x}(t_{i-j}) + h_i \sum_{j=1}^{p} b_j f[\hat{x}(t_{i-j}), t_{i-j}]$$

If $b_0 \neq 0$, computing $\hat{x}(t_i)$ requires the solution of the following nonlinear equation

$$\hat{x}(t_i) - b_0 f[\hat{x}(t_i), t_i] = -\sum_{j=1}^{p} a_j \hat{x}(t_{i-j}) + h_i \sum_{j=1}^{p} b_j f[\hat{x}(t_{i-j}), t_{i-j}]$$

For this reason linear multistep methods for which $b_0 = 0$ are called *explicit*, while the others are called *implicit*. The coefficients $a_j, b_j, \quad j = 0, \ldots, p$ cannot be chosen arbitrarily. For instance it is customary to impose the *exactness constraints of order k* [27, p. 480], where $k \leq 2p+2$: if $x(t)$ is a polynomial of degree k or less, it is required that $x(t_i) = \hat{x}(t_i)$ for all t_i's. If the timestep $t_i - t_{i-1}$ is constant, this requirement translates into the following equations :

$$\begin{aligned} \sum_{j=0}^{p} a_j &= 0 \\ \sum_{j=0}^{p} (-j)^l a_j + l \sum_{j=0}^{p} (-j)^{l-1} b_j &= 0 \qquad l = 1, \ldots, k \end{aligned}$$

An important characteristic of a linear multistep method is its *region of absolute stability*. The motivation for its definition will become clear after looking at the following example. Suppose that the differential equation

$$\dot{x} = -2.02x \tag{3.9}$$

has to be integrated over the interval $[0, 10]$ with the initial condition $x(0) = 1$. We want to compare the results given by two linear multistep methods : the first one is known as the *Forward Euler* method, and is defined by the recurrence relation

$$\hat{x}(t_i) = \hat{x}(t_{i-1}) + h_i f[\hat{x}(t_{i-1}), t_{i-1}] \tag{3.10}$$

while the second is the *Backward Euler* method, defined by

$$\hat{x}(t_i) = \hat{x}(t_{i-1}) + h_i f[\hat{x}(t_i), t_i] \tag{3.11}$$

In both cases, we use a fixed timestep $t_i - t_{i-1} = 1$. The points generated by both methods are plotted in Fig. 3.2, together with the values of the exact solution $x(t_i) = \exp(-2.02t_i)$. It can be seen that the difference between the two integration methods is not simply a matter of accuracy, but also a matter of different *qualitative* behavior of the corresponding sequences : the Backward Euler method generates points that approach zero as $t_i \to \infty$, and thus resemble the general qualitative behavior of the exact solution. On the other hand the Forward Euler method generates an oscillating and divergent sequence.

If the experiment is repeated with a smaller stepsize $t_i - t_{i-1} = 0.25$, we obtain the results plotted in Fig. 3.3. In this case the qualitative behavior of the sequences generated by both integration methods is the same. This simple example suggests that the timestep size can have a remarkable effect on the behavior of an integration method. The definition of stability region formalizes this fact.

Definition 3.4 *Consider the differential equation*

$$\begin{aligned} \dot{x} &= -\lambda x \\ x(0) &= x_0 \neq 0 \end{aligned} \tag{3.12}$$

where λ is a complex number. Let $\hat{x}(t_i)$ be the sequence generated by a linear multistep method using a fixed timestep $t_i - t_{i-1} = h$. The **region of absolute stability** S *of the method is that subset of the complex plane such that if $h\lambda \in S$ then $\lim_{i\to\infty} \hat{x}(t_i) = 0$.*

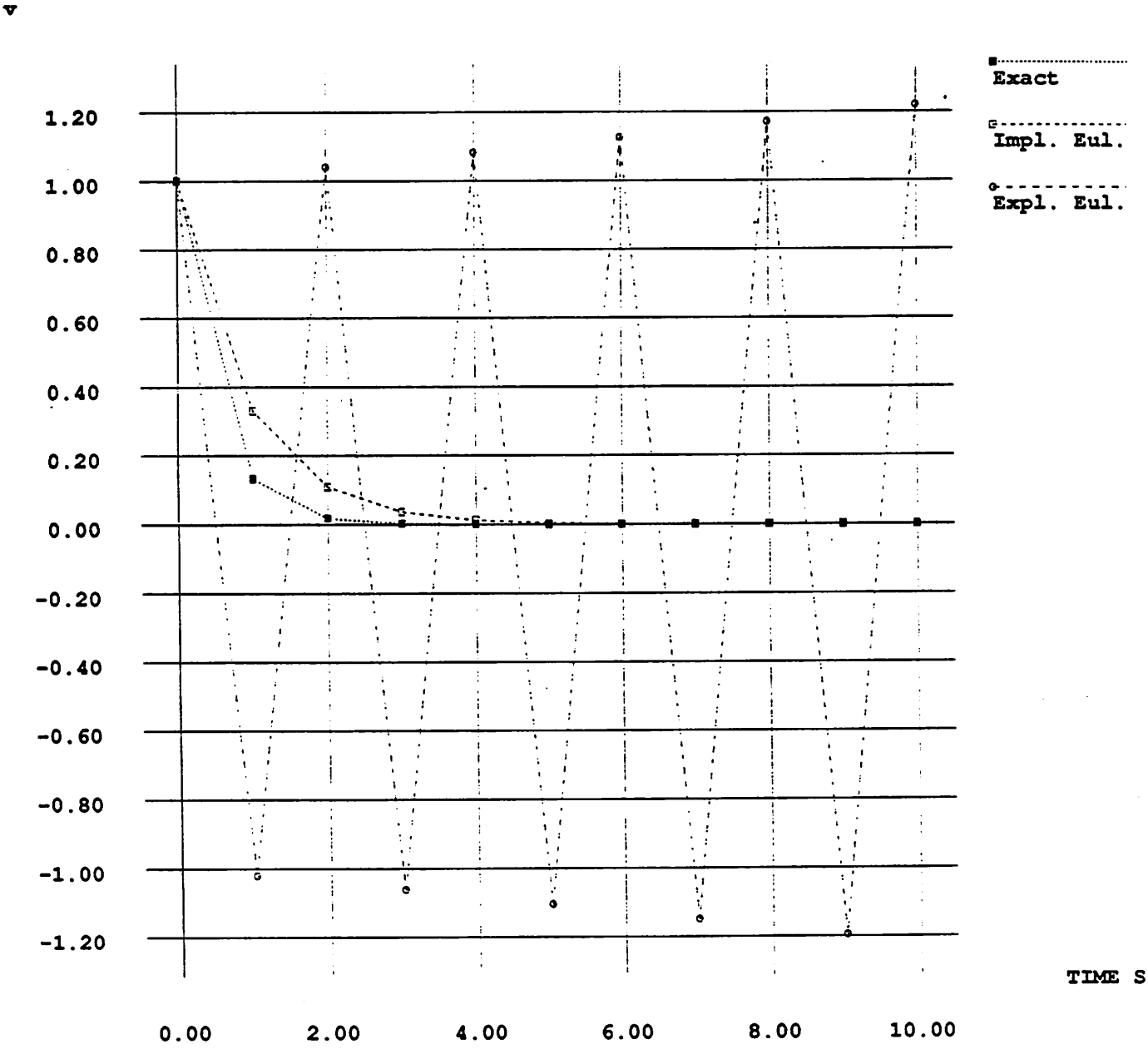


Figure 3.2: Numerical integration with timestep 1.

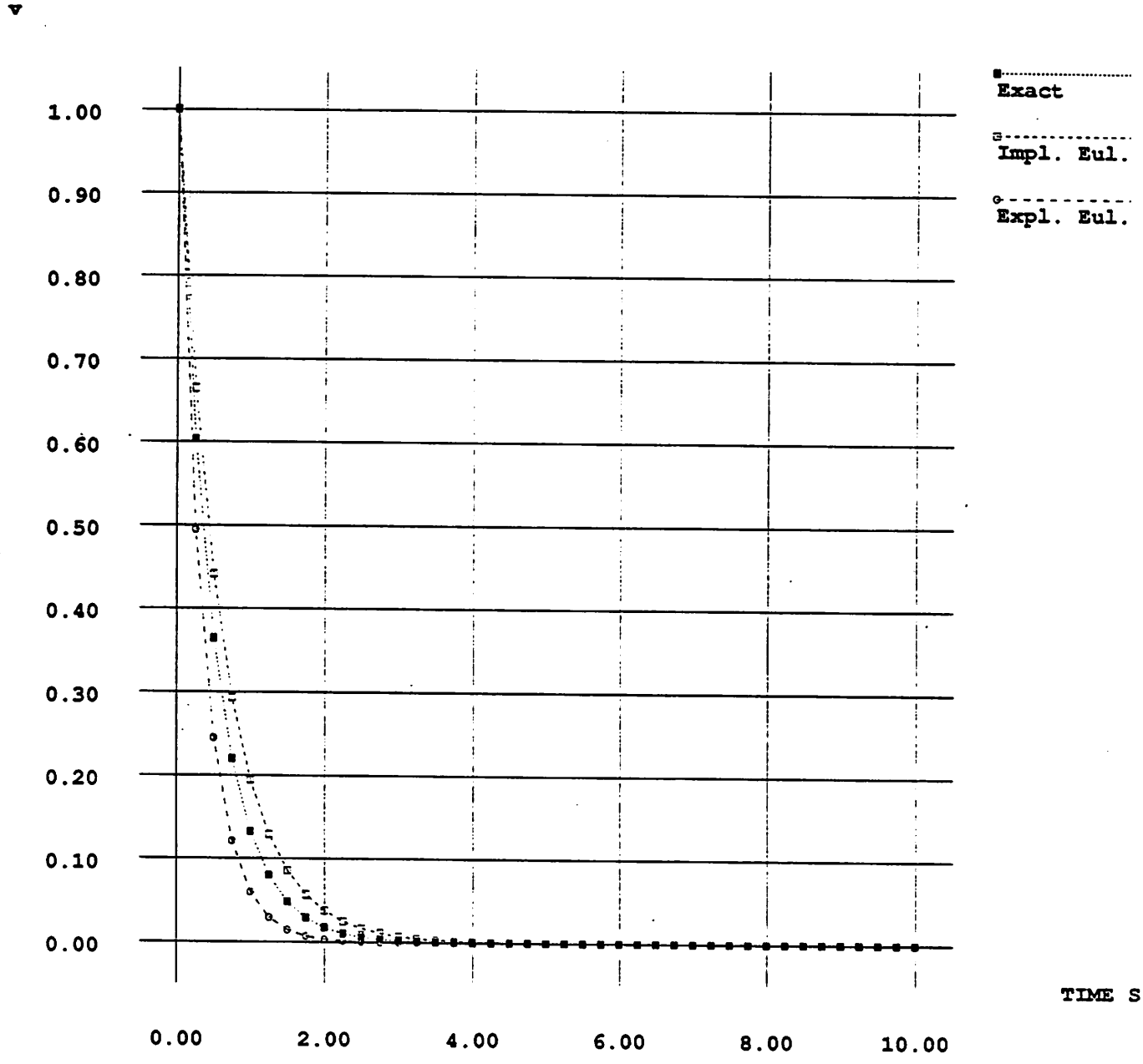


Figure 3.3: Numerical integration with timestep 0.25.

The region of absolute stability of a linear multistep method can be computed fairly easily [27]. As an example, the regions of absolute stability of the Forward and Backward Euler methods are shown in Fig. 3.4. If we pick $\lambda = 2.02$ and $h = 1$, we see that $h\lambda = 2.02$ falls inside the absolute stability region of the Backward Euler method, but just outside that of the Forward Euler method, while if $h = 0.25$ then $h\lambda = 0.505$ is inside both regions, confirming the results that we obtained in our simple test example.

It is now clear that the region of absolute stability of an integration method can play an important role in determining the largest timestep that is safe to use. Methods whose regions of absolute stability do not pose any restrictions on the timestep size are therefore of particular interest. Because most differential equations of interest are stable, this translates into the requirement that the region of absolute stability should include the whole right half-plane of C. This is formalized in the following definition

Definition 3.5 *A linear multistep method is* **A-stable** *if its region of absolute stability includes the right half-plane of C.*

Thus A-stable methods have the desirable property of having their timestep size limited only by accuracy requirements. A more refined analysis of the properties of integration methods [26] shows that this property is shared by a wider class of methods, called **stiffly stable.**

Definition 3.6 *A linear multistep method is* **stiffly stable** *if there exist numbers* $\delta \geq 0$, $\mu > 0$ *and* $\theta > 0$ *such that, with reference to Fig. 3.5, the method is accurate in regions I and II and stable in region III.*

At the end of the previous section we also mentioned that a particular class of

linear multistep methods, called Backward Differentiation Formulae, are particularly attractive because the local truncation error can be estimated very easily. The next section contains a precise definition of those methods and a brief illustration of their stability properties.

3.3.3 Backward Differentiation Formulae

The Backward Differentiation Formulae are obtained from eqn. (3.8) by setting $b_0 = 1$ and $b_1 = b_2 = \ldots = b_p = 0$:

$$f[\hat{x}(t_i), t_i] = \frac{1}{h_i} \sum_{j=0}^{p} a_j \hat{x}(t_{i-j}) \tag{3.13}$$

The coefficients a_j are determined by the requirement that $\hat{x}(t_i) = x(t_i)$ whenever $x(t)$ is a polynomial of degree p or less. Because $f[x(t_i), t_i] = \dot{x}(t_i)$, the requirement $\hat{x}(t_i) = x(t_i)$ means that the equation

$$\dot{x}(t_i) = \frac{1}{h_i} \sum_{j=0}^{p} a_j x(t_{i-j}) \tag{3.14}$$

must be satisfied *for all polynomials of degree less than or equal to p* (hence the name of Backward Differentiation Formulae). It can be shown [27] that this is equivalent to asking that the coefficients a_j satisfy the following set of linear equations

$$\begin{pmatrix} 1 & 1 & \ldots & 1 \\ 0 & \frac{t_i - t_{i-1}}{h_i} & \ldots & \frac{t_i - t_{i-p}}{h_i} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \left(\frac{t_i - t_{i-1}}{h_i}\right)^p & \ldots & \left(\frac{t_i - t_{i-p}}{h_i}\right)^p \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_p \end{pmatrix} = \begin{pmatrix} 0 \\ -1 \\ \vdots \\ 0 \end{pmatrix}$$

Because $b_0 = 1$, BDF methods are implicit; this means that a nonlinear equation must be solved every time $\hat{x}(t_i)$ has to be computed. Most algorithms used to solve

nonlinear equations, such as the Newton-Raphson method [27], require an approximation to the solution as a starting point; the closer the initial approximation, the faster the algorithm will solve the nonlinear equations. For this reason it is convenient to have an *a-priori* estimate of $\hat{x}(t_i)$, which can be computed using a so-called *predictor method.* The predictor commonly used in conjunction with BDF methods is an interpolating polynomial through the points $x(t_{i-1}), \ldots, x(t_{i-p-1})$:

$$x^P(t_i) = \sum_{j=0}^{p} \gamma_j \hat{x}(t_{i-j-1}) \tag{3.15}$$

Aside from simplicity, the main advantage of using this predictor comes from the following theorem [28] :

Theorem 3.3 *Let $x^P(t_i)$ and $\hat{x}(t_i)$ be computed from eqns. (3.15) and (3.13) respectively. Neglecting higher order terms, the local truncation error at t_i is given by*

$$LTE(t_i) \cong \frac{t_i - t_{i-1}}{t_i - t_{i-p-1}}[\hat{x}(t_i) - x^P(t_i)]$$

Using the above formula the local truncation error can be computed trivially, in contrast to other approaches which require approximating higher order derivatives of the solution with finite differences, a computationally expensive and notoriously inaccurate process.

We now turn to the question of stability of BDF methods. We will not enter into the computational details, which can be found elsewhere [27]. For our purposes it suffices to show the regions of absolute stability of BDF methods with p from one through four (Fig. 3.6). We see that all of them are stiffly stable and that methods with $p = 1$ and $p = 2$ are A-stable. The BDF method with $p = 2$ offers a good compromise between

stability, accuracy and storage requirements and is therefore a reasonable choice for the numerical integration of eqns. (3.2,3.3).

3.4 The Adjoint Network

An often occurring problem in circuit design is to minimize or maximize a certain *performance function* of a network (e.g. power consumption) by giving appropriate values to some design parameters (for instance, the values of selected elements of the network). This goal can be achieved using an optimization algorithm, provided that the derivatives of the performance function with respect to the parameters are available. Several algorithms for the computation of those derivatives (often called *network sensitivities*) have appeared in the literature. We will conclude this chapter by illustrating briefly one of them, known as the Adjoint Network Method [27]. The reason for this digression is that from a mathematical point of view our approach to macromodeling has many points in common with the network performance minimization problem; in particular, the formulas for the computation for the derivatives are the same in both cases. Chapter 4 contains a rigorous mathematical justification of this fact, but at this point we would like to interpret the computation of network sensitivities from the point of view of circuit theory. This can be done by introducing the concept of adjoint network, and showing that the computation of sensitivities is equivalent to performing an analysis on it. We will confine ourselves to the stationary case, even if the macromodeling problem involves the computation of derivatives in the time domain, because the underlying ideas remain the same. A detailed treatment of the Adjoint Network

Method in its most general setting can be found in [29]. Here we follow a simplified approach along the lines of [30].

Suppose that we are given the equation

$$\mathbf{f}(\mathbf{v}) = \mathbf{0} \tag{3.16}$$

where $\mathbf{f} : \mathbf{R}^n \rightarrow \mathbf{R}^n$ is of class $\mathbf{C}^1$. Let $\mathbf{v}_0$ be a solution of eqn. (3.16), and assume that $\frac{\partial \mathbf{f}}{\partial \mathbf{v}}(\mathbf{v}_0)$ is nonsingular. Let $c : \mathbf{R}^n \rightarrow \mathbf{R}$ be a real-valued function also of class $\mathbf{C}^1$. Assume now that $\mathbf{f}$ is perturbed by a small amount $\delta\mathbf{f}$, and let $\delta\mathbf{v}$ and δc be the corresponding perturbation of $\mathbf{v}_0$ and $c(\mathbf{v}_0)$. Our goal is to give an expression for δc as a function of $\delta\mathbf{f}$. Because

$$\delta c = \frac{\partial c}{\partial \mathbf{v}}(\mathbf{v}_0)\delta\mathbf{v} \tag{3.17}$$

we have to compute $\delta\mathbf{v}$ first. This can be done in the following way : we have

$$\mathbf{f}(\mathbf{v}_0 + \delta\mathbf{v}) + \delta\mathbf{f}(\mathbf{v}_0 + \delta\mathbf{v}) = \mathbf{0}$$

Taking a Taylor expansion for $\mathbf{f}$ and $\delta\mathbf{f}$ and neglecting terms of order higher that one, the previous equation becomes

$$\frac{\partial \mathbf{f}}{\partial \mathbf{v}}(\mathbf{v}_0)\delta\mathbf{v} + \delta\mathbf{f}(\mathbf{v}_0) = \mathbf{0}$$

because $\mathbf{f}(\mathbf{v}_0) = \mathbf{0}$. This equation can be readily solved for $\delta\mathbf{v}$:

$$\delta\mathbf{v} = -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right)^{-1}(\mathbf{v}_0)\delta\mathbf{f}(\mathbf{v}_0)$$

Substituting this expression in eqn. (3.17) we obtain the formula we were looking for :

$$\delta c = -\frac{\partial c}{\partial \mathbf{v}}(\mathbf{v}_0)\left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right)^{-1}(\mathbf{v}_0)\delta\mathbf{f}(\mathbf{v}_0) \tag{3.18}$$

We now want to put this equation in a slightly different form : let λ be the solution of

$$\left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right)^T \lambda = -\left(\frac{\partial c}{\partial \mathbf{v}}\right)^T$$

where it is understood that all the derivatives are to be evaluated at $\mathbf{v}_0$. It is easily seen that eqn. (3.18) can be rewritten as

$$\delta c = \lambda^T \delta \mathbf{f}(\mathbf{v}_0) \tag{3.19}$$

The advantage of writing the equation in this second form is that λ can be computed independently of $\delta \mathbf{f}$. Once λ is known, eqn. (3.19) can be used to calculate δc for an arbitrarily given $\delta \mathbf{f}$.

Consider now a network $\mathcal{N}$ containing only linear resistors and current sources which are either constant or linearly voltage-controlled. Let $\mathbf{v}$ be the vector of node voltages, and let $c(\mathbf{v})$ be a performance function depending on the dc solution $\mathbf{v}_0$ of the network. We want to compute first-order variations of $c(\mathbf{v}_0)$ corresponding to small perturbations of one or more elements of the network. Specializing the conclusions of the previous paragraph to this case we have $\mathbf{f}(\mathbf{v}) = \mathbf{Y}\mathbf{v} - \mathbf{i}$, where $\mathbf{Y}$ is the node-admittance matrix of the network [27] and $\mathbf{i}$ is the vector representing the independent current sources. The computation of λ requires solving the linear system

$$\mathbf{Y}^T \lambda = -\left(\frac{\partial c}{\partial \mathbf{v}}\right)^T (\mathbf{v}_0) \tag{3.20}$$

We will now show that there is a network whose node-admittance matrix is $\mathbf{Y}^T$: it is called the *adjoint network* of $\mathcal{N}$.

Recall that the contribution of each element of $\mathcal{N}$ to $\mathbf{Y}$ can be described by an *element stamp* or *pattern* [31]; for instance, the element pattern for a linear resistor of

	i	j
i	G	$-G$
j	$-G$	G

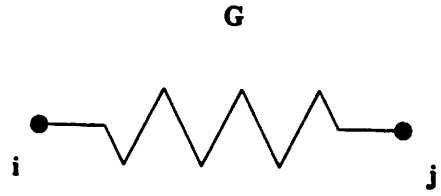


Table 3.1: Element pattern for a linear resistor

conductance G connecting nodes i and j is shown in Table 3.1. This means that this resistor contributes a quantity $+G$ to the elements y_{ii} and y_{jj} of $\mathbf{Y}$ and a quantity $-G$ to the elements y_{ij} and y_{ji}. Therefore to build the adjoint network of $\mathcal{N}$ we can take every element of $\mathcal{N}$ and replace it with another element having a pattern which is the transposed of the original one. The pattern of a linear resistor is symmetric, so to every linear resistor of $\mathcal{N}$ corresponds a linear resistor of equal value in the adjoint network. On the other hand the pattern for a linear voltage-controlled current source of transconductance G_M, shown in Table 3.2, is not symmetric. The transposed

	k	l
i	G_M	$-G_M$
j	$-G_M$	G_M

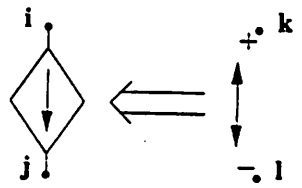


Table 3.2: Element pattern for a voltage-controlled current source

pattern is shown in Table 3.3 and it is readily seen that the corresponding element is

	i	j
k	G_M	$-G_M$
l	$-G_M$	G_M

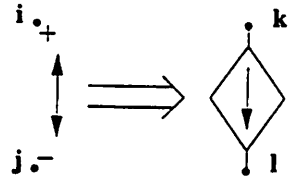


Table 3.3: The transposed pattern for a voltage-controlled current source

another voltage-controlled current source with the controlling and controlled branches interchanged. Therefore voltage-controlled current sources of $\mathcal{N}$ are reversed in the adjoint network.

Eqn. (3.20) also shows that the independent sources in the adjoint network are related to the performance function : if c depends on v_i, then the adjoint network contains and independent current source between node i and ground whose value is $-\frac{\partial c}{\partial v_i}$.

We can therefore conclude that the computation of network sensitivities is equivalent to performing an analysis of the adjoint network. Although these results have been derived assuming stationarity, they are representative of a more general fact. In Chapter 4 the macromodeling problem will be formulated as an optimization problem in the time domain, and it will be proven that the derivatives of the function to be performed can be computed by performing a time-domain analysis of the adjoint network.

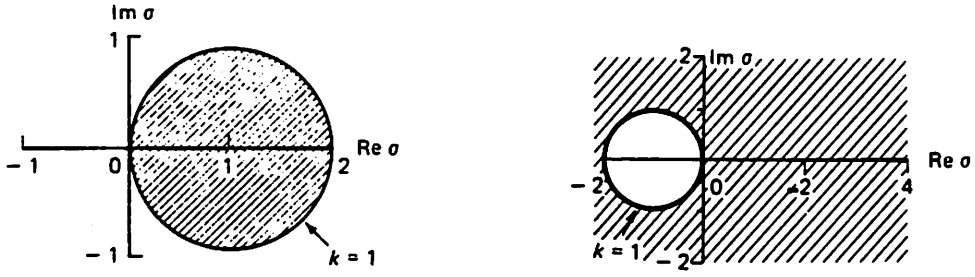


Figure 3.4: Stability regions (shaded areas) for Forward and Backward Euler methods.

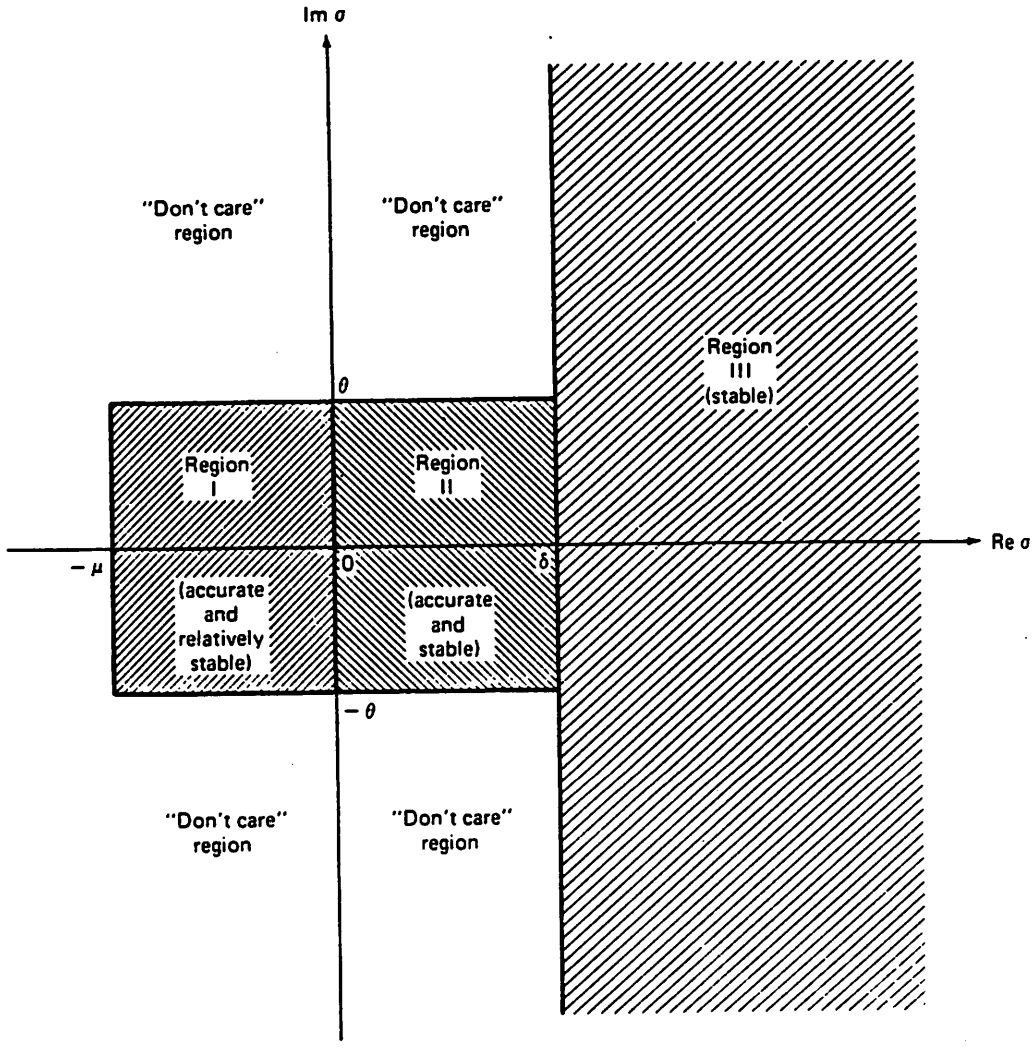


Figure 3.5: Definition of stiff stability

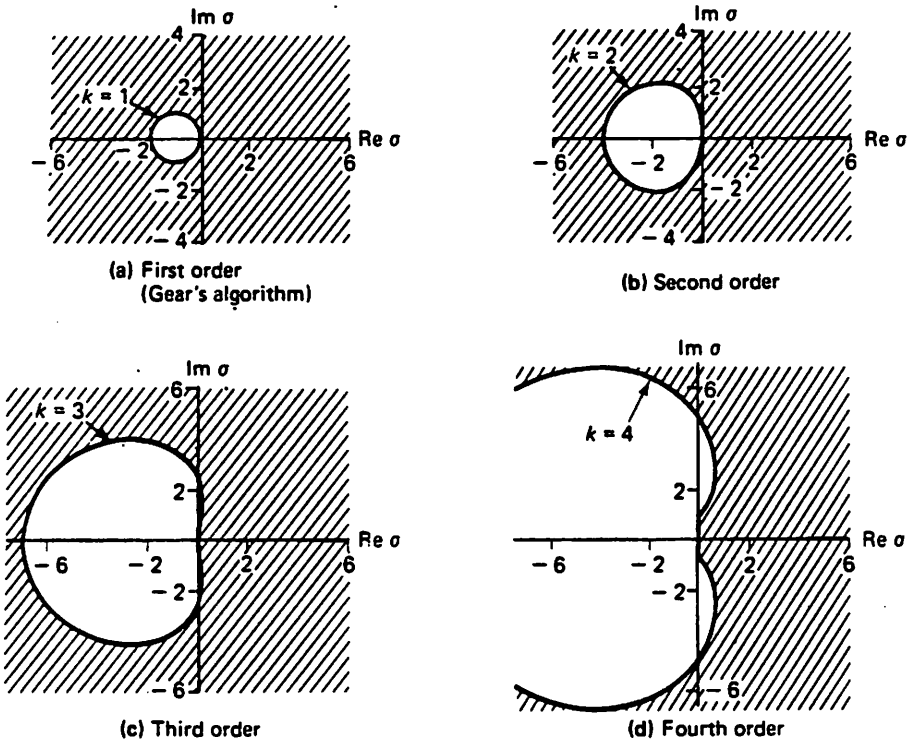


Figure 3.6: Stability regions (shaded areas) for BDFs.

Chapter 4

Theoretical Framework

In this chapter we characterize our approach to the macromodeling problem from a mathematical point of view. We assume that the class of models can be parametrized by a vector α belonging to a set $\mathcal{P}$ of *admissible parameters.* We introduce a *cost function* $c(\alpha, \mathbf{u})$ which measures how different the output of the original system corresponding to the input $\mathbf{u}$ is from the output of the model corresponding to the same input. Then we show that the macromodeling problem can be given the following mathematical formulation

$$\min_{\alpha \in \mathcal{P}} \max_{\mathbf{u} \in \mathcal{U}} c(\alpha, \mathbf{u})$$

where $\mathcal{U}$ is the set of *admissible inputs.* Finally we give expressions for the differential of c with respect to variations in the input $\mathbf{u}$ and in the parameters α.

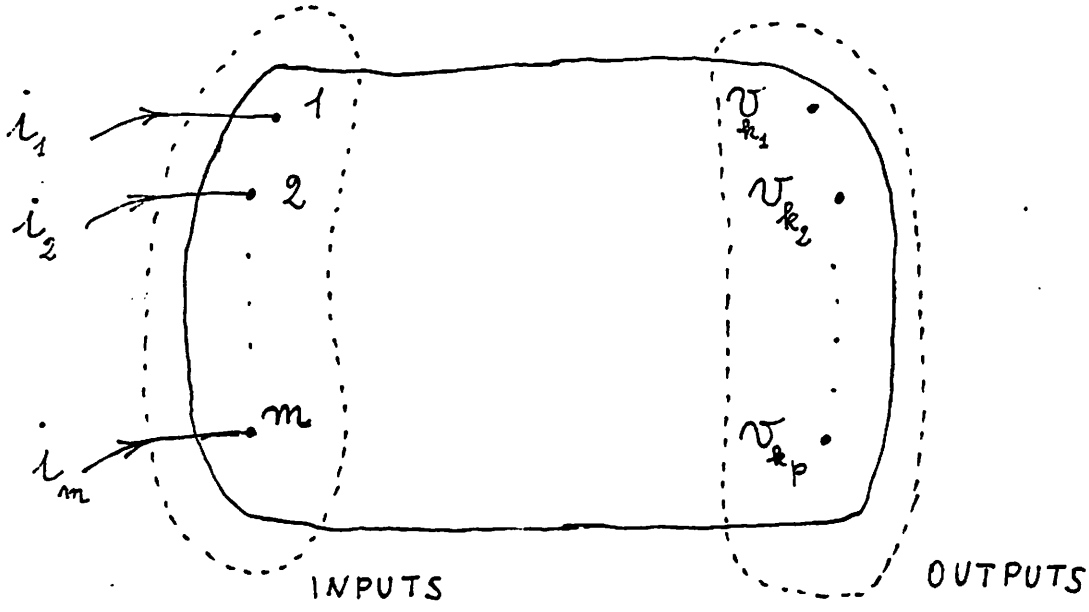


Figure 4.1: Abstract representation of the block to be modeled

4.1 Circuits as Dynamical Systems

Let us assume that the circuit we want to find a macromodel for contains n nodes, m of which are to be considered *inputs* and p *outputs* (fig. 4.1). The input and output nodes may partially or totally overlap, or be completely distinct. Number the nodes so that the inputs have numbers 1 through m; the outputs will then be $k_1, \ldots, k_p$. As input to this system we take the vector $\mathbf{i} = (i_1, \ldots, i_m)^T$ of the currents flowing from the outside into the input nodes, while the output is the the vector $\mathbf{v}^{(o)} = (v_{k_1}, \ldots, v_{k_p})^T$ of the voltages at the output nodes. Under the assumptions stated at the beginning of Chapter 3, the vector $\mathbf{q} = (q_1, \ldots, q_n)^T$ of the node charges can be used to represent the state of the circuit, and there exists a $\mathbf{C}^1$ function $\mathbf{g} : \mathbf{R}^n \to \mathbf{R}^n$ such that $\mathbf{v} = \mathbf{g}(\mathbf{q})$, where $\mathbf{v} = (v_1, \ldots, v_n)^T$ is the vector of the node voltages. We have also seen that the

state equations can be written in the following way:

$$\begin{aligned} \dot{\mathbf{q}}_1 &= -\mathbf{f}_1[\mathbf{v}_1(\mathbf{q}_1)] + \mathbf{B}_1\mathbf{i} \\ \mathbf{v}_1^{(o)} &= \mathbf{g}_1(\mathbf{q}_1) \end{aligned} \tag{4.1}$$

where $\mathbf{f}_1$ is a C^1 function, and $\mathbf{B}_1$ is a rectangular matrix given by

$$\mathbf{B}_1 = \begin{pmatrix} \mathbf{I}_{m\times m} \\ \mathbf{0}_{(n-m)\times m} \end{pmatrix}$$

The reason for using the subscript 1 in the above equations will become apparent in a moment.

At this point we have to keep in mind that the macromodel for the system described by eqn. (4.1) will be used in the simulation of a bigger circuit of which it will be a part. Most circuit simulators are based on Modified Node Analysis, in which the equations describing the circuit are derived by applying KCL to each node. If we require the macromodel to have a structure that lends itself easily to that type of analysis, we are led to the conclusion that it would be convenient for the macromodel to be described by a set of differential equations having the same structure as eqn. (4.1). The difference of course will be in the *number* of equations needed to represent the model.

Having determined what the structure of the model will be, we still need to define a class of *admissible models* in a practical way. Because the model itself is an electrical circuit, it is natural to choose the values of some of its elements (e.g. resistances or capacitances) as parameters. In this way each model will be represented by a vector of parameters $\alpha = (\alpha_1, \ldots, \alpha_r)$, and the class of admissible models will be determined by the definition of a set $\mathcal{P} \in \mathrm{R}^{\nabla}$ of admissible parameter values.

Under these assumptions we conclude that the model should be represented by a set of equations of the form

$$\begin{aligned} \dot{\mathbf{q}}_2 &= -\mathbf{f}_2[\mathbf{v}_2(\mathbf{q}_2), \alpha] + \mathbf{B}_2\mathbf{i} \\ \mathbf{v}_2 &= \mathbf{g}_2(\mathbf{q}_2, \alpha) \end{aligned} \tag{4.2}$$

We will differentiate between the original system and the macromodel by appending the subscript 1 to variables and equations pertaining to the former and the subscript 2 to those pertaining to the latter.

It is readily seen that both eqn. (4.1) and eqn. (4.2) represent dynamical systems where $\mathbf{i}$ is the input vector, $\mathbf{q}$ is the state vector and $\mathbf{v}$ is the output vector.

4.2 Macromodeling as a Min-Max Problem

In the previous section we have seen that both the original circuit and its model can be regarded as dynamical systems of a particular kind. In this and in the following sections we prove theorems that hold for all systems having the same structure. For this reason we temporarily switch to the notation commonly used in system theory, where $\mathbf{u}$ denotes the input, $\mathbf{x}$ the state and $\mathbf{y}$ the output.

In this notation eqns. (4.1) and (4.2) become

$$\begin{aligned} \dot{\mathbf{x}}_1 &= \mathbf{f}_1(\mathbf{x}_1) + \mathbf{B}_1\mathbf{u} \\ \mathbf{y}_1 &= \mathbf{g}_1(\mathbf{x}_1) \end{aligned} \tag{4.3}$$

$$\begin{aligned} \dot{\mathbf{x}}_2 &= \mathbf{f}_2(\mathbf{x}_2, \alpha) + \mathbf{B}_2\mathbf{u} \\ \mathbf{y}_2 &= \mathbf{g}_2(\mathbf{x}_2, \alpha) \end{aligned} \tag{4.4}$$

We will make the following assumptions :

- $\mathbf{g}_1$ and $\mathbf{g}_2$ are continuous functions of their arguments.
- $\mathbf{f}_1$ is Lipschitz continuous.
- $\mathbf{f}_2$ is Lipschitz continuous in $\mathbf{x}$ uniformly in α, i.e. there exists a constant $L > 0$ which does not depend on α such that

$$\|\mathbf{f}_2(\mathbf{x}, \alpha) - \mathbf{f}_2(\mathbf{y}, \alpha)\| \leq \|\mathbf{x} - \mathbf{y}\| \qquad \forall \alpha \in \mathcal{P}.$$

Let S_1, S_2 be the dynamical systems represented by eqns. (4.3) and (4.4) respectively. We want to define a notions of "closeness" between them which is related to the difference in their input-output behaviors, and does not depend on their internal dynamics. For this purpose we fix a time interval $[0, T]$, as well as a subset $\mathcal{U}$ of $L^\infty([0, T], \mathrm{R}^m)$, which will be referred to as the set of *admissible inputs.* It will be convenient to regard $\mathcal{U}$ as a topological space with the topology induced by the weak-* topology of $L^\infty([0, T], \mathrm{R}^m)$. We can now apply Theorems 2.8 and 2.9 to eqns. (4.3) and (4.4) and state the following

Proposition 4.1 *Fix* $\mathbf{x}_{1,0}, \mathbf{x}_{2,0}$. *For every* $\mathbf{u} \in \mathcal{U}$ *and* $\alpha \in \mathcal{P}$ *there exist unique functions* $\mathbf{x}_1(t), \mathbf{x}_2(t)$ *satisfying eqns. (4.3,4.4) and the initial conditions* $\mathbf{x}_1(0) = \mathbf{x}_{1,0}, \mathbf{x}_2(0) = \mathbf{x}_{2,0}$.

For a given $\mathbf{u} \in \mathcal{U}$ let $\mathbf{y}_1(t), \mathbf{y}_2(t)$ be the corresponding outputs of S_1 and S_2 respectively, and define

$$c(\alpha, \mathbf{u}) = \frac{1}{2} \int_0^T \|\mathbf{y}_2(t) - \mathbf{y}_1(t)\|^2 dt$$

c is a function of α and $\mathbf{u}$ because $\mathbf{y}_1(t)$ and $\mathbf{y}_2(t)$ depend implicitly on α and $\mathbf{u}$ through

eqns. (4.3) and (4.4). Then the distance $d(S_1, S_2)$ between S_1 and S_2 is defined as

$$d(S_1, S_2) = \sup_{\mathbf{u} \in \mathcal{U}} c(\alpha, \mathbf{u}) \tag{4.5}$$

In the next section we will state conditions on $\mathcal{U}$ which ensure that the above supremum is finite. This is the case, for instance, if $\mathcal{U}$ is the set

$$\mathcal{U} = \{\mathbf{u}(t) \in L^\infty[0,T] : |u_i(t)| \leq M_i, t \in [0,T], i = 1, \ldots, m\}$$

In this case the supremum is in fact a maximum, i.e. there exists $\mathbf{u}_{max} \in \mathcal{U}$ such that

$$d(S_1, S_2) = c(\alpha, \mathbf{u}_{max})$$

It is immediately recognized that the computation of $\mathbf{u}_{max}$ is equivalent to solving the following optimal control problem :

$$\begin{array}{l} \max_{\mathbf{u} \in \mathcal{U}} \frac{1}{2} \int_0^T \|\mathbf{y}_2(t) - \mathbf{y}_1(t)\|^2 dt \\ \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{g}(\mathbf{x}_1) \\ \mathbf{g}(\mathbf{x}_2, \alpha) \end{pmatrix} \\ \begin{pmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{f}_1(\mathbf{x}_1) + \mathbf{B}_1\mathbf{u} \\ \mathbf{f}_2(\mathbf{x}_2, \alpha) + \mathbf{B}_2\mathbf{u} \end{pmatrix} \end{array} \tag{4.6}$$

The corresponding Hamiltonian is

$$H(\mathbf{x}, \boldsymbol{\lambda}, \mathbf{u}, \alpha) = \frac{1}{2}\|\mathbf{g}_2(\mathbf{x}_2, \alpha) - \mathbf{g}_1(\mathbf{x}_1)\|^2 + \boldsymbol{\lambda}_1^T[\mathbf{f}_1(\mathbf{x}_1) + \mathbf{B}_1\mathbf{u}] + \boldsymbol{\lambda}_2^T[\mathbf{f}_2(\mathbf{x}_2, \alpha) + \mathbf{B}_2\mathbf{u}]$$

Making use of Pontryagin's maximum principle we can give the following characterization of $\mathbf{u}_{max}$:

Let $\mathbf{u}_{max}$ *be* $(u_1(t), \ldots, u_m(t))^T$. *For every* $i = 1, \ldots, m$ *and for almost all* $t \in [0,T]$ *we have either* $u_i(t) = M_i$ *or* $u_i(t) = -M_i$, *provided that* $\mathbf{u}_{max}$ *is not a singular control (see Corollary 2.1).*

This result is somewhat surprising, because it goes against the natural expectation that the worst case input could be almost any function, depending on the specific structure of S_1 and S_2. Instead, *because the systems that we are considering are linear in the input*, we see that we can restrict our attention to a much smaller class of functions, those that are piecewise constant in the interval $[0, T]$; only the number and the location of the switching points remain unknown. From a computational point of view this is a big advantage, which we can exploit in the numerical implementation of a macromodeling algorithm.

Now that we have a measure of the difference in the dynamical behavior of the original system and of the model, it is natural to look for the model that minimizes this difference. In our notation this requires solving the following problem

$$\min_{\alpha \in \mathcal{P}} \left(\max_{\mathbf{u} \in \mathcal{U}} c(\alpha, \mathbf{u}) \right)$$

4.3 The Existence of an Optimal Model

We have seen that our macromodeling problem can be given the following mathematical formulation

$$\min_{\alpha \in \mathcal{P}} \max_{\mathbf{u} \in \mathcal{U}} c(\alpha, \mathbf{u})$$

In this section we prove that the problem so stated does have a solution provided that $\mathcal{P}$ and $\mathcal{U}$ satisfy the following hypotheses :

- The set $\mathcal{P}$ is compact in the usual topology of R^r.
- $\mathcal{U}$ is compact in the weak-* topology of $L^\infty([0, T], \mathrm{R}^m)$.

- There exists a constant $M_0 > 0$ such that

$$\|\mathbf{u}\| \leq M_0 \qquad \forall \mathbf{u} \in \mathcal{U}$$

The first step is to prove that c is a continuous function of its arguments. This is not hard to do, once we have established the fact that $\mathbf{x}_1(t)$ and $\mathbf{x}_2(t)$ in eqns. (4.3) and (4.4) depend continuously on $(\boldsymbol{\alpha}, \mathbf{u})$.

Because the considerations that follow apply to $\mathbf{x}_1$ and $\mathbf{x}_2$ indifferently we will drop the subscripts and refer simply to the following integral equation :

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_0^t \mathbf{f}[\mathbf{x}(\tau), \boldsymbol{\alpha}]\, d\tau + \mathbf{B} \int_0^t \mathbf{u}(\tau)\, d\tau \tag{4.7}$$

(see section 2.4). First we prove a few lemmas.

Lemma 4.1 *Let* $\{\mathbf{u}_k\} \subset L^\infty([0,T], \mathrm{R}^m)$ *be a sequence such that* $\|\mathbf{u}_k\| \leq M_0 \ \ \forall k$ *and* $\lim_{k\to\infty} \mathbf{u}_k = \mathbf{u}$ *in the weak-* topology of* $L^\infty([0,T], \mathrm{R}^m)$. *Define*

$$\begin{aligned} \mathbf{v}_k(t) &= \int_0^t \mathbf{u}_k(\tau)\, d\tau \\ \mathbf{v}(t) &= \int_0^t \mathbf{u}(\tau)\, d\tau \end{aligned}$$

Then $\lim_{k\to\infty} \mathbf{v}_k = \mathbf{v}$ *uniformly in* $[0,T]$.

Proof. Let $u_{k,i}$, u_i be the *i-th* component of $\mathbf{u}_k$ and $\mathbf{u}$ respectively. By definition of weak-* limit, we have

$$\lim_{k\to\infty} \int_0^T g(\tau) u_{k,i}(\tau)\, d\tau = \int_0^T g(\tau) u_i(\tau)\, d\tau$$

for any function $g \in L^1([0,T], \mathrm{R})$. In particular we can take g to be the characteristic function $\chi_{[0,t]}$ of the interval $[0,t]$ ($\chi_{[0,t]}(\tau) = 1$ if

$\tau \in [0,t]$, $\chi_{[0,t]}(\tau) = 0$ otherwise). Then the above equation becomes

$$\lim_{k\to\infty} \int_0^t u_{k,i}(\tau)\, d\tau = \int_0^t u_i(\tau)\, d\tau$$

which shows that $v_{k,i}$ converges to v_i *pointwise* in $[0,T]$. But the sequence $\{v_{k,i}\}$ is equicontinous, because

$$|\dot{v}_{k,i}(t)| = |u_{k,i}(t)| \leq \|\mathbf{u}_k\| \leq M_0$$

(Theorem 2.7). Therefore $\{v_{k,i}\}$ converges to v_i *uniformly* in $[0,T]$ (Theorem 2.5). Because this is true for all the components of $\mathbf{v}_k$, it follows that $\mathbf{v}_k$ converges to $\mathbf{v}$ uniformly. □

Lemma 4.2 *Let $\alpha \in \mathcal{P}$, $\mathbf{u} \in \mathcal{U}$, and let $\mathbf{x}(t)$ be a solution of the integral equation (4.7). Then there exists a constant M_1 which is independent of α and $\mathbf{u}$ such that* $\|\mathbf{x}(t) - \mathbf{x}_0\| \leq M_1 \ \ \forall t \in [0,T]$.

Proof. We know that $\mathbf{f}$ satisfies the Lipschitz inequality

$$\|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{x}_0)\| \leq L\|\mathbf{x} - \mathbf{x}_0\| \ \ \forall \alpha \in \mathcal{P}$$

As a consequence we have

$$\|\mathbf{f}(\mathbf{x})\| \leq \|\mathbf{f}(\mathbf{x}_0)\| + \|\mathbf{f}(\mathbf{x}) - \mathbf{f}(\mathbf{x}_0)\| \leq \|\mathbf{f}(\mathbf{x}_0)\| + L\|\mathbf{x} - \mathbf{x}_0\|$$

Using this inequality in the differential equation we obtain

$$\begin{aligned} \|\mathbf{x}(t) - \mathbf{x}_0\| &\leq \int_0^t \|\mathbf{f}(\mathbf{x}_0\|\, d\tau + \int_0^t L\|\mathbf{x}(\tau) - \mathbf{x}_0\|\, d\tau + \\ &+ \ \|\mathbf{B}\| \int_0^t \|\mathbf{u}\|\, d\tau \leq \\ &\leq \ T(\|\mathbf{f}(\mathbf{x}_0) + \|\mathbf{B}\|M_0) + \int_0^t L\|\mathbf{x}(\tau) - \mathbf{x}_0\|\, d\tau \end{aligned}$$

We can now use Gronwall's inequality (Lemma 2.1) to obtain the following bound for $\|\mathbf{x}(t) - \mathbf{x}_0\|$

$$\|\mathbf{x}(t) - \mathbf{x}_0\| \leq T(\|\mathbf{f}(\mathbf{x}_0)\| + \|\mathbf{B}\|M_0)e^{Lt}$$

which proves the theorem if we take

$$M_1 = T(\|\mathbf{f}(\mathbf{x}_0)\| + \|\mathbf{B}\|M_0)e^{LT}$$

□

Lemma 4.3 *Let $\alpha \in \mathcal{P}$, $\mathbf{u} \in \mathcal{U}$, and let $\mathbf{x}(t)$ be a solution of the integral equation (4.7). Let $\{\alpha_k\} \in \mathcal{P}$ be a sequence converging to α, and let $\{\mathbf{u}_k\} \in \mathcal{U}$ be a sequence converging to $\mathbf{u}$ in the weak-* topology. Let $\mathbf{x}_k(t)$ be the solution of the equation*

$$\mathbf{x}_k(t) = \mathbf{x}_0 + \int_0^t \mathbf{f}[\mathbf{x}_k(\tau), \alpha_k]\, d\tau + \mathbf{B}\int_0^t \mathbf{u}(\tau)\, d\tau$$

Then

$$\lim_{k\to\infty} \int_0^t (\mathbf{f}[\mathbf{x}_k(\tau), \alpha_k] - \mathbf{f}[\mathbf{x}_k(\tau), \alpha])\, d\tau = 0$$

uniformly in $[0, T]$.

Proof. Let $\mathcal{K}$ be the set

$$\mathcal{K} = \{(\mathbf{x}, \alpha) : \|\mathbf{x} - \mathbf{x}_0\| \leq M_1, \alpha \in \mathcal{P}\}$$

where M_1 is the constant defined in the previous lemma. $\mathcal{K}$ is compact, because it is the cartesian product of two compact sets (Theorem 2.1). Since $\mathbf{f}$ is uniformly continuous on $\mathcal{K}$ (Theorem 2.3), for every $\varepsilon > 0$ there exists $\delta_\varepsilon > 0$ such that $\|\mathbf{f}(\mathbf{x}_2, \alpha_2) - \mathbf{f}(\mathbf{x}_1, \alpha_1)\| < \varepsilon$ whenever

$\|\mathbf{x}_2 - \mathbf{x}_1\| < \delta_\varepsilon$, $\|\alpha_2 - \alpha_1\| < \delta_\varepsilon$ and $(\mathbf{x}_1, \alpha_1), (\mathbf{x}_2, \alpha_2) \in \mathcal{K}$. The previous lemma proves that we have $\|\mathbf{x}_k(t) - \mathbf{x}_0\| \leq M_1$ for every k and for every $t \in [0, T]$. Therefore if $\|\alpha_k - \alpha\| < \delta_\varepsilon$ we have

$$\|\mathbf{f}[\mathbf{x}_k(t), \alpha_k] - \mathbf{f}[\mathbf{x}_k(t), \alpha]\| < \varepsilon \quad \forall t \in [0, T]$$

The lemma is then an easy consequence of this inequality. □

We can now prove that the solution of eqn. (4.7) depends continuously on $(\alpha, \mathbf{u})$.

Theorem 4.1 *Let* $\{\mathbf{x}_k\}, \{\alpha_k\}, \{\mathbf{u}_k\}, \mathbf{x}, \alpha, \mathbf{u}$ *satisfy the same assumptions as in the previous lemma. Then* $\lim_{k \to \infty} \mathbf{x}_k(t) = \mathbf{x}(t)$ *uniformly in* $[0, T]$.

Proof. We have

$$\begin{aligned}
\mathbf{x}_k(t) - \mathbf{x}(t) &= \int_0^t (\mathbf{f}[\mathbf{x}_k(\tau), \alpha_k] - \mathbf{f}[\mathbf{x}(\tau), \alpha])\, d\tau + \\
&+ \mathbf{B} \int_0^t [\mathbf{u}_k(\tau) - \mathbf{u}(\tau)]\, d\tau = \\
&= \int_0^t (\mathbf{f}[\mathbf{x}_k(\tau), \alpha] - \mathbf{f}[\mathbf{x}(\tau), \alpha])\, d\tau + \\
&+ \int_0^t (\mathbf{f}[\mathbf{x}_k(\tau), \alpha_k] - \mathbf{f}[\mathbf{x}_k(\tau), \alpha])\, d\tau + \\
&+ \mathbf{B} \int_0^t [\mathbf{u}_k(\tau) - \mathbf{u}(\tau)]\, d\tau
\end{aligned}$$

Exploiting the fact that $\mathbf{f}$ is Lipschitz continuous we obtain the following inequality

$$\|\mathbf{x}_k(t) - \mathbf{x}(t)\| \leq \int_0^t L \|\mathbf{x}_k(\tau) - \mathbf{x}(\tau)\|\, d\tau + a_k(t)$$

where

$$a_k(t) = \| \int_0^t (\mathbf{f}[\mathbf{x}_k(\tau), \alpha_k] - \mathbf{f}[\mathbf{x}_k(\tau), \alpha])\, d\tau + \mathbf{B} \int_0^t [\mathbf{u}_k(\tau) - \mathbf{u}(\tau)]\, d\tau \|$$

Lemmas 4.1 and 4.3 imply that $\lim_{k\to\infty} a_k(t) = 0$ uniformly in $[0,T]$. We now use the generalized Gronwall inequality (Lemma 2.2) to obtain

$$\|\mathbf{x}_k(t) - \mathbf{x}(t)\| \leq a_k(t) + L\int_0^t a_k(\tau)e^{L(t-\tau)}\,d\tau$$

It follows that $\|\mathbf{x}_k(t) - \mathbf{x}(t)\|$ tends to zero uniformly on $[0,T]$. □

Corollary 4.1 *$c(\alpha, \mathbf{u})$ is a continuous function of its arguments.*

Proof. The definition of c :

$$c(\alpha, \mathbf{u}) = \frac{1}{2}\int_0^T \|\mathbf{y}_2(t) - \mathbf{y}_1(t)\|^2\,dt$$

shows that c is a continuous function of $\mathbf{y}_1(t)$ and $\mathbf{y}_2(t)$ as defined in eqns. (4.3) and (4.4). The same equations also show that $\mathbf{y}_1(t)$ and $\mathbf{y}_2(t)$ depend continuously on $\mathbf{x}_1(t)$ and $\mathbf{x}_2(t)$, and we know from the previous theorem that $\mathbf{x}_1(t)$ and $\mathbf{x}_2(t)$ depend continuously on α and $\mathbf{u}$. □

Now that we have established the continuity of c we can prove the following lemma :

Lemma 4.4 *Define*

$$\psi(\alpha) = \max_{\mathbf{u}\in\mathcal{U}} c(\alpha, \mathbf{u})$$

Then ψ is a continuous function of α.

Proof. First of all we note that ψ is well defined because for every fixed α $c(\alpha, \mathbf{u})$ is a continuous function of $\mathbf{u}$ on the compact set $\mathcal{U}$ and

has therefore a minimum there (Theorem 2.2). Let α_0 be any point in $\mathcal{P}$; in order to prove the lemma we have to show that given arbitrarily $\varepsilon > 0$ we can find a $\delta_\varepsilon > 0$ such that

$$|\psi(\alpha) - \psi(\alpha_0)| < \varepsilon$$

whenever $\|\alpha - \alpha_0\| < \delta_\varepsilon$. Because c is continuous, for every $\varepsilon > 0$ and every point $\mathbf{u} \in \mathcal{P}$ there exist a number $\delta(\mathbf{u})$ and a neighborhood $\mathcal{N}(\mathbf{u})$ of $\mathbf{u}$ such that

$$|c(\alpha, \mathbf{v}) - c(\alpha_0, \mathbf{u})| < \varepsilon$$

whenever $\|\alpha - \alpha_0\| < \delta(\mathbf{u})$ and $\mathbf{v} \in \mathcal{N}(\mathbf{u})$. The collection $\{\mathcal{N}(\mathbf{u})\}$ as $\mathbf{u}$ ranges over $\mathcal{U}$ is an open covering of the compact set $\mathcal{U}$. Therefore there exists a finite subcollection $\mathcal{N}(\mathbf{u}_i) : i = 1, \ldots, q$ which still covers $\mathcal{U}$. Let

$$\delta = \min\{\delta_\varepsilon(\mathbf{u}_i) : i = 1, \ldots, q\}$$

and assume that $\|\alpha - \alpha_0\| < \delta_\varepsilon$. Every $\mathbf{u} \in \mathcal{U}$ belongs to some $\mathcal{N}(\mathbf{u}_i)$; therefore we have :

$$\begin{aligned} |c(\alpha, \mathbf{u}) - c(\alpha_0, \mathbf{u})| \leq \\ \leq \; & |c(\alpha, \mathbf{u}) - c(\alpha_0, \mathbf{u}_i)| + \\ + \; & |c(\alpha_0, \mathbf{u}_i) - c(\alpha_0, \mathbf{u})| \leq \\ < \; & \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon \end{aligned}$$

Because $c(\alpha, \mathbf{u}) = c(\alpha_0, \mathbf{u}) + [c(\alpha, \mathbf{u}) - c(\alpha_0, \mathbf{u})]$, we conclude that

$$c(\alpha_0, \mathbf{u}) - \varepsilon \leq c(\alpha, \mathbf{u}) \leq c(\alpha_0, \mathbf{u}) + \varepsilon$$

and taking the maximum with respect to $\mathbf{u}$ we get

$$\psi(\alpha_0) - \varepsilon \leq \psi(\alpha) \leq \psi(\alpha_0) + \varepsilon$$

which proves the continuity of ψ. □

It is now a simple matter to conclude that ψ must have a minimum on the compact set $\mathcal{P}$ (Theorem 2.2), which proves our claim that the our macrmodeling problem has always a solution under the stated the conditions. We can summarize this result in the following

Proposition 4.2 *Let $\mathcal{P}$ be the set of admissible parameters and let $\mathcal{U}$ be the set of admissible inputs. If both $\mathcal{P}$ and $\mathcal{U}$ are compact there exist $\alpha_0 \in \mathcal{P}$ and $\mathbf{u}_0 \in \mathcal{U}$ such that*

$$c(\alpha_0, \mathbf{u}_0) = \min_{\alpha \in \mathcal{P}} \left(\max_{\mathbf{u} \in \mathcal{U}} c(\alpha, \mathbf{u}) \right) \tag{4.8}$$

4.4 Computation of Derivatives

Having proven that modeling can be regarded as a min-max problem, we are now faced with the task of actually computing the pair $(\alpha_0, \mathbf{u}_0)$ appearing in eqn. (4.8). It must be said immediately that finding the global optimum of a real valued function is almost always impossible, even when such optimum is known to exist. Therefore we should not expect to be able to find the optimal model. The best we can hope for is to compute a local optimum, starting from a given initial model. For this purpose we must be able to compute the gradient of the distance with respect to variations in the equations describing the system. In this section we will give an expression for the gradient that can be used for numerical computations.

The computation of the derivative of c with respect to $\mathbf{u}$ has already been dealt with in section 2.5. In this section we will show how to compute $\frac{\partial c}{\partial \alpha}$. By definition, this requires keeping $\mathbf{u}$ fixed; therefore we can drop explicit dependence on $\mathbf{u}$ everywhere, which has the advantage of simplifying notation considerably.

Eqns. (4.3,4.4) can be gathered together in the following way :

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{g}_1(\mathbf{x}_1) \\ \mathbf{g}_2(\mathbf{x}_2, \alpha) \end{pmatrix}$$
$$\begin{pmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{f}_1(\mathbf{x}_1) + \mathbf{B}_1\mathbf{u}(t) \\ \mathbf{f}_2(\mathbf{x}_2, \alpha) + \mathbf{B}_2\mathbf{u}(t) \end{pmatrix} \tag{4.9}$$

where $(\mathbf{x}_1, \mathbf{x}_2)^T \in \mathrm{R}^n$ and $\alpha \in \mathrm{R}^r$. For theoretical purposes it is convenient to add two more state variables, x_0 and x_{n+1}, and two equations :

$$\begin{aligned} \dot{x}_0 &= 1 \\ \dot{x}_{n+1} &= \tfrac{1}{2}\|\mathbf{g}_2(\mathbf{x}_2, \alpha) - \mathbf{g}_1(\mathbf{x}_1)\|^2 \end{aligned} \tag{4.10}$$

If we impose the initial conditions $x_0(0) = x_{n+1}(0) = 0$, we can combine eqns. (4.9) and (4.10) to obtain an equivalent system of the form :

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \alpha) \tag{4.11}$$

where $\mathbf{x} = (x_0, \mathbf{x}_1, \mathbf{x}_2, x_{n+1})^T \in \mathrm{R}^{n+2}$ and

$$\mathbf{f} = \begin{pmatrix} 1 \\ \mathbf{f}_1(\mathbf{x}_1) + \mathbf{B}_1\mathbf{u}(x_0) \\ \mathbf{f}_2(\mathbf{x}_2, \alpha) + \mathbf{B}_2\mathbf{u}(x_0) \\ \tfrac{1}{2}\|\mathbf{g}_2(\mathbf{x}_2, \alpha) - \mathbf{g}_1(\mathbf{x}_1)\|^2 \end{pmatrix} \tag{4.12}$$

Note that $c(\alpha) = x_{n+1}(T, \alpha)$. If we now define

$$\phi(\mathbf{x}) = x_{n+1}$$

we can write the obvious identity

$$\frac{\partial c}{\partial \alpha} = \frac{\partial \phi[\mathbf{x}(T, \alpha)]}{\partial \alpha}$$

Therefore computing the gradient of c is seen to be a particular case of the following problem: evaluate $\frac{\partial \phi[\mathbf{x}(T,\alpha)]}{\partial \alpha}$ where $\phi : \mathrm{R}^n \rightarrow \mathrm{R}$ subject to eqn. (4.11) with $\mathbf{x}(0)$ given *and fixed.*

There is a striking similarity between this problem and a classical optimal control problem, in which we have to compute the variation in ϕ with respect to changes in the input $\mathbf{u}(t)$. We will show that this similarity is not coincidental, because most results of optimal control theory can be extended to our case. In particular, we will show that the gradient of ϕ can be computed by means of a path integral involving the Hamiltonian function associated to eqn. (4.11).

As in a classical control problem, the Hamiltonian associated to eqn. (4.11) is defined to be

$$H(\mathbf{x}, \lambda, \alpha) = \lambda^T \mathbf{f}(\mathbf{x}, \alpha)$$

λ^T is the vector of the *Lagrange multipliers*, which satisfies the equation

$$\dot{\lambda}^T = -\frac{\partial H}{\partial \mathbf{x}} = -\lambda^T \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}, \alpha) \tag{4.13}$$

with the "initial" condition $\lambda(T) = \frac{\partial \phi}{\partial \mathbf{x}}(\mathbf{x}[T, \alpha])$. Our goal is to give an expression for the difference $c(\alpha_1) - c(\alpha_0)$ that involves the function $\mathbf{f}(\mathbf{x}, \alpha)$; the gradient of c will be obtained as the limit of that expression as $\alpha_1 \rightarrow \alpha_0$. For this purpose we introduce

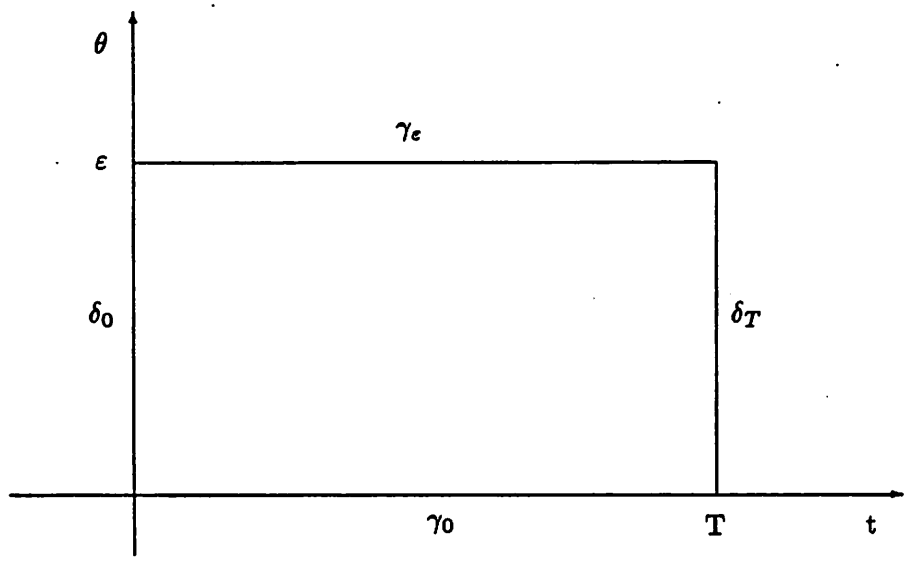


Figure 4.2: The rectangle R_ϵ

a real parameter θ and we let α vary as $\alpha(\theta) = (1-\theta)\alpha_0 + \theta\alpha_1$. For brevity, we will write $\mathrm{f}_\theta(\mathbf{x})$ and $H_\theta(x, \lambda)$ for $\mathrm{f}[\mathbf{x}, \alpha(\theta)]$ and $H[\mathbf{x}, \lambda, \alpha(\theta)]$, respectively. We also introduce the 1-form $\sum_i \lambda_i \, dx_i$, which we will abbreviate as $\lambda^T \, \mathrm{dx}$. Let R_ϵ be the rectangle $[0, T] \times [0, \epsilon]$ in the (t, θ) plane, i.e.

$$R_\epsilon = \{(t, \theta) \in \mathrm{R}^2 : 0 \leq t \leq T, 0 \leq \theta \leq \epsilon\}$$

and let ∂R_ϵ be the boundary of R_ϵ oriented counterclockwise (see Fig. 4.2). Let $\mathbf{x}(t, \theta)$ and $\lambda(t, \theta)$ be the functions defined by eqns. (4.11) and (4.13) with the specified boundary conditions. We want to evaluate the integral $\oint_{R_\epsilon} \lambda^T \, \mathrm{dx}$. ∂R_ϵ consists of four line segments :

- the segment $\gamma_0 = \{0 \leq t \leq T, \theta = 0\}$. On this segment θ is constant, so $\mathbf{x}$ and λ depend only on t. *They are the trajectories described by the solutions of eqns.*

(4.11) and (4.13) when $\alpha = \alpha_0$. The corresponding value of the integral is

$$\int_{\gamma_0} \lambda^T \, \mathrm{dx} = \int_{\gamma_0} \lambda^T \mathrm{f}_0[\mathrm{x}(t, \alpha_0)] \, dt = \int_{\gamma_0} H_0(\mathrm{x}, \lambda) \, dt$$

- the segment $\gamma_\epsilon = \{0 \leq t \leq T, \theta = \epsilon\}$. Reasoning as in the previous case, we conclude that

$$\int_{\gamma_\epsilon} \lambda^T \, \mathrm{dx} = \int_{\gamma_\epsilon} H_\epsilon(\mathrm{x}, \lambda) \, dt$$

- the segment $\delta_0 = \{t = 0, 0 \leq \theta \leq \varepsilon\}$. On this segment x is constant, because $\mathrm{x}(0, \alpha)$ is supposed to be the same for every α. Therefore $\mathrm{dx} = \mathbf{0}$, and the corresponding value of the integral is zero.

- the segment $\delta_T = \{t = T, 0 \leq \theta \leq \varepsilon\}$. In this case we have $\lambda(T, \alpha) = \frac{\partial \phi}{\partial \mathrm{x}}[\mathrm{x}(T, \alpha)]$ so that

$$\int_{\delta_T} \lambda^T \, \mathrm{dx} = \int_{\delta_T} \frac{\partial \phi}{\partial \mathrm{x}} \, \mathrm{dx} = \int_{x_0}^{x_1} d\phi = c(\alpha_\epsilon) - c(\alpha_0)$$

Adding up all the contributions from the various segments, we obtain the following equation

$$\begin{aligned} \oint_{\partial R_\epsilon} \lambda^T \, \mathrm{dx} &= \int_{\gamma_0} \lambda^T \, \mathrm{dx} + \int_{\delta_T} \lambda^T \, \mathrm{dx} - \int_{\gamma_\epsilon} \lambda^T \, \mathrm{dx} - \int_{\delta_0} \lambda^T \, \mathrm{dx} = \\ &= \int_{\gamma_0} H_0 \, dt + c(\alpha_\epsilon) - c(\alpha_0) - \int_{\gamma_\epsilon} H_\epsilon \, dt \end{aligned} \tag{4.14}$$

Now we make use of Green's formula :

$$\oint_{\partial R_\epsilon} f \, dt + g \, d\theta = - \int_{R_\epsilon} \left(\frac{\partial f}{\partial \theta} - \frac{\partial g}{\partial t} \right) dt d\theta$$

In our case the integrand is

$$\begin{aligned} \lambda^T(t, \theta) \, \mathrm{dx}(t, \theta) &= \lambda^T(t, \theta) \left(\frac{\partial \mathrm{x}}{\partial \mathrm{t}} \, dt + \frac{\partial \mathrm{x}}{\partial \theta} \, d\theta \right) = \\ &= \lambda^T(t, \theta) [\mathrm{f}_\theta(\mathrm{x}) \, dt + \frac{\partial \mathrm{x}}{\partial \theta} \, d\theta] = H_\theta(\mathrm{x}, \lambda) \, dt + \lambda^T \frac{\partial \mathrm{x}}{\partial \theta} \, d\theta \end{aligned}$$

so that we can write

$$\oint_{\partial R_\epsilon} \lambda^T \, \mathrm{dx} = -\int_{R_\epsilon} [\frac{\partial H}{\partial \theta} - \frac{\partial}{\partial t}(\lambda^T \frac{\partial \mathrm{x}}{\partial \theta})] \, dt d\theta \tag{4.15}$$

But

$$\begin{aligned} \frac{\partial}{\partial t}(\lambda^T \frac{\partial \mathrm{x}}{\partial \theta}) &= -\lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \mathrm{x}} \frac{\partial \mathrm{x}}{\partial \theta} + \lambda^T \frac{\partial}{\partial \theta} \mathrm{f}_\theta(\mathrm{x}) = \\ &= -\lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \mathrm{x}} \frac{\partial \mathrm{x}}{\partial \theta} + \lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \mathrm{x}} \frac{\partial \mathrm{x}}{\partial \theta} + \lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \theta}(\mathrm{x}) = \\ &= \lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \theta}(\mathrm{x}) \end{aligned}$$

and eqn. (4.15) becomes

$$\oint_{\partial R_\epsilon} \lambda^T \, \mathrm{dx} = \int_{R_\epsilon} \lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \theta} \, dt d\theta - \int_{R_\epsilon} \frac{\partial H}{\partial \theta} \, dt d\theta \tag{4.16}$$

Combining eqns. (4.14) and (4.16) together and rearranging the terms we obtain the following expression for $c(\alpha_\epsilon) - c(\alpha_0)$:

$$c(\alpha_\epsilon) - c(\alpha_0) = \int_{\gamma\epsilon} H_\epsilon \, dt - \int_{\gamma_0} H_0 \, dt - \int_{R_\epsilon} \frac{\partial H_\theta}{\partial \theta} \, dt d\theta + \int_{R\epsilon} \lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \theta} \, dt d\theta \tag{4.17}$$

This expression can be further simplified because

$$\int_{R_\epsilon} \frac{\partial H_\theta}{\partial \theta} \, dt d\theta = \int_0^T (\int_0^\epsilon \frac{\partial H_\theta}{\partial \theta} \, d\theta) \, dt = \int_0^T (H_\epsilon - H_0) \, dt = \int_{\gamma\epsilon} H_\epsilon \, dt - \int_{\gamma_0} H_0 \, dt$$

Therefore the first three terms on the right hand side of eqn. (4.17) cancel out, and we obtain the following final equation :

$$c(\alpha_\epsilon) - c(\alpha_0) = \int_{R_\epsilon} \lambda^T \frac{\partial \mathrm{f}_\theta}{\partial \theta} \, dt d\theta \tag{4.18}$$

Let $\delta\alpha = \alpha_1 - \alpha_0$. The *weak differential of* c *at* α_0 *in the direction* $\delta\alpha$ is

$$\delta c(\alpha_0; \delta\alpha) = \lim_{\epsilon \to 0} \frac{c(\alpha_\epsilon) - c(\alpha_0)}{\epsilon} \tag{4.19}$$

which in our case becomes

$$\delta c(\alpha_0; \delta\alpha) = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \int_0^T \left(\int_0^\epsilon \lambda^T \frac{\partial \mathbf{f}_\theta}{\partial \theta} \, d\theta \right) dt \tag{4.20}$$

Because all the functions involved are of class C^1, we can interchange the operations of limit and integration, and we obtain the following expression for δc :

$$\delta c(\alpha_0; \delta\alpha) = \int_{\gamma_0} \lambda^T \frac{\partial \mathbf{f}}{\partial \alpha}(\mathbf{x}, \alpha_0) \delta\alpha \, dt = \int_{\gamma_0} \frac{\partial H}{\partial \alpha} \delta\alpha \, dt \tag{4.21}$$

It is immediate to recognize the similarity between eqn. (4.21) and the corresponding expression for the variation of ϕ in the classical optimal control problem, which is

$$\delta\phi = \int_{\gamma_0} \lambda^T \frac{\partial \mathbf{f}}{\partial \mathbf{u}}[\mathbf{x}(t), \mathbf{u}(t)] \delta\mathbf{u}(t) \, dt = \int_{\gamma_0} \frac{\partial H}{\partial \mathbf{u}} \delta\mathbf{u} \, dt \tag{4.22}$$

After deriving eqn. (4.21) in a rigorous way, it might be of some interest to give a less correct but more intuitive explanation of why it is true. For this purpose, let $\mathbf{X}(t, \mathbf{x}_0)$ be the general integral of eqn. (4.11), i.e. the value of the solution of eqn. (4.11) with initial condition $\mathbf{x}_0$ at time t. Recall that the Lagrange multipliers give the derivative of c with respect to variations in the initial conditions :

$$\lambda(t) = \frac{\partial c(\mathbf{X}[T - t, \mathbf{x}(t)])}{\partial \mathbf{x}(t)}$$

Now let $\mathbf{x}(t)$ be the trajectory of eqn. (4.11) when $\alpha = \alpha_0$; if we add a small perturbation $\delta\mathbf{f}$ to $\mathbf{f}$ we obtain a nearby trajectory, say $\mathbf{x}(t) + \delta\mathbf{x}(t)$. The corresponding change in the value of c can be obtained by adding up all the contributions of each individual perturbation $\delta\mathbf{x}(t)$, as t varies from 0 to T :

$$\delta c = \int_0^T \frac{\partial c(\mathbf{X}[T - t, \mathbf{x}(t)])}{\partial \mathbf{x}(t)} \delta\mathbf{x}(t) = \int_0^T \lambda^T(t) \delta\mathbf{x}(t) \tag{4.23}$$

But $\delta\mathbf{x}(t) = \delta\dot{\mathbf{x}}\,dt \approx \delta\mathbf{f}\,dt$, and we get

$$\delta c = \int_0^T \boldsymbol{\lambda}^T(t)\delta\mathbf{f}[\mathbf{x}(t)]\,dt = \int_{\gamma_0} \boldsymbol{\lambda}^T \frac{\partial \mathbf{f}}{\partial \boldsymbol{\alpha}}(\mathbf{x}, \boldsymbol{\alpha}_0)\delta\boldsymbol{\alpha}\,dt \tag{4.24}$$

which is eqn. (4.21).

It is also interesting to point out that eqn. (4.21) can be regarded as the extension to the time domain of a well-known expression for the sensitivity of node voltages of an electrical circuit with respect to variations in the value of the components valid for the stationary case. In fact it can be shown that if $\mathbf{x}_0$ satisfies the equation

$$\mathbf{f}(\mathbf{x}_0) = 0$$

and $c(\mathbf{x})$ is a real-valued function, the variation $\delta c = c(\mathbf{x}_0 + \delta\mathbf{x}) - c(\mathbf{x}_0)$ when $\mathbf{f}$ is perturbed by a small amount $\delta\mathbf{f}$ can be computed as

$$\delta c = \boldsymbol{\lambda}^T \delta\mathbf{f}(\mathbf{x}_0)$$

where $\boldsymbol{\lambda}$ satisfies the equation

$$-\boldsymbol{\lambda}^T \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \frac{\partial c}{\partial \mathbf{x}}(\mathbf{x}_0)$$

4.5 Application to Circuit Equations

We now want to specialize the results of the previous section to the case of circuit equations; therefore we return to the notation used at the beginning of this chapter. The Hamiltonian becomes :

$$H(\mathbf{q}, \boldsymbol{\lambda}, \mathbf{i}, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{v}_2^{(o)} - \mathbf{v}_1^{(o)}\|^2 + \boldsymbol{\lambda}_1^T[-\mathbf{f}_1(\mathbf{v}_1) + \mathbf{B}_1\mathbf{i}] + \boldsymbol{\lambda}_2^T[-\mathbf{f}_2(\mathbf{v}_2, \boldsymbol{\alpha}) + \mathbf{B}_2\mathbf{i}]$$

where the dependence on $\mathbf{q}$ is implicit through the voltages $\mathbf{v}_1, \mathbf{v}_2$. The Lagrange multipliers satisfy the differential equations

$$\dot{\lambda}_i^T = -\frac{\partial H}{\partial \mathbf{q}_i} = -\frac{\partial H}{\partial \mathbf{v}_i}\frac{\partial \mathbf{v}_i}{\partial \mathbf{q}_i}, \qquad i = 1,2 \tag{4.25}$$

But $\frac{\partial \mathbf{v}_i}{\partial \mathbf{q}_i}$ is by definition the inverse of the capacitance matrix $\mathbf{C}_i(\mathbf{v})$, and we can rewrite equations (4.25) in the following form

$$\mathbf{C}_1^T\dot{\lambda}_1 - \frac{\partial \mathbf{f}_1}{\partial \mathbf{v}_1}^T \lambda_1 = \mathbf{v}_2^{(o)} - \mathbf{v}_1^{(o)} \tag{4.26}$$

$$\mathbf{C}_2^T\dot{\lambda}_2 - \frac{\partial \mathbf{f}_2}{\partial \mathbf{v}_2}^T \lambda_2 = -\mathbf{v}_2^{(o)} + \mathbf{v}_1^{(o)} \tag{4.27}$$

In order to compute the derivative of the cost function with respect to variations in the circuit elements, we have to distinguish between two cases : variations in memoryless elements, and variations in elements with memory. Let α be a parameter appearing in the branch equations of some memoryless element. H is a function of α through $\mathbf{f}_2$, and eqn. (4.21) becomes

$$\delta c = -\int_0^T \lambda_2^T \frac{\partial \mathbf{f}_2}{\partial \alpha}(\mathbf{v}_2, \alpha)\delta\alpha \, dt \tag{4.28}$$

For example, suppose that the element which depends on α is a resistor or current source (independent or controlled) connected between nodes i and j, and denote its branch current by I_{ij} (see Fig. 4.3). Then the equation relative to node i contains the term $+I_{ij}$, and the equation relative to node j contains the term $-I_{ij}$, and those are

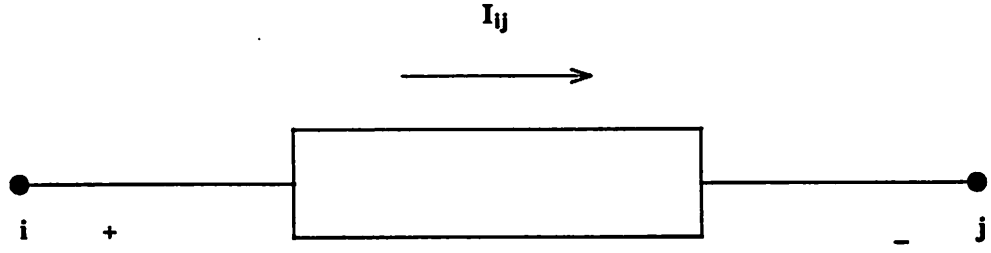


Figure 4.3: Computation of sensitivities : memoryless element

the only equations affected by I_{ij}. Therefore we have

$$\frac{\partial \mathbf{f}_2}{\partial \alpha} = \begin{pmatrix} 0 \\ \cdot \\ \frac{\partial I_{ij}}{\partial \alpha} \\ \cdot \\ -\frac{\partial I_{ij}}{\partial \alpha} \\ \cdot \\ 0 \end{pmatrix}$$

and

$$\lambda_2^T \frac{\partial \mathbf{f}_2}{\partial \alpha} = (\lambda_i - \lambda_j)\frac{\partial I_{ij}}{\partial \alpha} \tag{4.29}$$

For instance, in the case of a linear resistor $I_{ij} = G(v_i - v_j)$. If the resistor conductance is taken as parameter we have $\frac{\partial I_{ij}}{\partial G} = (v_i - v_j)$, so that the derivative of the cost function with respect to variations in G is

$$\delta c = -\int_{\gamma_0} (\lambda_i - \lambda_j)(v_i - v_j)\delta G \, dt \tag{4.30}$$

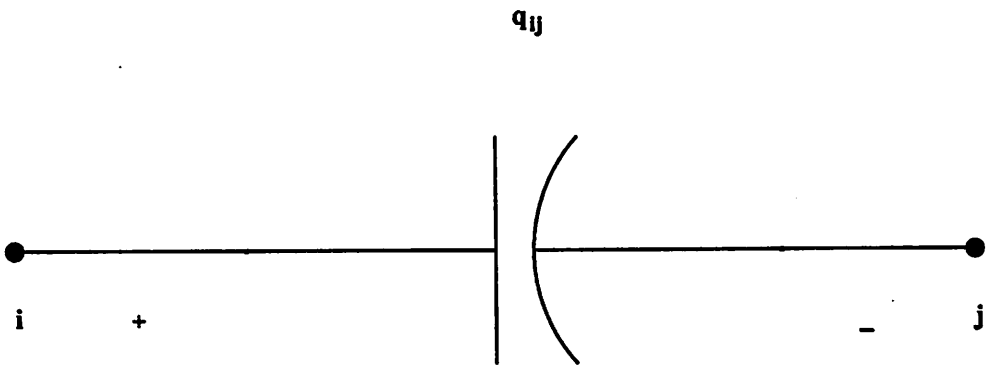


Figure 4.4: Computation of sensitivities : capacitive element

The case of an element with memory is slightly more complex, because α affects the equations relating the charges to the currents, i.e. we have $\mathbf{q}_2 = \mathbf{q}_2(\mathbf{v}_2, \alpha)$ and $\mathbf{v}_2 = \mathbf{v}_2(\mathbf{q}_2, \alpha)$. By definition, the function $\mathbf{q}_2[\mathbf{v}_2(\mathbf{q}_2, \alpha), \alpha]$ does not depend on α, so by taking its total derivative with respect to α we obtain the following equation

$$0 = \frac{\partial \mathbf{q}_2}{\partial \mathbf{v}_2}\frac{\partial \mathbf{v}_2}{\partial \alpha} + \frac{\partial \mathbf{q}_2}{\partial \alpha} \tag{4.31}$$

$\frac{\partial H}{\partial \alpha}$ can now be computed as follows :

$$\frac{\partial H}{\partial \alpha} = \frac{\partial H}{\partial \mathbf{v}_2}\frac{\partial \mathbf{v}_2}{\partial \alpha} = -\dot{\boldsymbol{\lambda}}_2^T \mathbf{C}_2 \frac{\partial \mathbf{v}_2}{\partial \alpha} = -\dot{\boldsymbol{\lambda}}_2^T \frac{\partial \mathbf{q}_2}{\partial \mathbf{v}_2}\frac{\partial \mathbf{v}_2}{\partial \alpha} = \dot{\boldsymbol{\lambda}}_2^T \frac{\partial \mathbf{q}_2}{\partial \alpha} \tag{4.32}$$

To make a concrete example, let α be the capacitance C of a linear capacitor connected between nodes i and j (Fig. 4.4). Then the charge on the capacitor is $q_{ij} = C(v_i - v_j)$, and the charge equation (eqn. 3.1) at node i contains the term $+q_{ij}$, while the charge

equation at node j contains the term $-q_{ij}$. Then

$$\frac{\partial \mathbf{q}_2}{\partial C} = \begin{pmatrix} 0 \\ \cdot \\ (v_i - v_j) \\ \cdot \\ -(v_i - v_j) \\ \cdot \\ 0 \end{pmatrix}$$

and

$$\dot{\boldsymbol{\lambda}}_2^T \frac{\partial \mathbf{q}_2}{\partial C} = (\dot{\lambda}_i - \dot{\lambda}_j)(v_i - v_j)$$

Substituting this expression in eqn. (4.28) gives the following equation for δc

$$\delta c = \int_{\gamma_0} (\dot{\lambda}_i - \dot{\lambda}_j)(v_i - v_j)\delta C \, dt \tag{4.33}$$

Chapter 5

Saddle Point Algorithm

In this chapter I explore the issues involved in solving

$$\min_{\alpha \in \mathcal{P}} \max_{\mathbf{u} \in \mathcal{U}} c(\alpha, \mathbf{u}) \tag{5.1}$$

from a numerical analysis point of view. It is apparent that this problem can be tackled in many different ways; in order to justify our choice of a particular algorithm, I first explore other possible alternatives, and I explain why they have to be discarded. Then I present the chosen algorithm, and I give proofs for some of its numerical properties which are relevant to our case. The actual computer implementation will be dealt with in the next chapter.

5.1 A Conceptual Algorithm

We now return to the problem of finding the gradient of $d(S_1, S_2)$ (eqn. (4.5)). Because d is defined to be the supremum of a family of functions, in general it will not have a gradient in the usual meaning of the term. However it can be shown that it has a so-called *generalized gradient*, which is a set of vectors forming a closed convex cone in R^n. The generalized gradient of a function f is usually denoted by ∂f. The following theorem is relevant to our case :

Theorem 5.1 (Clarke) *Let* $d : \mathrm{R}^n \rightarrow \mathrm{R}$ *be defined as*

$$d(\mathbf{x}) = \sup_{w \in W} f(\mathbf{x}, w)$$

where W *is a compact set and* $f : \mathrm{R}^n \times W \rightarrow \mathrm{R}$. *Suppose that* $f(\mathbf{x}, w)$ *and* $\frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}, w)$ *are continuous functions of* $(\mathbf{x}, w)$. *Define*

$$M(\mathbf{x}) = \{w \in W : f(\mathbf{x}, w) = d(\mathbf{x})\}$$

Then $\partial d(\mathbf{x})$ *is the closed convex cone generated by the vectors* $\frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}, w)$ *as* w *ranges in the set* $M(\mathbf{x})$.

There are several algorithms that can be used to find local minima of a non-differentiable function that has a generalized gradient [32]. Therefore we could try to solve our macromodeling problem using the following

Algorithm

Step 0 : Choose an initial set of parameters α for the model. S_1.

Step 1 : Compute all inputs $\mathbf{u} \in \mathcal{U}$ that maximize the function $c(\boldsymbol{\alpha}, \mathbf{u})$. Call the set of such inputs $M(\boldsymbol{\alpha})$.

Step 2 : Compute ∂d as the closed convex cone generated by the set $\{\partial c/\partial \boldsymbol{\alpha} : \mathbf{u} \in M(\boldsymbol{\alpha})\}$.

Step 3 : Compute a new set of parameters using an algorithm suitable for minimizing a non-differentiable function. Repeat from Step 1 until a local minimum is found.

Although theoretically feasible, this algorithm cannot be implemented in practice, mainly because Step 1 requires solving a global optimization problem. Moreover the actual numerical implementation of an algorithm for non-differentiable optimization presents several problems; for these reasons we decided to take a completely different approach.

It can be noted immediately that any point that solves eqn. (5.1) is a saddle point for the function $c(\boldsymbol{\alpha}, \mathbf{u})$. This observation prompted the search for an algorithm that could find saddle points of a differentiable function. It is true that very little can be said a priori about a solution computed in this way; however finding global extrema of a general function is almost an impossible task, so, from a practical point of view, this is the best that we can do. On the other hand we can now work with functions which are differentiable, and this makes a difference as far as the actual implementation is concerned. In the following sections I show how several algorithms that are normally used to find local minima of a function can be modified to find saddle points.

Another problem that must be tackled now is the fact that $\mathbf{u}$ belongs to $L^{\infty}([0,T], \mathrm{R}^m)$, which is an infinite dimensional vector space. In the next chapter we will see that the

set $\mathcal{U}$ of admissible inputs can be restricted to a point where it can be parametrized by a finite dimensional set of parameters. Thus c becomes an ordinary function defined on $R^{r \times s}$, where r is the dimension of α and s is the dimension of the parameter defining $\mathbf{u}$. To stress this point we will recur to a slight change in notation, and denote the arguments of c by $\mathbf{x}$ and $\mathbf{y}$. Throughout the rest of this chapter it will be assumed that $c(\mathbf{x}, \mathbf{y})$ is twice continuously differentiable; we introduce the joint variable $\mathbf{z} = (\mathbf{x}, \mathbf{y})^T$, and we denote the Hessian matrix of c by

$$\mathbf{F}(\mathbf{z}) = \frac{\partial c^2}{\partial \mathbf{z}^2} = \begin{pmatrix} \frac{\partial c^2}{\partial \mathbf{x}^2} & \frac{\partial c^2}{\partial \mathbf{x} \partial \mathbf{y}} \\ \frac{\partial c^2}{\partial \mathbf{y} \partial \mathbf{x}} & \frac{\partial c^2}{\partial \mathbf{y}^2} \end{pmatrix} = \begin{pmatrix} \mathbf{P}(\mathbf{z}) & \mathbf{R}(\mathbf{z}) \\ \mathbf{R}(\mathbf{z})^T & -\mathbf{Q}(\mathbf{z}) \end{pmatrix} \tag{5.2}$$

We will assume that there exists a constant $m > 0$ such that $<\mathbf{P}(\mathbf{z})\mathbf{w}_1, \mathbf{w}_1> \geq m\|\mathbf{w}_1\|^2$, $<\mathbf{Q}(\mathbf{z})\mathbf{w}_2, \mathbf{w}_2> \geq m\|\mathbf{w}_2\|^2$ for all $\mathbf{z}, \mathbf{w}_1, \mathbf{w}_2$. $\mathbf{g}(\mathbf{z})$ will denote the gradient of c : $\mathbf{g}(\mathbf{z}) = \left(\frac{\partial c}{\partial \mathbf{z}}\right)^T$. Explicit dependence on $\mathbf{z}$ will be omitted whenever it is not essential.

We also introduce the matrix

$$\mathbf{U} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -\mathbf{I} \end{pmatrix}$$

and we note that the following inequality is also satisfied

$$\begin{aligned} <\mathbf{w}, \mathbf{UFw}> &= \frac{1}{2} <\mathbf{w}, (\mathbf{UF} + \mathbf{FU})\mathbf{w}> = \\ = \begin{pmatrix} \mathbf{w}_1^T & \mathbf{w}_2^T \end{pmatrix} &\begin{pmatrix} \mathbf{P} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{pmatrix} \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{pmatrix} \geq m(\|\mathbf{w}_1\|^2 + \|\mathbf{w}_2\|^2) = m\|\mathbf{w}\|^2 \end{aligned}$$

In other words, the symmetric matrix $\frac{1}{2}(\mathbf{UF} + \mathbf{FU})$ is positive definite uniformly in $(\mathbf{x}, \mathbf{y})$. An interesting result which will be needed later is that so is the matrix $\frac{1}{2}(\mathbf{UF}^{-1} + \mathbf{F}^{-1}\mathbf{U})$. This can be readily seen from the identity

$$\mathbf{F}^{-1}(\mathbf{UF} + \mathbf{FU})\mathbf{F}^{-1} = (\mathbf{UF}^{-1} + \mathbf{F}^{-1}\mathbf{U})$$

and from the well-known fact that $\mathbf{A} > \mathbf{0}$ implies $\mathbf{C}^T\mathbf{AC} > \mathbf{0}$ for any nonsingular matrix C.

5.2 A Modification of the Steepest Descent Algorithm

The classical steepest descent algorithm can be easily modified in the following way to find a saddle point :

Modified Steepest Descent Algorithm

Data : $\mathbf{z}_0, \varepsilon, c$.

Step 0 : Set $k = 0$.

Step 1 : Set $\mathbf{g}_k = \mathbf{g}(\mathbf{z}_k), \mathbf{d}_k = -\mathbf{U}\mathbf{g}_k$.

Step 2 : Compute $\mathbf{z}_{k+1} = \mathbf{z}_k + t_k\mathbf{d}_k$ such that the equation $< \mathbf{U}\mathbf{g}_{k+1}, \mathbf{d}_k >= 0$ is satisfied.

Step 3 : If $\|\mathbf{g}_{k+1}\| < \varepsilon$ stop. Else set $k = k + 1$, and go to Step 1.

It is easy to prove the following

Theorem 5.2 *Let* $\{\mathbf{z}_k\}$ *be a sequence generated by the above algorithm, and suppose that* $\lim_{k\to\infty} \mathbf{z}_k = \hat{\mathbf{z}}$. *Define* $\hat{\mathbf{g}} = \mathbf{g}(\hat{\mathbf{z}})$. *Then* $\hat{\mathbf{g}} = \mathbf{0}$.

Proof. Because of the continuity of $\frac{\partial c}{\partial z}$, we have $\lim_{k\to\infty} \mathbf{g}_k = \hat{\mathbf{g}}$. We also have $< \mathbf{U}\mathbf{g}_{k+1}, \mathbf{d}_k >= 0$ for all k. Therefore

$$0 = \lim_{k\to\infty} < \mathbf{U}\mathbf{g}_{k+1}, \mathbf{d}_k >= -\lim_{k\to\infty} < \mathbf{U}\mathbf{g}_{k+1}, \mathbf{U}\mathbf{g}_k >=$$

$$-\lim_{k\to\infty} < \mathbf{g}_{k+1}, \mathbf{g}_k >= -\|\hat{\mathbf{g}}\|^2$$

□

This is probably the easiest algorithm one could think of to find a saddle point. Unfortunately, unlike the original steepest descent algorithm, it does not always converge, not even locally, as can be seen if we take c to be $c(x,y) = x^2 + 20xy - y^2$. It can be easily verified that if $(x_0, y_0) \neq (0,0)$ the algorithm will always generate a divergent sequence. A different algorithm is therefore needed, and one that comes to mind immediately is Newton's method, because of its well-known local convergence properties [18]. Unfortunately computing the Hessian matrix is prohibitively expensive in our case, and this rules out a straightforward implementation of Newton's method. On the other hand the local convergence property should be retained, which suggests that an appropriate algorithm would be one that builds an approximation to the Hessian matrix using information obtained exclusively from the function and its first derivatives. There is a wide class of such algorithms, normally referred to as *quasi-Newton* or *variable metric algorithms.*

5.3 A Variable Metric Algorithm

As mentioned in the previous section, the idea underlying all variable metric algorithms is to build an approximation to the Hessian using information obtained esclusively from the first derivatives of the function. A detailed analysis of those methods was first published by Huang [33], and it can be found in most modern texts on nonlinear optimization (e.g. [34]). Such a general analysis is beyond the scope of this dissertation, and we will confine ourselves to a much more limited presentation of one specific

algorithm, namely the Davidon, Fletcher and Powell (or DFP) method. This will be sufficient to illustrate the basic idea common to all variable metric algorithms. We will substantially follow the presentation of the DFP method given by Luenberger [35]. The algorithm is defined as follows:

Davidon-Fletcher-Powell Method

Data : $\mathbf{H}_0, \mathbf{z}_0, \varepsilon, c.$

Step 0 : Set $k = 0.$

Step 1 : Set $\mathbf{g}_k = \mathbf{g}(\mathbf{z}_k), \mathbf{d}_k = -\mathbf{H}_k\mathbf{g}_k.$

Step 2 : Compute $\mathbf{z}_{k+1} = \mathbf{z}_k + t_k\mathbf{d}_k$ such that the equation $< \mathbf{g}_{k+1}, \mathbf{d}_k >= 0$ is satisfied.

Step 3 : If $\|\mathbf{g}_{k+1}\| < \varepsilon$ stop. Else set

$$\begin{aligned} \mathbf{p}_k &= \mathbf{z}_{k+1} - \mathbf{z}_k \\ \mathbf{q}_k &= \mathbf{g}_{k+1} - \mathbf{g}_k \\ \mathbf{H}_{k+1} &= \mathbf{H}_k + \frac{\mathbf{p}_k\mathbf{p}_k^T}{< \mathbf{p}_k, \mathbf{q}_k >} - \frac{\mathbf{H}_k\mathbf{q}_k\mathbf{q}_k^T\mathbf{H}_k^T}{< \mathbf{H}_k\mathbf{q}_k, \mathbf{q}_k >} \\ k &= k+1 \end{aligned}$$

and go to Step 1.

When applied to the minimization of a convex function this algorithm has many attractive properties. We will mention those that are potentially interesting for the problem at hand, without giving proofs.

Theorem 5.3 *Suppose that* $c = \frac{1}{2} < \mathbf{Fz} - \mathbf{b}, \mathbf{z} >$, *with* $\mathbf{F} > 0$ *and constant. Then* $\mathbf{z}_{k+1} = \mathbf{F}^{-1}\mathbf{b}$ *and* $\mathbf{H}_{k+1} = \mathbf{F}^{-1}$ *for some* $k \leq n$.

Theorem 5.4 *Let* $\mathbf{F}(\mathbf{z})$ *be Lipschitz continuous. Suppose that* $\mathbf{H}_0 > 0$ *and that* $\mathbf{F}(\mathbf{z}) > 0$ *for all* $\mathbf{z}$. *Then* $\mathbf{H}_k > 0$ *for all* k.

Theorem 5.5 *Let* $\mathbf{F}(\mathbf{z})$ *be Lipschitz continuous. Suppose that there exists* $\hat{\mathbf{z}} \in \mathrm{R}^n$ *such that* $\mathbf{g}(\hat{\mathbf{z}}) = 0$ *and that* $\mathbf{F}(\mathbf{z}) > 0$ *for all* $\mathbf{z}$. *Let* $\mathbf{H}_0$ *be any symmetric positive definite matrix. Then the sequence* $\{\mathbf{z}_k\}$ *converges superlinearly to* $\hat{\mathbf{z}}$ *for every choice of the initial point* $\mathbf{z}_0$.

Theorem 5.3 says that the DFP method can find the minimum of a positive definite quadratic function and the inverse of its Hessian matrix in a finite number of steps. Theorem 5.5 asserts that the same algorithm will generate a sequence that converges superlinearly to the minimum of a general convex function. This makes the DFP method a very attractive candidate for our purposes, and indeed it has been used to find saddle points of some special functions [36]. Unfortunately the method cannot be proved to converge to a zero of the gradient if the Hessian is not positive definite, and numerical experiments confirm this. A brief explanation of why this is the case follows. Suppose that $\lim_{k\to\infty} \mathbf{z}_k = \hat{\mathbf{z}}$; under the not unreasonable assumption that $\lim_{k\to\infty} \mathbf{H}_k = \mathbf{F}(\hat{\mathbf{z}})^{-1} = \hat{\mathbf{H}}$, we have

$$0 = \lim_{k\to\infty} < \mathbf{g}_{k+1}, \mathbf{d}_k > = - \lim_{k\to\infty} < \mathbf{g}_{k+1}, \mathbf{H}_k \mathbf{g}_k > = - \lim_{k\to\infty} < \hat{\mathbf{g}}, \hat{\mathbf{H}}\hat{\mathbf{g}} >$$

Because $\hat{\mathbf{H}}$ is not positive definite, $< \hat{\mathbf{g}}, \hat{\mathbf{H}}\hat{\mathbf{g}} >$ does not imply that $\hat{\mathbf{g}} = \mathbf{0}$.

These results make it clear that the DFP algorithm as is cannot be used to find stationary points of a function where the Hessian is indefinite. One possibility is to

modify the line search in Step 2 so that $\mathbf{g}_{k+1}$ satisfies the equation $<\mathbf{U}\mathbf{g}_{k+1}, \mathbf{d}_k>= 0$. This implies that $0 = \lim_{k\to\infty} <\mathbf{U}\mathbf{g}_{k+1}, \mathbf{H}_k\mathbf{g}_k>=<\mathbf{U}\hat{\mathbf{g}}, \hat{\mathbf{H}}\hat{\mathbf{g}}>= \frac{1}{2} <\hat{\mathbf{g}}, (\mathbf{U}\hat{\mathbf{H}}+\hat{\mathbf{H}}\mathbf{U})\hat{\mathbf{g}}>$ and now we can conclude that $\hat{\mathbf{g}} = \mathbf{0}$ because $\frac{1}{2}(\mathbf{U}\hat{\mathbf{H}} + \hat{\mathbf{H}}\mathbf{U}) > \mathbf{0}$. Unfortunately most properties of the matrices $\mathbf{H}_k$ rely on the equation $< \mathbf{g}_{k+1}, \mathbf{d}_k >= 0$, so any changes in the line search will most likely destroy the usefulness of the algorithm.

Let us summarize what we have learned from this seemingly endless string of negative results :

- A straightforward modification on the steepest descent method does not work, therefore we need to compute the Hessian matrix in some way.
- Computing the Hessian matrix directly is too expensive.
- Variable metric algorithms compute an approximation of the Hessian matrix from the first derivatives, but the equation used in the line search does not guarantee convergence if the function is not convex.
- The equation used in the line search can be changed to ensure that any limit point will be stationary, but this destroys the other properties of the variable metric algorithms.
- Therefore we need an algorithm that can compute an approximation to the Hessian without relying on any particular properties of the line searches.

A few algorithms of this type exist, and among them we have chosen one that has proven to have good numerical performance in our case.

5.4 Broyden's Method

This algorithm was first proposed by Broyden [37] as an iterative method to solve nonlinear systems of equations. Like variable metric algorithms it approximates the inverse of the Hessian matrix by successive updates generated from the vectors $\mathbf{p}_k = \mathbf{z}_{k+1} - \mathbf{z}_k$ and $\mathbf{q}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$. Its starting point, which it shares with other methods, is the following approximate equality, which can be obtained from the Taylor expansion of c

$$\mathbf{g}(\mathbf{z}_{k+1}) - \mathbf{g}(\mathbf{z}_k) \cong \mathbf{F}(\mathbf{z}_{k+1})(\mathbf{z}_{k+1} - \mathbf{z}_k)$$

Suppose that we are trying to build a sequence $\{\mathbf{B}_k\}$ that approximates $\mathbf{F}(\mathbf{z}_k)$. The above equation suggests that the matrices $\mathbf{B}_k$ should be required to satisfy the following equation, commonly known as the *secant relation* :

$$\mathbf{q}_k = \mathbf{B}_{k+1}\mathbf{p}_k \tag{5.3}$$

It has been noted [38] that eqn. (5.3) is equivalent to the following one

$$\int_{\mathbf{z}_k}^{\mathbf{z}_{k+1}} (\mathbf{B}_{k+1}\mathbf{F}^{-1}(\mathbf{z}) - \mathbf{I})\, d\mathbf{z} = \mathbf{0} \tag{5.4}$$

In this form, the secant relation can be interpreted as an attempt to find the best approximation to $\mathbf{F}$ on the segment $[\mathbf{z}_k, \mathbf{z}_{k+1}]$. Broyden's method is characterized by two additional requirements : the difference between $\mathbf{B}_{k+1}$ and $\mathbf{B}_k$ should be a rank-one matrix, and we should have $\mathbf{B}_{k+1}\mathbf{r} = \mathbf{B}_k\mathbf{r}$ for every vector $\mathbf{r}$ orthogonal to $\mathbf{p}_k$. This translates into the following equations

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \mathbf{u}\mathbf{p}_k^T \tag{5.5}$$

$$\mathbf{q}_k = \mathbf{B}_k\mathbf{q}_k + \mathbf{u} < \mathbf{p}_k, \mathbf{p}_k > \tag{5.6}$$

which, solved for u yield the following formula, commonly known as *Broyden's update* :

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{(\mathbf{q}_k - \mathbf{B}_k\mathbf{p}_k)\mathbf{p}_k^T}{< \mathbf{p}_k, \mathbf{p}_k >} \tag{5.7}$$

It is important to point out that eqn. (5.7) does not rely on any particular assumptions about $\mathbf{z}_k$ or $\mathbf{z}_{k+1}$; this leaves us complete freedom of choice for the line search algorithm.

Because we want an approximation to $\mathbf{F}^{-1}$ instead of $\mathbf{F}$, we let $\mathbf{H}_k = \mathbf{B}_k^{-1}$ and we make use of the following result, known as the Sherman-Morrison formula [39] :

Proposition 5.1 *If the matrix* $\mathbf{A}$ *is nonsingular and* $< \mathbf{v}, \mathbf{A}^{-1}\mathbf{u} > \neq -1$, *the matrix* $\mathbf{A} + \mathbf{u}\mathbf{v}^T$ *is also nonsingular, and its inverse is given by*

$$(\mathbf{A} + \mathbf{u}\mathbf{v}^T)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{u}\mathbf{v}^T\mathbf{A}^{-1}}{1 + < \mathbf{v}, \mathbf{A}^{-1}\mathbf{u} >}$$

Applying this formula to eqn. (5.7) leads to the following updating relation for the matrices $\mathbf{H}_k$:

$$\mathbf{H}_{k+1} = \mathbf{H}_k - \frac{(\mathbf{H}_k\mathbf{q}_k - \mathbf{p}_k)\mathbf{p}_k^T\mathbf{H}_k}{< \mathbf{p}_k, \mathbf{H}_k\mathbf{q}_k >} \tag{5.8}$$

which is the one actually used in our computer implementation of the algorithm. The following result is worth mentioning

Theorem 5.6 ([38]) *Let* $\hat{\mathbf{z}}$ *be a stationary point for c, and suppose that* $\mathbf{F}(\hat{\mathbf{z}})$ *is nonsingular and that* $\mathbf{F}(\mathbf{z})$ *is Lipschitz continuous in a neighborhood of* $\hat{\mathbf{z}}$. *Then there exist positive constants* ε, δ *such that the sequence*

$$\mathbf{z}_{k+1} = \mathbf{z}_k - \mathbf{H}_k\mathbf{g}(\mathbf{z}_k)$$

converges to $\hat{\mathbf{z}}$ *whenever* $\|\mathbf{z}_0 - \hat{\mathbf{z}}\| < \varepsilon, \|\mathbf{H}_0 - \mathbf{F}^{-1}(\hat{\mathbf{z}})\| < \delta$ *and the matrices* $\mathbf{H}_k$ *are updated according to eqn. (5.8).*

Incorporating Broyden's updates in the modified version of the steepest descent algorithm produces the following algorithm to find a saddle point of c.

Saddle Point Algorithm

Data : $\mathbf{H}_0 = \mathbf{U}, \mathbf{z}_0, \varepsilon, c.$

Step 0 : Set $k = 0$.

Step 1 : Set $\mathbf{g}_k = \mathbf{g}(\mathbf{z}_k), \mathbf{d}_k = -\mathbf{H}_k\mathbf{g}_k$.

Step 2 : Compute $\mathbf{z}_{k+1} = \mathbf{z}_k + t_k\mathbf{d}_k$ such that the equation $< \mathbf{U}\mathbf{g}_{k+1}, \mathbf{d}_k >= 0$ is satisfied.

Step 3 : If $\|\mathbf{g}_{k+1}\| < \varepsilon$ stop. Else set

$$\begin{aligned} \mathbf{p}_k &= \mathbf{z}_{k+1} - \mathbf{z}_k \\ \mathbf{q}_k &= \mathbf{g}_{k+1} - \mathbf{g}_k \\ \mathbf{H}_{k+1} &= \mathbf{H}_k - \frac{(\mathbf{H}_k\mathbf{q}_k - \mathbf{p}_k)\mathbf{p}_k^T\mathbf{H}_k}{< \mathbf{p}_k, \mathbf{H}_k\mathbf{q}_k >} \\ k &= k+1 \end{aligned}$$

and go to Step 1.

The behavior of the algorithm can be summarized in the following

Theorem 5.7 *Let* $\{\mathbf{z}_k\}$ *be a sequence generated by the previous algorithm, and suppose that* $\lim_{k\to\infty} \mathbf{z}_k = \hat{\mathbf{z}}$ *and* $\lim_{k\to\infty} \mathbf{H}_k = \mathbf{F}^{-1}(\hat{\mathbf{z}})$. *Then* $\hat{\mathbf{z}}$ *is a stationary point for* c.

Proof. Because $\lim_{k\to\infty} \mathbf{H}_k = \mathbf{F}^{-1}(\hat{\mathbf{z}})$, the matrices $\frac{1}{2}(\mathbf{U}\mathbf{H}_k + \mathbf{H}_k\mathbf{U})$ will be positive definite for large enough k. The rest of the proof follows the same argument used at the end of the previous section. □

From a practical point of view, it is not possible to be sure that the matrices $\frac{1}{2}(\mathbf{U}\mathbf{H}_k+\mathbf{H}_k\mathbf{U})$ remain positive definite. A tentative check can be made by monitoring the sign of the inner product $< \mathbf{p}_k, \mathbf{H}_k\mathbf{q}_k > \cong < \mathbf{p}_k, \mathbf{H}_k\mathbf{F}(\mathbf{z}_k)\mathbf{p}_k >$, which should be positive at least for large k, because $\mathbf{H}_k$ is supposed to approximate $\mathbf{F}^{-1}(\mathbf{z}_k)$. In the computer implementation, the algorithm is restarted whenever that inner product becomes negative. As mentioned previously, the numerical performance of this method has proven to be very satisfactory, even if at the moment no theoretical results about its global convergence properties are available.

Chapter 6

Computer Implementation

In this chapter I explain how I implemented the algorithms presented in the previous chapters in a computer program that can be used to find macromodels for different types of functional blocks occurring in the design of integrated circuits. First I give some details of certain features of the program that are relevant from a computational point of view, such as parametrization of the input functions. Then I present a few circuits and the corresponding macromodels computed by the program, and I discuss the results so obtained.

6.1 Input Parametrization

In the previous chapter I developed an algorithm that can be used to find a saddle point of the cost function. This means that at the same time we can try to maximize the cost with respect to the input and minimize it with respect to the macromodel.

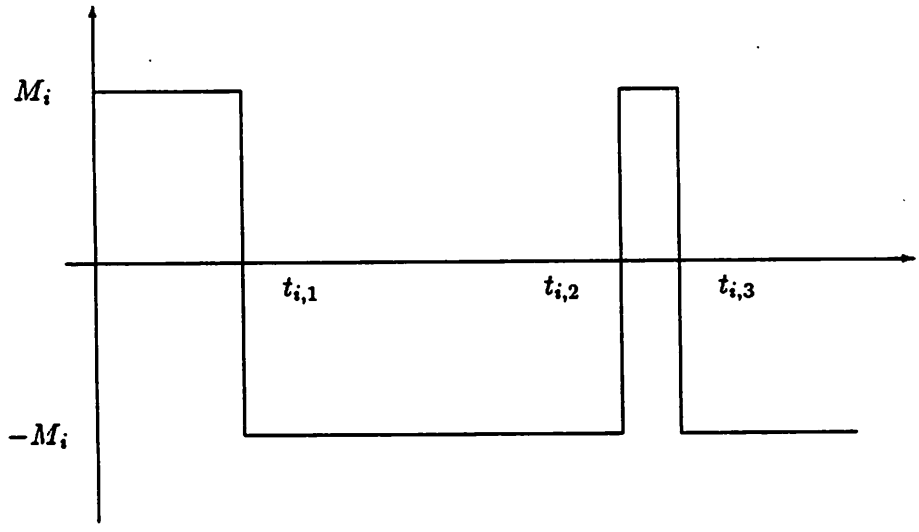


Figure 6.1: Input parametrization according to switching points

In order to apply this algorithm to our case, the inputs must be parametrized in some way. This can be done following an idea of Gonzalez and Rofman [40].

We have already seen that only inputs that are piecewise constants and that oscillate between their upper and lower bounds have to be considered. Therefore the only unknowns that remain are the number and the positions of the switching times for each component $u_i(t)$ of the input vector u. Arbitrarily fix the maximum number of switching points that each component can have; this is a parameter (let us call it N) that the user can set in the input deck describing the circuit. For each input u_i we denote its switching points by $(t_{i,1}, \ldots, t_{i,N})$ and its bound by $M_i > 0$ (see Fig. 6.1); for notational convenience we define $t_{i,0} = 0$ and $t_{i,N+1} = T$. Obviously we must have $t_{i,0} \leq t_{i,1} \leq \ldots \leq t_{i,N+1}$. For every $t \in [0,T]$ the value of $u_i(t)$ is computed as follows : let j be that index such that $t_{i,j} \leq t < t_{i,j+1}$; then $u_i(t) = (-1)^j M_i$. This completely determines the input, which can be therefore parametrized by a vector $(t_{1,1}, \ldots, t_{m,N})$

containing mN components, where m is the number of inputs.

After finding a parametrization for the input, we have to compute the gradient of the cost with respect to parameter variations; this can be done in the following way. Suppose that $\frac{\partial c}{\partial t_{i,j}}$ has to be computed; in order to simplify notation, drop the index i and denote the switching point simply by t_j. The input u (actually only its *i-th* component u_i) depends both on the time variable t and on the parameter t_j; to distinguish clearly between the two, we will use the notation $\mathbf{u}(t;t_j)$ (or $u_i(t;t_j)$). In this particular case, eqn. (4.21) becomes

$$\frac{\partial c}{\partial t_j} = \lim_{\Delta t \to 0} \int_0^T \frac{H[\mathbf{v}(\tau), \boldsymbol{\lambda}(\tau), \mathbf{u}(\tau; t_j + \Delta t)] - H[\mathbf{v}(\tau), \boldsymbol{\lambda}(\tau), \mathbf{u}(\tau; t_j)]}{\Delta t} d\tau \qquad (6.1)$$

The input u_i will be connected to node n_1 in the original system and to node n_2 in the macromodel; let λ_{n_1} and λ_{n_2} be the corresponding Lagrange multipliers. Then we have

$$\begin{aligned} & H[\mathbf{v}(\tau), \boldsymbol{\lambda}(\tau), \mathbf{u}(\tau; t_j + \Delta t)] - H[\mathbf{v}(\tau), \boldsymbol{\lambda}(\tau), \mathbf{u}(\tau; t_j)] = \\ &= \boldsymbol{\lambda}^T(\mathbf{f}[\mathbf{v}(\tau)] + \mathbf{u}(\tau; t_j + \Delta t)) - \boldsymbol{\lambda}^T(\mathbf{f}[\mathbf{v}(\tau)] + \mathbf{u}(\tau; t_j)) = \\ &= \boldsymbol{\lambda}^T(\mathbf{u}(\tau; t_j + \Delta t) - \mathbf{u}(\tau; t_j)) = \\ &= (\lambda_{n_1} + \lambda_{n_2})(u_i(\tau; t_j + \Delta t) - u_i(\tau; t_j)) \end{aligned}$$

But $u_i(\tau; t_j + \Delta t) - u_i(\tau; t_j)$ is equal to zero, unless $\tau \in [t_j + \Delta t, t_j]$, in which case it is equal to $(-1)^{j+1} M_i$. Therefore eqn. (6.1) becomes

$$\frac{\partial c}{\partial t_j} = (-1)^{j+1} M_i \lim_{\Delta t \to 0} \frac{1}{\Delta t} \int_{t_j}^{t_j + \Delta t} (\lambda_{n_1}(\tau) + \lambda_{n_2}(\tau))\, d\tau = (-1)^{j+1} M_i (\lambda_{n_1}(t_j) + \lambda_{n_2}(t_j)) \qquad (6.2)$$

This gives us the expression for $\frac{\partial c}{\partial t_j}$ that we were looking for.

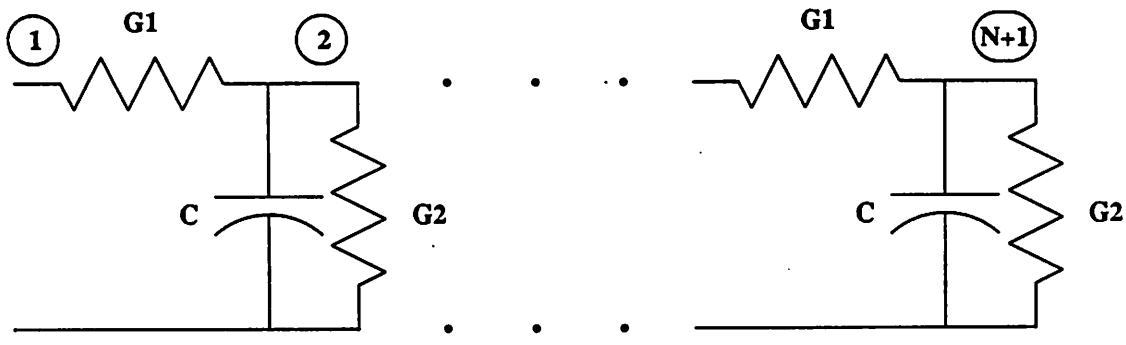


Figure 6.2: RC ladder model for a neuron

6.2 Experimental Results

6.2.1 Distributed RC Line

In the study of biological neurons [41], one direction of investigation is the development of mathematical models suitable for simulating the behavior of the neuron on a digital computer [42]. One of the components of the neuron, the dendrite, is usually represented by a distributed RC line. Because distributed elements are difficult to simulate using standard techniques, the RC line itself is approximated with an RC ladder containing N identical cells (see Fig. 6.2). The element values are determined by the equations $G_1 = NG_a, G_2 = G_l/N, C_1 = C/N$, where G_a is the axial conductance of the distributed line, G_l is the longitudinal conductance and C is the total capacitance. Obviously the larger N is, the closer the ladder approximates the behavior of a distributed line. However it is conceivable that a ladder whose cells are allowed to be different could achieve the same degree of accuracy as one containing a larger number of cells which are all identical. To test this hypothesis as well as the performance of the macromodeling algorithm we took an RC ladder as in Fig. 6.2 with $N = 40$. The

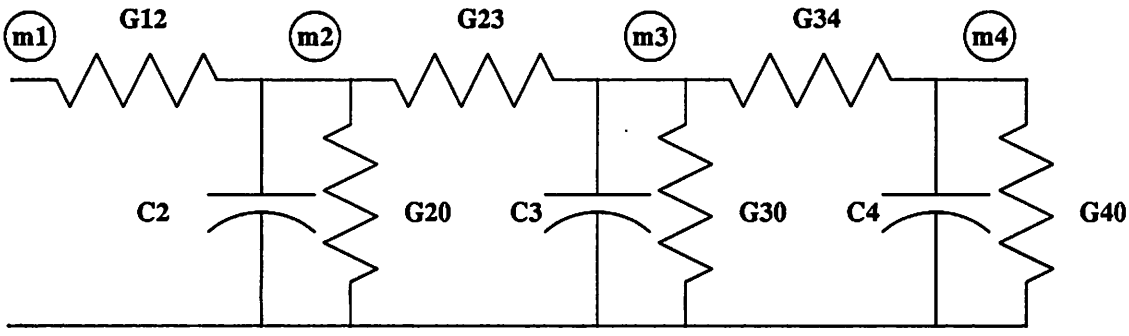


Figure 6.3: Reduced RC ladder model for a neuron

values of G_a, G_l and C are 4.71e-2 mho, 1.13e-2 mho and .452 farad respectively, giving values for G_1, G_2 and C of 1.88 mho, 2.83e-4 mho and 1.13e-2 farad. We modeled this ladder with a smaller one containing only 3 cells (Fig. 6.3). The initial element values were computed using the above formulas with $N = 3$, giving an initial model with 3 identical cells, but all the element values were allowed to vary, for a total of 9 model parameters to be optimized. Both end nodes were inputs as well as outputs, with node 1 in Fig. 6.2 corresponding to node *m1* in Fig. 6.3 and node $N + 1$ corresponding to node *m4*. Optimizing this model took approximately 670 seconds of CPU time on a Microvax II running Ultrix[1] V2.0. The element values of the optimized model are reported in Table 6.1.

The accuracy of the optimized model can be judged by looking at the following graphs. Fig. 6.4 shows the output voltages at nodes 1 and *m1* before optimization; the graph of their difference is shown in Fig. 6.5. The output voltages at nodes 41 and *m4* and their difference are plotted in Figs. 6.6 and 6.7 respectively. The corresponding inputs are square current waveforms chosen arbitrarily as an initial guess to the worst-

[1] Ultrix is a trademark of Digital Equipment Corporation.

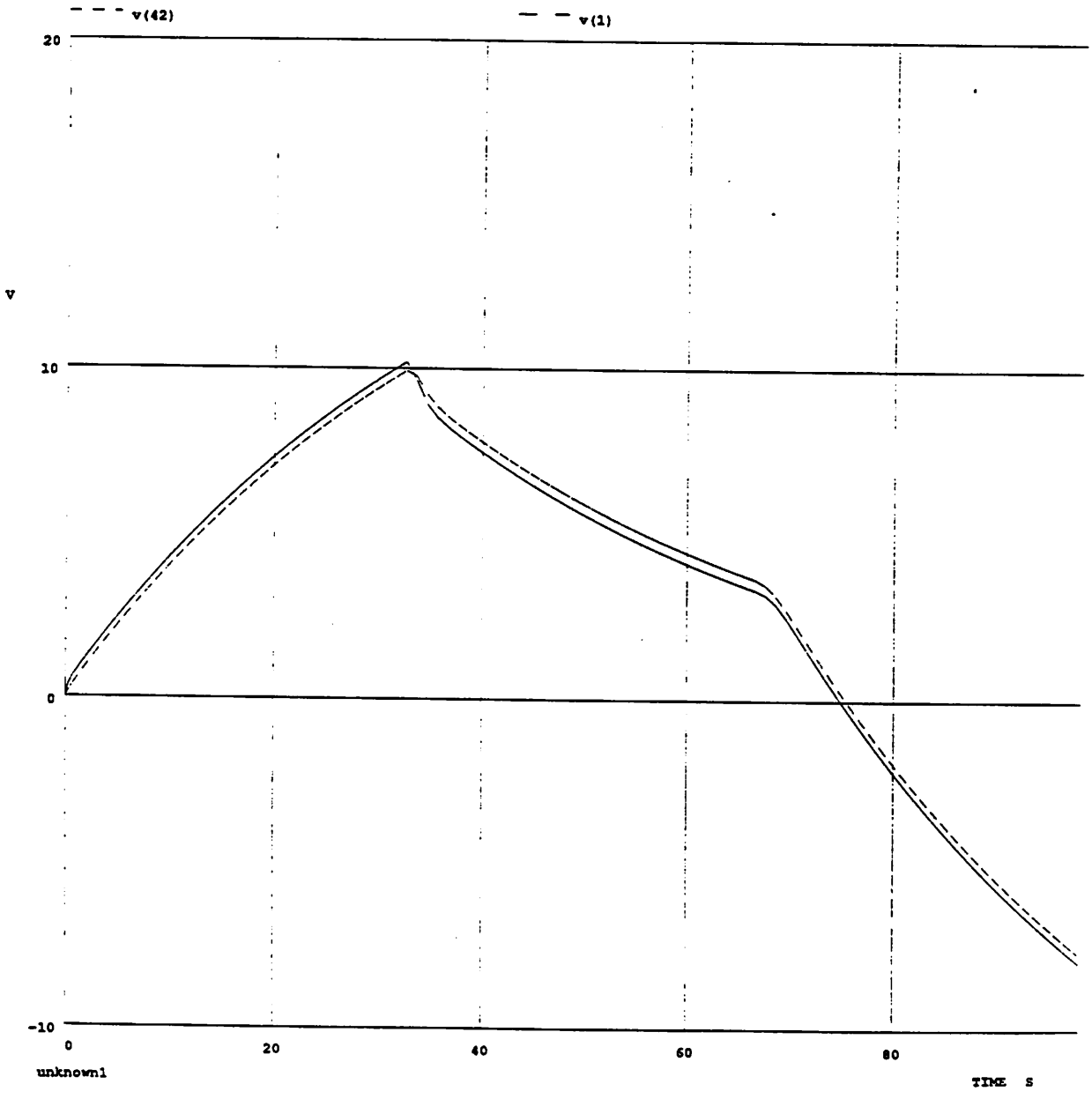


Figure 6.4: Voltages at nodes 1 and *m1* before optimization

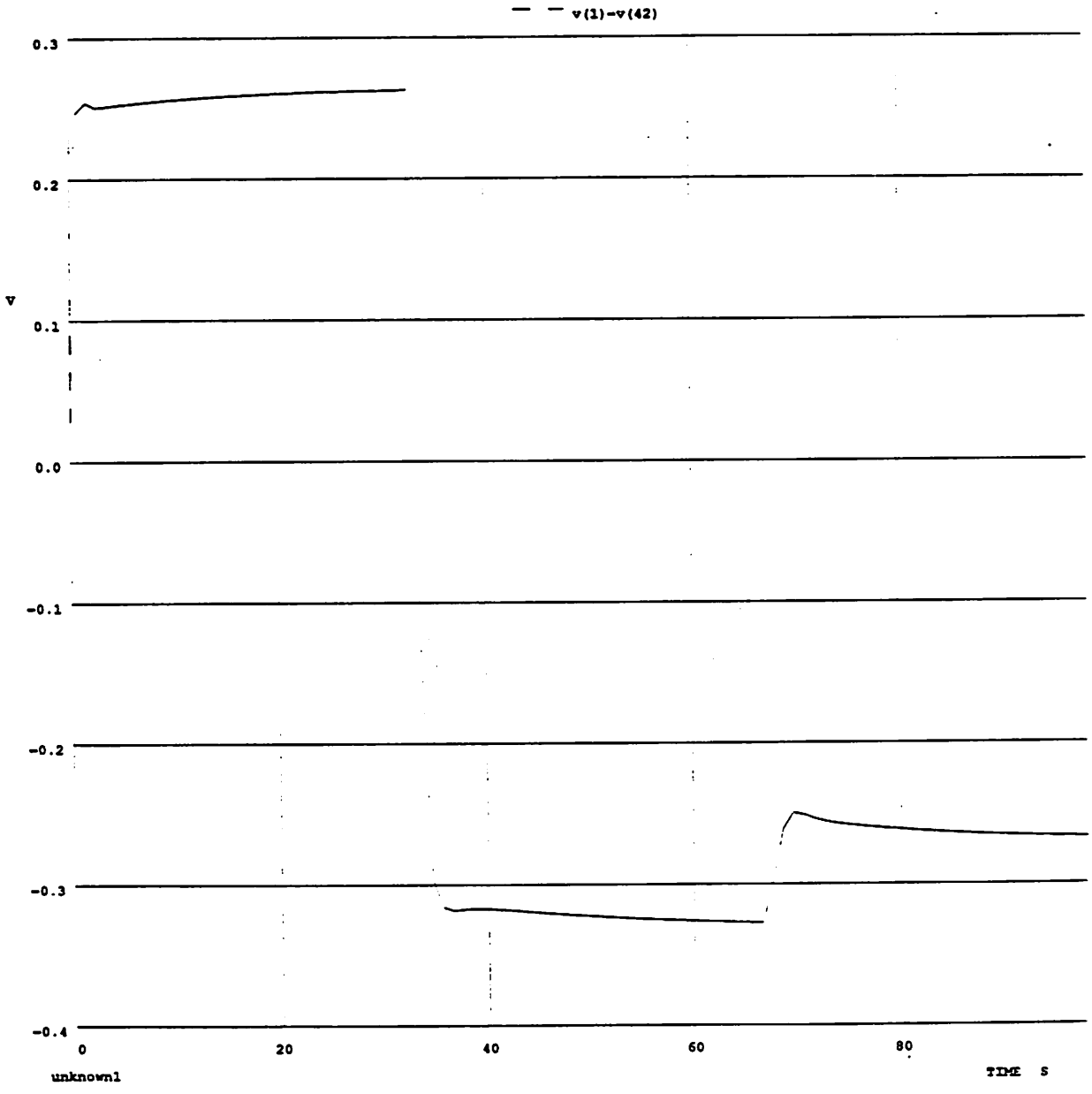


Figure 6.5: Difference between voltages at nodes 1 and *m1* before optimization

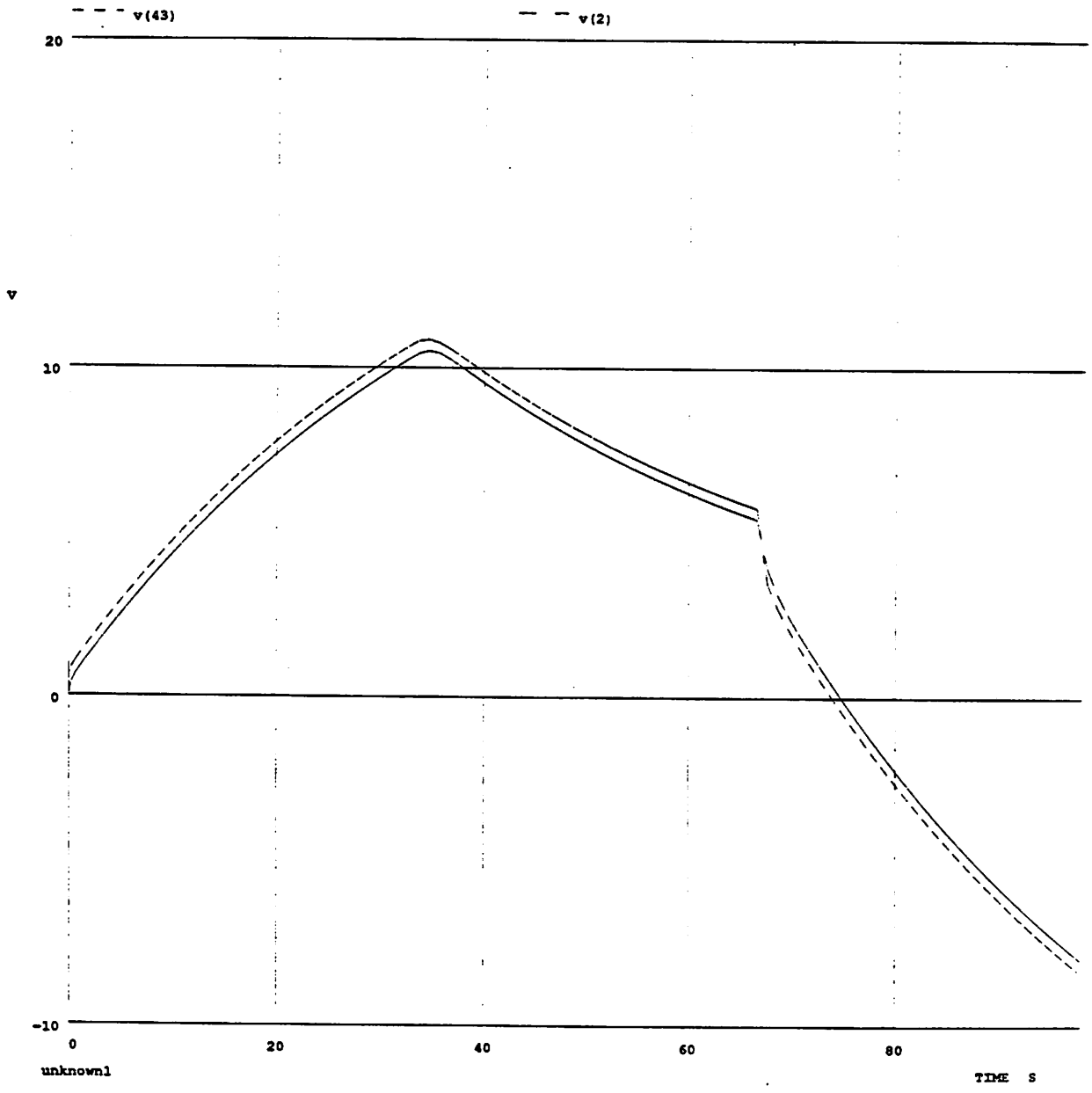


Figure 6.6: Voltages at nodes 41 and $m4$ before optimization

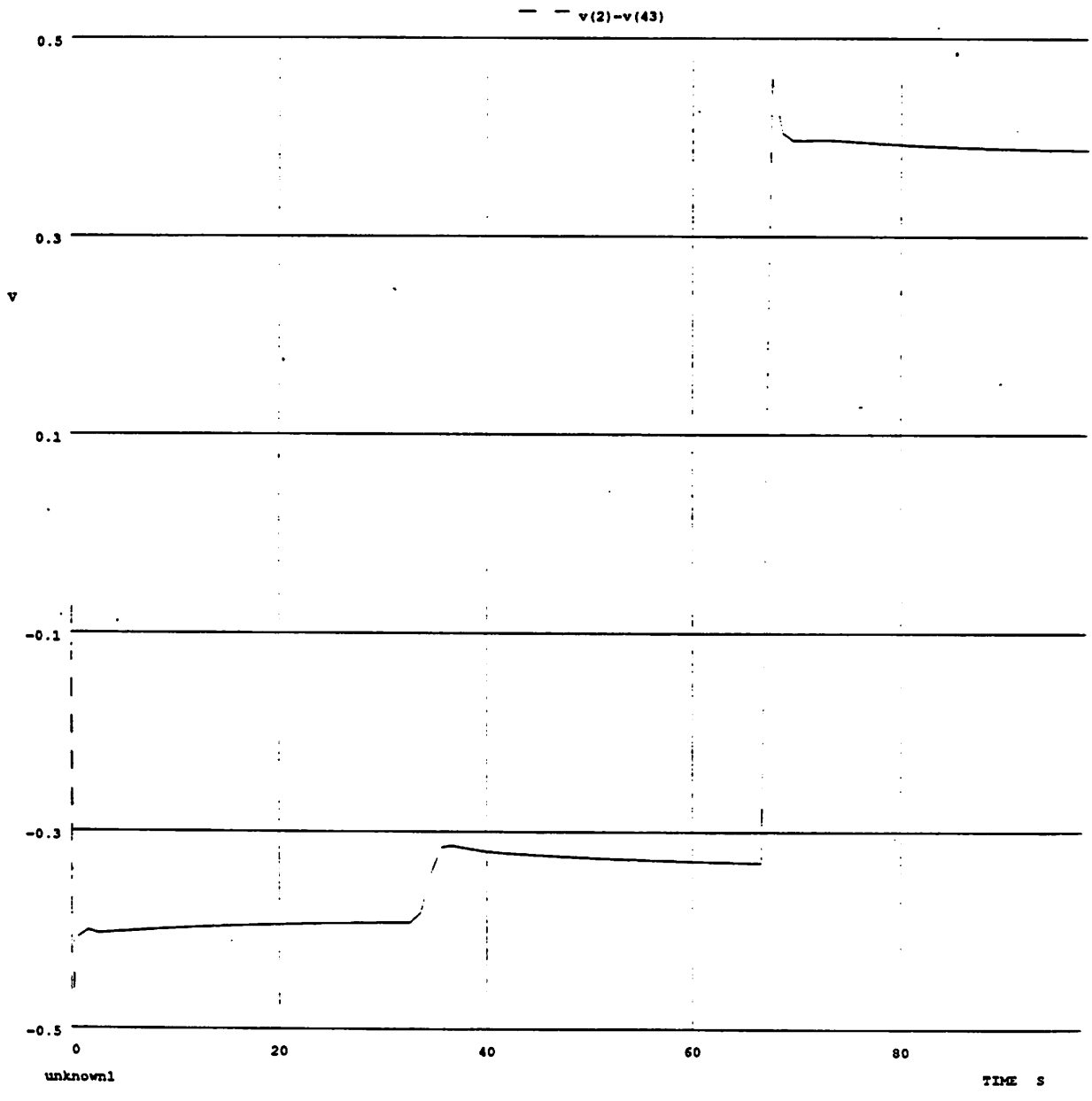


Figure 6.7: Difference between voltages at nodes 41 and $m4$ before optimization

G_{12}	1.81e-1	mho
G_{20}	3.82e-3	mho
C_2	1.62e-2	F
G_{23}	6.15e-1	mho
G_{30}	3.78e-3	mho
C_3	1.59e-1	F
G_{34}	7.19e-2	mho
G_{40}	3.69e-3	mho
C_4	1.33e-1	F

Table 6.1: Element values for the optimized neuron model

case inputs. After optimization the waveforms at the output nodes of the two ladders are indistinguishable, and therefore they are not plotted. An idea of the improvement in the approximation is given by the change in the cost function, whose value decreased from 10.995 (before optimization) to 0.245 (after optimization). The improvement is shown graphically in Figs. 6.8 and 6.9, which show the voltage differences at the end nodes after optimization. The input currents are those computed by the program as being the worst-case input (of course we are not guaranteed that this is true globally, but only with respect to neighboring inputs). It can be seen that while the maximum values of the differences remain substantially the same, the area decreases substantially, corresponding to the fact that we are optimizing the L^2 norm.

The performance of the optimized model as a part of a complete neural circuit is displayed in Fig. 6.10, which shows the voltage waveforms at one end of the neuron modeled in three different ways : with an RC ladder of 40 identical cells (40ident), with the optimized 3-cell model computed above (3-cell model) and with RC ladder of 3 identical cells (3ident). The two spikes are blown up in Figs. 6.11 and 6.12. The improvement is noticeable, especially as far the amplitude of the two spikes is

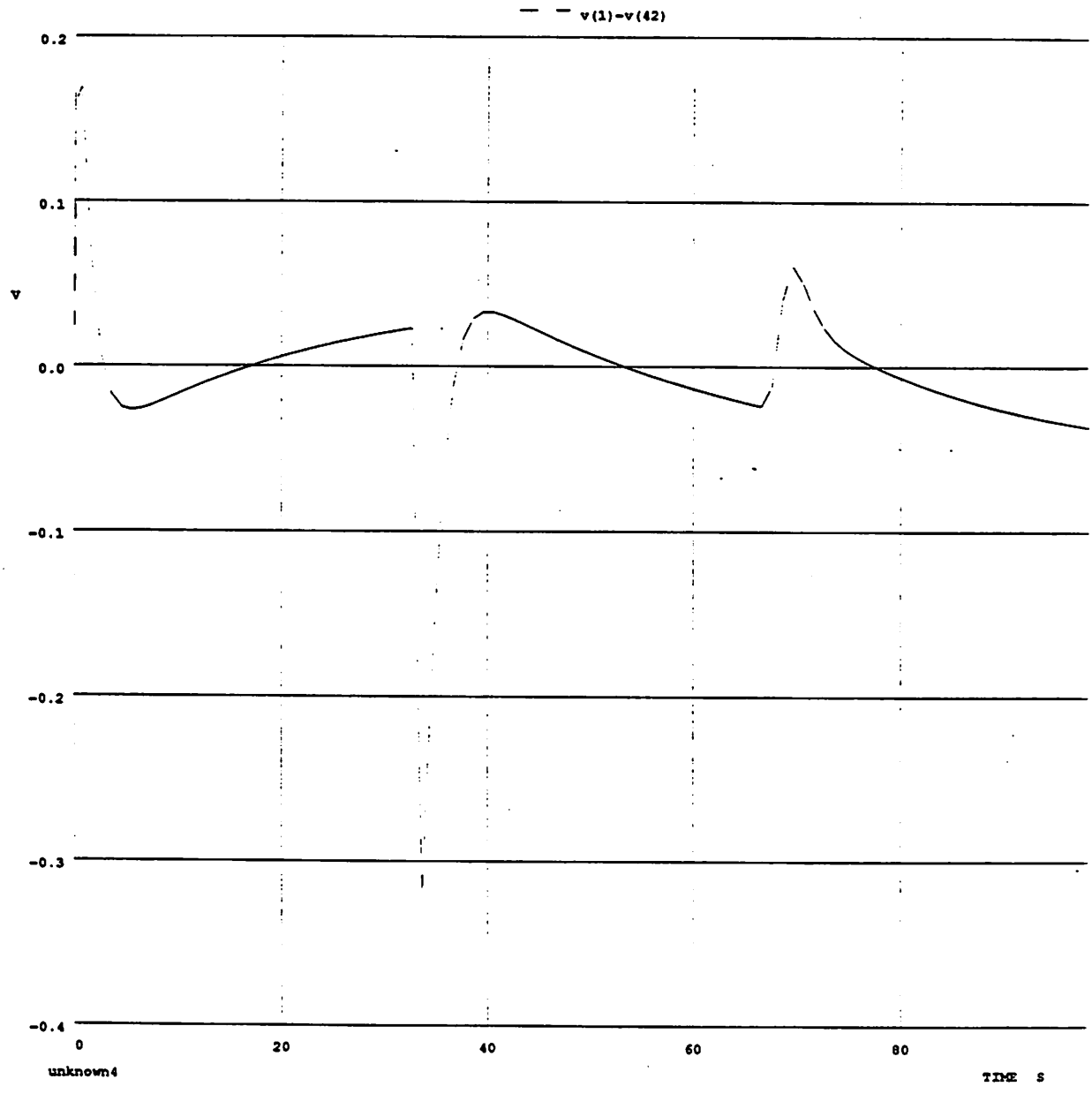


Figure 6.8: Difference between voltages at nodes 1 and *m1* after optimization

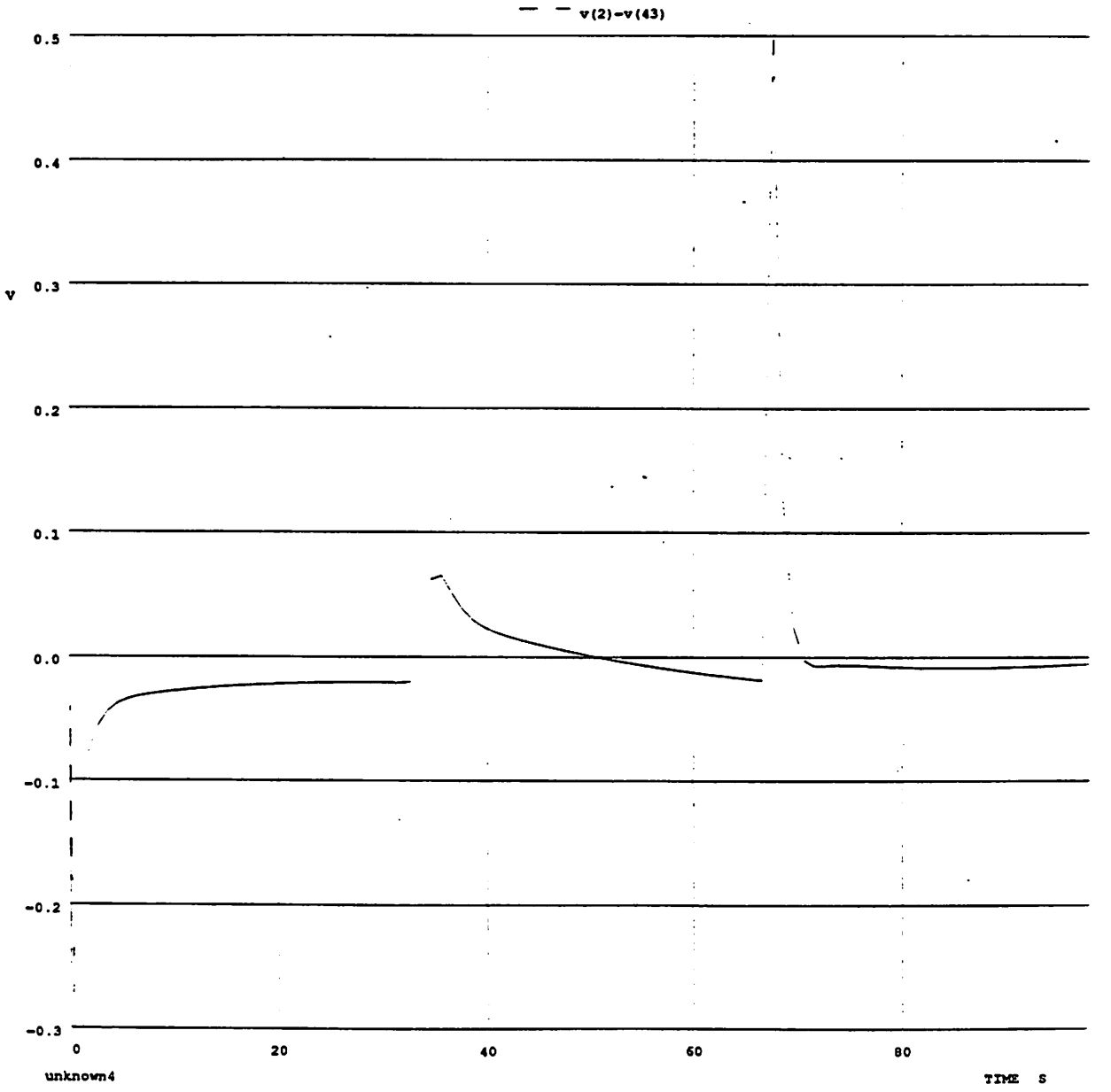


Figure 6.9: Difference between voltages at nodes 41 and $m4$ after optimization

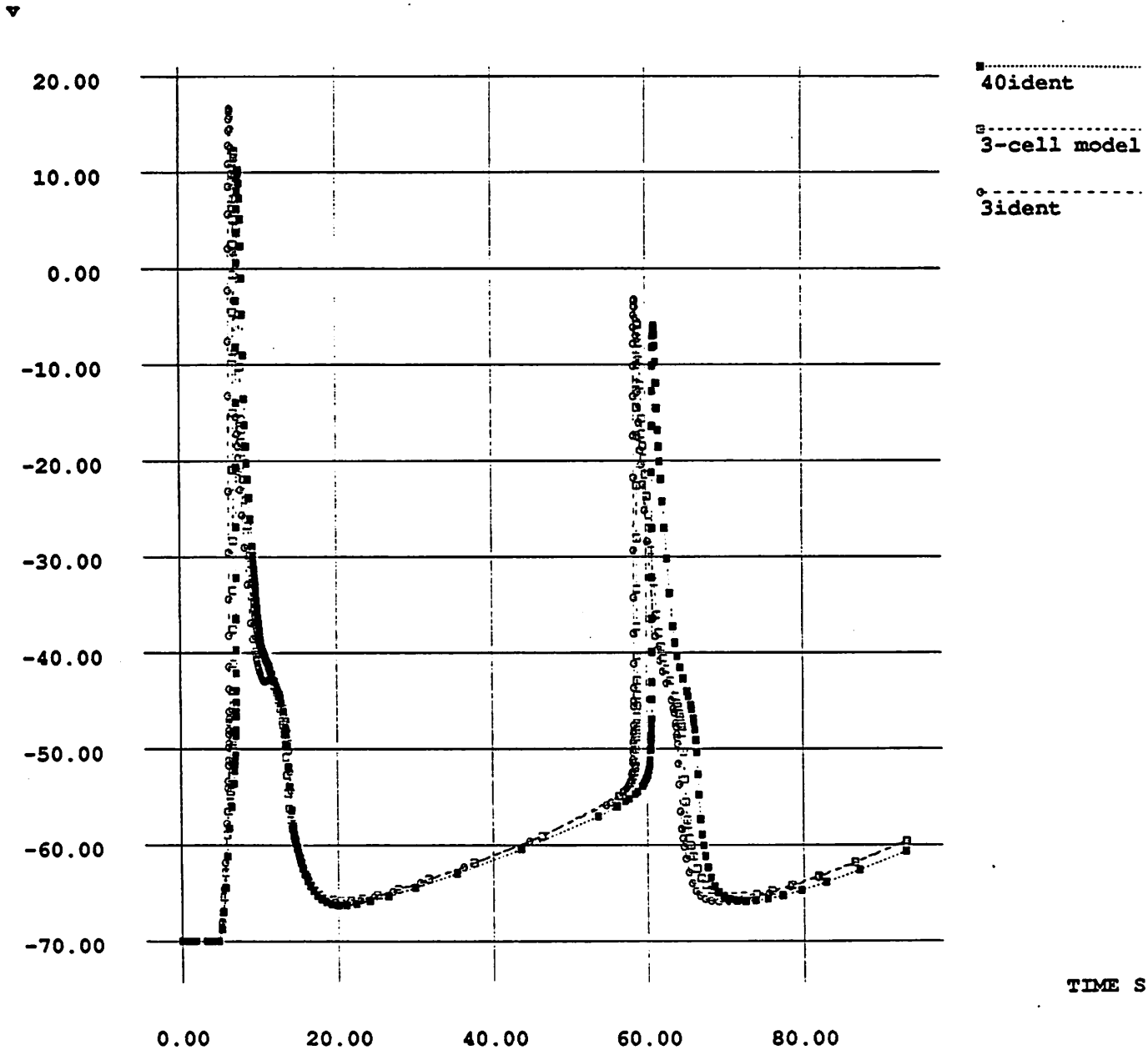


Figure 6.10: Voltages corresponding to three different neuron models

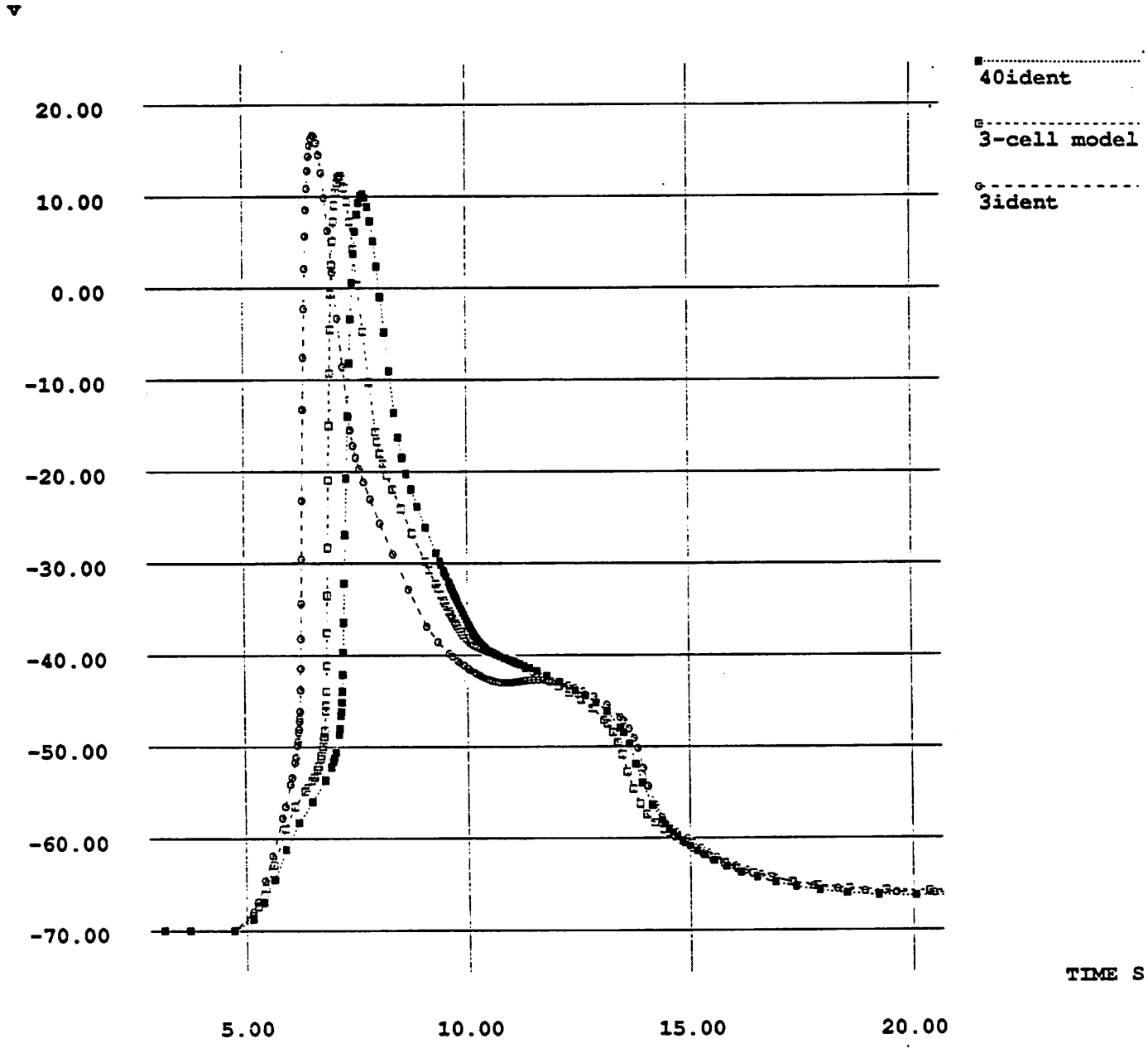


Figure 6.11: Blow-up of the first voltage spike

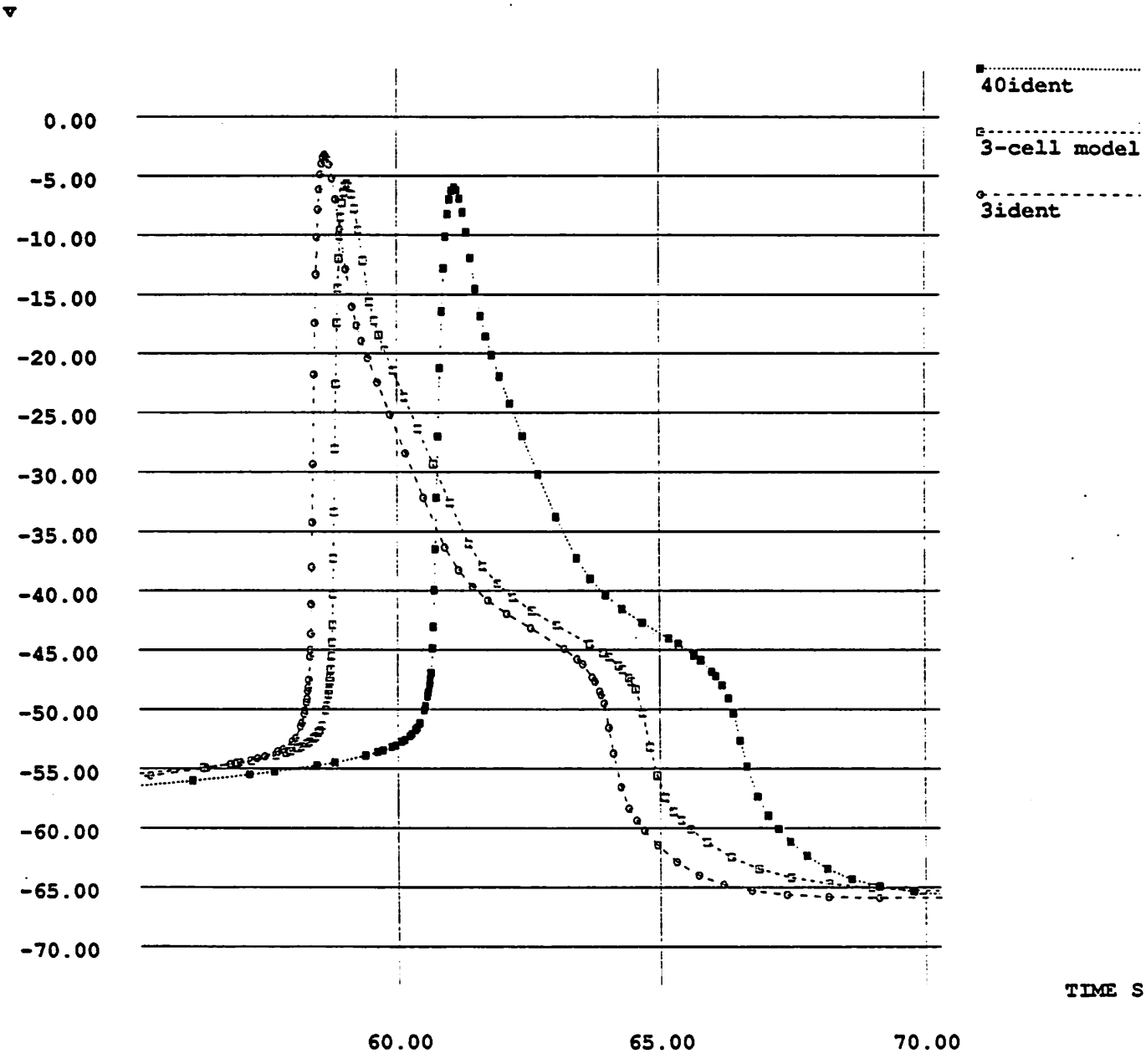


Figure 6.12: Blow-up of the second voltage spike

concerned : apart from a small amount of time lead, the waveform corresponding to the optimized model is an almost identical copy of the waveform generated by the 40-cell ladder. The difference in time is explained by the fact the the spikes are generated by two highly nonlinear elements placed at both ends of the neuron : when the voltage at one end of the neuron reaches a certain threshold the spike is triggered. Because this threshold is reached by the 3-cell model before the 40-cell ladder, the corresponding spikes occur at different times. It can be argued that including the nonlinear elements when performing the model optimization could result in improved accuracy.

6.3 Conclusions

In this thesis, a new approach to the macromodeling problem was introduced. A review of the existing literature on this topic showed that the methods proposed by other authors suffer from limitations that make them unsuitable for general purpose circuit simulation. The algorithm presented in this thesis is based exclusively on an input-output representation of the circuit to be modeled. Because no reliance on any particular properties of the circuit is assumed, the algorithm can be used to model a very wide class of circuits.

From a mathematical point of view, this approach to the macromodeling problem is equivalent to solving a min-max problem in an abstract space whose elements are dynamical systems. The distance between two systems in this space is defined in terms of the maximum difference of their outputs as the input ranges over a set of admissible waveforms. In one of the main results of this thesis, it is shown that only

a relatively small class of inputs need be considered: it is this result that makes the approach pursued here attractive from a practical point of view as well as theoretically interesting.

The practical implementation of the algorithm requires the computation of the derivatives of the distance functional. This the same problem as the computation of network sensitivities in the time domain, discussed for instance in [29] and [16]. However in this thesis the relevant formulas are derived following a different approach, which is based on an extension of the Hamiltonian formulation of a classical optimal control problem. In this way variations in the input and in the model parameters can be dealt with in a unified setting.

The method used for the numerical solution of the min-max problem is a modification of an algorithm originally developed for the solution of nonlinear systems of algebraic equations. This particular choice was dictated by the need to satisfy conflicting requirements about speed and computational cost, while at the same time guaranteeing convergence to a stationary points. It was shown that variable metric algorithms, which are very popular for minimizing functions, rely heavily on the positive definiteness of the Hessian and cannot be modified easily to solve min-max problem. Broyden's method, on the other hand, does not depend on any particular properties either of the Hessian or of the line search, and is therefore much more flexible. It was shown that the modified version of Broyden's method presented here generates sequences whose limit points are stationary points of the cost function.

The algorithm was implemented in a computer program that generates models of electrical networks. Both the input and the output are circuit descriptions in ordinary

SPICE format, so that the models can be read by any circuit simulator using the same interface. Numerical tests were performed on a 40-cell RC ladder representing the dendrite of a neuron. The results obtained show that a 3-cell ladder generated by the program can attain almost the same accuracy. It is conceivable that even greater accuracy could be achieved if two highly nonlinear elements present at both ends of the dendrite were included in the simulations performed during the modeling process. The CPU time required was modest and certainly worth the savings that can be obtained using the smaller model in just a few simulations.

From a theoretical point of view, there are several points that are worth investigating further into. In particular, it would be helpful to have more detailed information about the convergence properties, both local and global, of the algorithms for the location of saddle points of functions. Another interesting question concerns the integration of the differential equations after a change in the parameters describing the model : the algorithm, as currently implemented, does not use any information that could be gained from the waveforms corresponding to the previous values of the model parameters. This approach is clearly simple-minded, because it is conceivable that the old waveforms could be used as a starting point for the computation of the new ones, especially in the final phases of the modeling process, when the changes in the model parameters are likely to be very small.

From a practical point of view, the algorithm needs to be tested on nonlinear circuits, both analog and digital, in order to assess its effectiveness as a general purpose tool to generate models for circuit simulation. This requires the implementation of nonlinear devices such as MOSFETS. Another useful feature would be the capability

to sweep one or more parameters in the system to be modeled, and to generate a corresponding table of models. This would make it particularly easy to generate models for whole classes of circuits having the same topology but different electrical parameters (e.g. NMOS inverters with different values of the pull-up/pull-down ratios). Once generated, the models could be stored in a cell library and thus be ready for use whenever a simulation of a circuit containing that particular cell is required.

Bibliography

[1] J. K. White and A. Sangiovanni-Vincentelli, *Relaxation Techniques for the Simulation of VLSI Circuits.* Norwell, Massachusetts 02061: Kluwer Academic Publishers, 1987.

[2] J. Ousterhout, "Switch-Level Delay Models for Digital MOS VLSI," in *Proceedings of the 21st Design Automation Conference*, pp. 542–548, IEEE, June 1984.

[3] J. Rubinstein, P. Penfield, Jr., and M. A. Horowitz, "Signal Delay in RC Tree Networks," *IEEE Transactions on Computer-Aided Design*, vol. CAD-2, no. 3, pp. 202–211, 1983.

[4] T. Lin and C. A. Mead, "Signal Delay in General RC Networks with Application to Timing Simulation of Digital Integrated Circuits," in *Proceedings of the Conference on Advanced Research in VLSI*, pp. 93–99, MIT, January 1984.

[5] W. C. Elmore, "The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers," *Journal of Applied Physics*, vol. 19, no. 1, pp. 55–63, 1948.

[6] M. D. Matson, "Macromodeling of Digital MOS VLSI Circuits," November 1984. MIT Technical Report VLSI Memo No. 84-212.

[7] S. R. Nassif and S. W. Director, "WASIM : A Waveform Based Simulator for VLSICs," in *Proceedings of the 1985 International Conference on Computer-Aided Design*, pp. 29–31, IEEE, November 1985.

[8] W. M. Coughran, Jr., E. Grosse, and D. J. Rose, "CAzM : A Circuit Analyzer with Macromodeling," *IEEE Transactions on Electron Devices*, vol. ED-30, no. 9, pp. 1207–1213, 1983.

[9] B. C. Moore, "Principal Component Analysis in Linear Systems : Controllability, Observability, and Model Reduction," *IEEE Transactions on Automatic Control*, vol. AC-26, no. 1, pp. 17–32, 1981.

[10] R. E. Kalman, "Mathematical Description of Linear Systems," *SIAM Journal on Control*, vol. 1, pp. 152–192, 1963.

[11] T. Kailath, *Linear Systems. Prentice-Hall Information and System Sciences Series*, Englewood Cliffs, New Jersey 07632: Prentice-Hall, Inc., 1980.

[12] G. H. Golub and C. Reinsch, "Singular Value Decomposition and Least Squares Solutions," *Numerische Mathematik*, vol. 14, pp. 403–420, 1970.

[13] G. H. Golub and C. F. Van Loan, *Matrix Computations.* Baltimore: Johns Hopkins University Press, 1983.

[14] S. Kung and D. W. Lin, "Optimal Hankel-Norm Model Reductions : Multivariate Systems," *IEEE Transactions on Automatic Control*, vol. AC-26, no. 4, pp. 832–852, 1981.

[15] K. Glover, "All optimal Hankel-norm approximations of linear multivariable systems and their L^∞-error bounds," *International Journal of Control*, vol. 39, no. 6, pp. 1115–1193, 1984.

[16] G. D. Hachtel and R. A. Rohrer, "Techniques for the Optimal Design and Synthesis of Switching Circuits," *Proceedings of the IEEE*, vol. 55, no. 11, pp. 1864–1877, 1967.

[17] J. R. Munkres, *Topology : A First Course.* Englewood Cliffs, New Jersey 07632: Prentice-Hall, Inc., 1975.

[18] L. V. Kantorovich and G. P. Akilov, *Functional Analysis in Normed Spaces.* Vol. 46 of *International Series of Monographs in Pure and Applied Mathematics*, Oxford, England: Pergamon Press Limited, 1964.

[19] A. E. Taylor and D. C. Lay, *Introduction to Functional Analysis.* New York, New York: John Wiley & Sons, 1980.

[20] H. L. Royden, *Real Analysis.* New York, New York: The Macmillan Company, 1966.

[21] J. K. Hale, *Ordinary Differential Equations.* New York: Wiley-Interscience, 1969.

[22] A. E. Bryson, Jr. and Y. Ho, *Applied Optimal Control.* Waltham, Massachusetts: Ginn and Company, 1969.

[23] L. S. Pontryagin, V. G. Boltyanski, R. V. Gamkrelidge, and E. F. Mischenko, *The Mathematical Theory of Optimal Processes.* New York, New York: Interscience, 1962.

[24] L. O. Chua, C. A. Desoer, and E. S. Kuh, *Linear and Nonlinear Circuits. McGraw-Hill Series in Electrical Engineering*, New York St. Louis San Francisco: McGraw-Hill Book Company, 1987.

[25] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables.* New York, New York: Academic Press, 1970.

[26] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations.* Englewood Cliffs, New Jersey 07632: Prentice-Hall, Inc., 1971.

[27] L. O. Chua and P. Lin, *Compter-Aided Analysis of Electronic Circuits. Prentice-Hall Series in Electrical and Computer Engineering*, Englewood Cliffs, New Jersey 07632: Prentice-Hall, Inc., 1975.

[28] R. K. Brayton, F. G. Gustavson, and G. D. Hachtel, "A New Efficient Algorithm for Solving Differential-Algebraic Systems Using Implicit Backward Differentiation Formulas," *Proceedings of the IEEE*, vol. 60, no. 1, pp. 98–108, 1972.

[29] S. W. Director and R. A. Rohrer, "The Generalized Adjoint Network and Network Sensitivities," *IEEE Transactions on Circuit Theory*, vol. CT-16, no. 3, pp. 318–323, 1969.

[30] C. A. Desoer, "Teaching Adjoint Networks to Juniors," *IEEE Transactions on Education*, vol. E-16, no. 1, pp. 10–14, 1973.

[31] W. J. McCalla, *Fundamentals of Computer-Aided Circuit Simulation.* Norwell, Massachusetts 02061: Kluwer Academic Publishers, 1988.

[32] N. Z. Shor, *Minimization Methods for Non-Differentiable Functions.* Berlin New York: Springer Verlag, 1985.

[33] H. Y. Huang, "Unified Approach to Quadratically Convergent Algorithms for Function Minimization," *Journal of Optimization Theory and Applications*, vol. 5, no. 6, pp. 405–423, 1970.

[34] M. Avriel, *Nonlinear Programming : Analysis and Methods.* Englewood Cliffs, New Jersey 07632: Prentice-Hall, Inc., 1976.

[35] D. G. Luenberger, *Linear and Nonlinear Programming.* Reading, Massachusetts: Addison-Wesley Publishing Company, second ed., 1984.

[36] J. E. Sinclair and R. Fletcher, "A New Method of Saddle-Point Location for the Calculation of Defect Migration Energies," *Journal of Physics C : Solid State Physics*, vol. 7, pp. 864–870, 1974.

[37] C. G. Broyden, "A Class of Methods for Solving Nonlinear Simultaneous Equations," *Mathematics of Computation*, vol. 19, no. 92, pp. 577–593, 1965.

[38] J. E. Dennis, Jr., "On the Convergence of Broyden's Method for Nonlinear Systems of Equations," *Mathematics of Computation*, vol. 25, no. 115, pp. 559–567, 1971.

[39] A. S. Householder, *The Theory of Matrices in Numerical Analysis.* New York: Blaisdell Publishing Company, 1964.

[40] R. Gonzalez and E. Rofman, "On bang-bang Control Policies," in *Optimization Techniques, Part 2*, (J. Cea, ed.), pp. 587–602, Springer-Verlag, 1976. Proceedings, 7th IFIP Conference; Nice, September 8-12,1975.

[41] E. R. Kandel and J. H. Schwartz, eds., *Principles of Neural Science.* New York Amsterdam Oxford: Elsevier/Noth-Holland, 1981.

[42] L. J. Borg-Graham, *Modelling the Somatic Electrical Response of Hippocampal Pyramidal Neurons.* Master's thesis, Massachusetts Institute of Technology, May 1987.