EXPLORING THE RELATIONSHIPS BETWEEN DEVICE-MODEL REPRESENTATION AND CIRCUIT ANALYSIS

by

Christopher K. Lennard

Memorandum No. UCB/ERL M92/61

5 June 1992

EXPLORING THE RELATIONSHIPS BETWEEN DEVICE-MODEL REPRESENTATION AND CIRCUIT ANALYSIS

by

Christopher K. Lennard

Memorandum No. UCB/ERL M92/61

5 June 1992

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

TITLE PAGE

EXPLORING THE RELATIONSHIPS BETWEEN DEVICE-MODEL REPRESENTATION AND CIRCUIT ANALYSIS

by

Christopher K. Lennard

Memorandum No. UCB/ERL M92/61

5 June 1992

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Exploring the Relationships Between Device-Model Representation and Circuit Analysis

Christopher K. Lennard

University of California
Berkeley, CA 94720

Department of Electrical Engineering
and Computer Sciences

Abstract

To date, no investigation into the effect of device modelling upon circuit analysis has been completed except in the case of piecewise linear representations. In this report a theoretical study of analysis techniques which could exploit the properties of low-order piecewise polynomial model descriptions is presented. The applicability is examined for analysis at both the nonlinear and differential equation level. The only approach which appears to present hope for further speed-up, beyond direct exploitation of the ease in computing derivatives for the Jacobian matrix construction, is one allowing selective updating of the set of piecewise-polynomial circuit relations.

Professor A. Richard Newton
Research Advisor

Acknowledgements

I would like to thank my research advisor, Prof. A. Richard Newton, whose drive finally motivated me to organise this material into a coherent whole. I also extend my gratitutude to those who persevered in reading this work and offering constructive criticism, namely Prof. Donald Pederson, J. Mark Noworolski and Jaijeet Roychowdhury.

Thanks, of course, are due also to some who may never read this report but whose friendship and support are as critical to its creation as any technical discussions: Mark, Gary, Adrian, Don, Richard, Abhijit, Eric, Dorothy, Sara, Nicola, Michelle, Lilia, Mum, Dad and Toffa.

I would like to acknowledge the author of the spell check program because of the '-b' (British) option and the restauranteurs of North Side for providing sufficient variety for a year's worth of lunches.

This work was supported in part by the Semiconductor Research Corporation under contract number 91-DC-008 and by the Digital Equipment Corporation. Their support is gratefully acknowledged.

Contents

List of Figures

Chapter 1

Introduction

Circuit simulation has been a critical part of integrated circuit (IC) design since the 1970's. Beyond its use as a verification tool, circuit simulators have become essential to the selection and optimization of parameter values in predetermined topologies. They play such a key role in today's industry that being able to speed up the simulation process by even a small percentage can translate to significant reductions in research and development expenditure.

This report contains an examination of the possibility of reducing the cost of simulation by exploiting the relationship between low-order polynomial descriptions of circuit element models and circuit analysis techniques. In particular the major focus is time-domain transient analysis, the most computationally expensive aspect of circuit simulation today. A thorough examination of the applicability of reduced-order modelling (ROM) is presented for every level of the simulation hierarchy.

1.1 Formulating the Problem

The circuit is represented in the form of a net-list which provides a description of the interconnection of devices and input sources. Through application of Kirchoff's Current / Voltage Laws and the constitutive branch equations, a set of nonlinear first-order differential equations can be extracted. Hence, the nonlinear time domain circuit simulation problem can be reduced to finding the solution to the arbitrarily complex relations of the form :

$$f(x(t), x(t), u(t)) = 0, \qquad 0 \leq t \leq T \tag{1.1}$$

where $x(t) \in \Re^n$ is a vector of node voltages and branch currents at time t, $\dot{x}(t) \in \Re^n$ is a vector of time derivatives and $u(t) \in \Re^n$ is the vector of input sources at time t.

1.2 Solving the Problem - Existing Approaches

1.2.1 The Hierarchy of a Circuit Simulator

It is not possible to find closed-form solutions to the set of describing equations in the general case. Hence, a nested iterative approach is taken [18] as shown in Figure 1.1 .

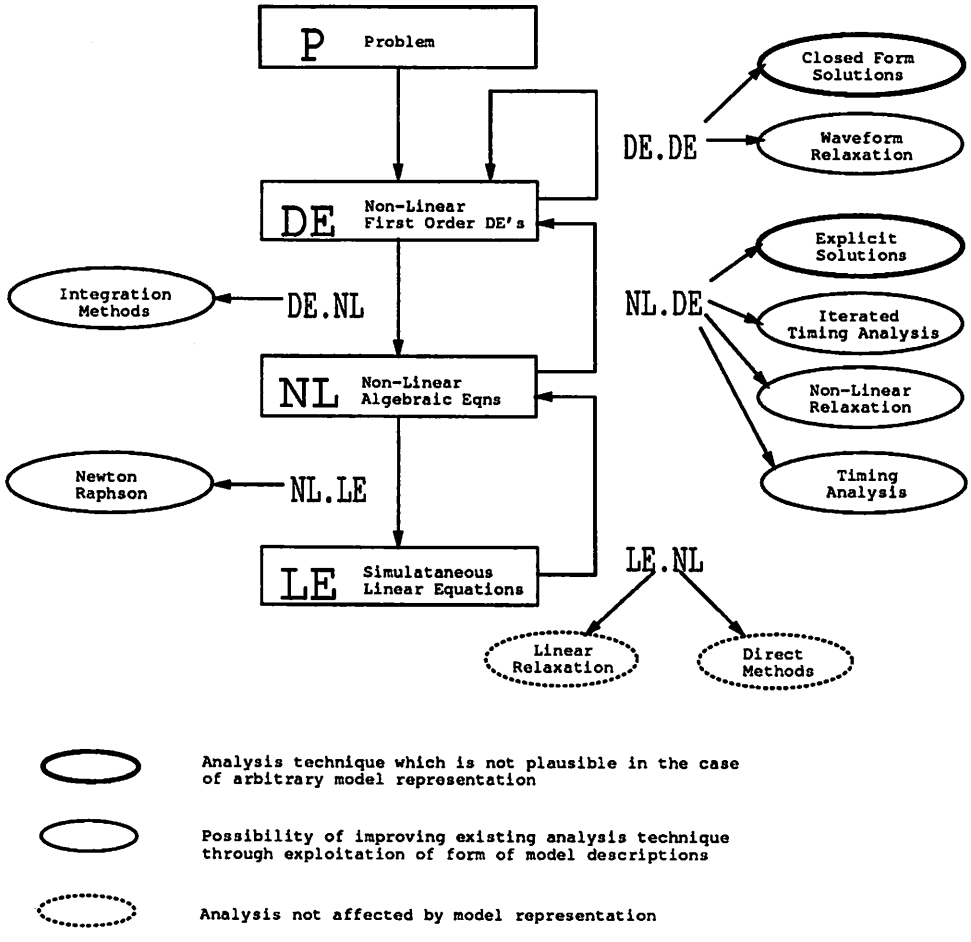


Figure 1.1: *Analysis Hierachy of a Circuit Simulator*

There are essentially two levels of reduction below the nonlinear, first-order differential equation (DE) level. To extract a set of nonlinear algebraic equations from the differential equations, integration methods are used, most commonly Linear Multistep (LMS) methods. In this way,

derivative-dependent elements such as capacitors are reduced to companion models depending only on the independent variables at past points and those being computed at the present time point. For example, SPICE defaults to the Trapezoidal Implicit Backwards Differentiation Formula (TR) [16] [12]:

$$x_{n+1} = 2\frac{v_{n+1} - v_n}{h_{n+1}} - x_n \tag{1.2}$$

For capacitors, the following relationship is obtained:

$$\Rightarrow i_{n+1} = \frac{2C}{h_{n+1}} v_{n+1} - \frac{2C}{h_{n+1}} v_n - i_n = G_{n+1} v_{n+1} + I_{C_{n+1}} \tag{1.3}$$

which generates the companion model as shown in Figure 1.2.

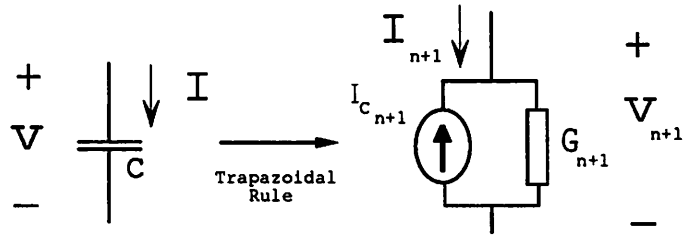


Figure 1.2: *Companion Model for Capacitors using TR*

Trapezoidal rule is used as it is the lowest-order A-stable method (stable only for stable systems) which experience has shown to give good results for practical circuits [16].

1.2.2 Direct Methods

Direct methods are a set techniques used to solve the nonlinear (NL) equation level problem. The nonlinear equations are reduced to the linear equation (LE) level which can be solved explicitly. Each solution thus obtained is taken as a point to reformulate a set of linear equations from the nonlinear ones thereby providing the basis for an iterative solution technique for the NL equations.

Reduction of a set of NL equations of the form:

$$f'(x(t_n), u(t_n)) = 0, \qquad t_0 \leq t_n \leq T \tag{1.4}$$

to a set of linear equations:

$$A^i . x_n^i = b^i, \qquad i^{th} \ \ iteration \tag{1.5}$$

is achieved by Newton-Raphson (NR) techniques. That is, linearization is achieved by finding all derivatives of the set of NL eequations at the initial guess or previous iteration position. This is the

same approach that is taken to solve for the dc operating conditions.
The formulation:

$$J(x_n^i).x_n^{i+1} = J(x_n^i).x_n^i - f(x_n^i) \tag{1.6}$$

where $J(x_n^i)$ is the Jacobian at iteration i, given by taking the derivative of Equation 1.4 with respect to all elements of the vector $x(t_n)$, is illustrated in Figure 1.3 for a simple two-dimensional example.

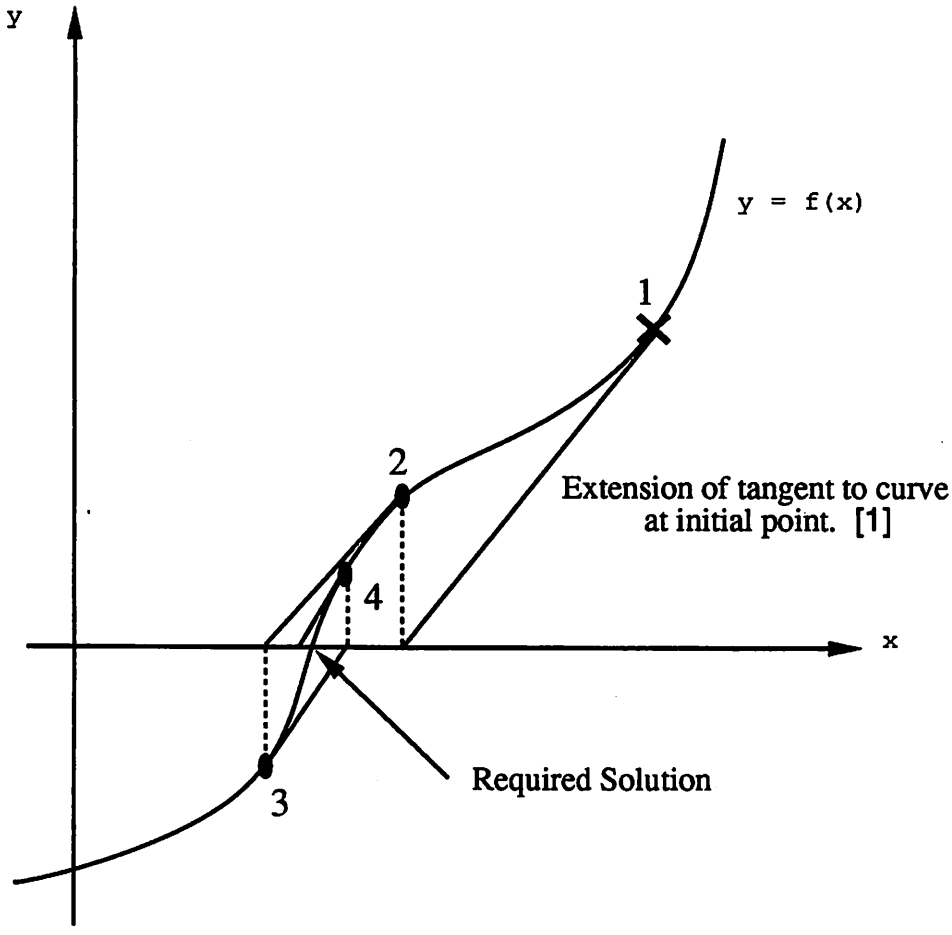


Figure 1.3: *Illustrating NR for a simple example*

In this example, Point 1 corresponds to the initial guess. The intersection of the tangent at this point with the line $y = 0$ gives a value of x_1 from which to begin the next iteration. At Point 2, corresponding to this value of x_1, the tangent is computed and the computation repeated as for Point 1. The procedure halts with answer x_i after the change in the solution between iterations, $|x_i - x_{i-1}|$, is less than some required error ϵ, or a specified upper bound on the number of iterations

is reached and the algorithm terminates with failure.

In practice, NR is generally damped by taking only a fraction of the step along the derivative vector, this fraction being determined by enforcing the reduction of some appropriate norm at each iteration. This implies that the path towards the solution along the nonlinearity is more closely followed. Although for the simple example above this may imply taking more steps to converge, in a higher dimensional space the better path approximation is generally found to reduce computation time [16].

The solution to Equation 1.6 is usually computed after LU decomposition or Gaussian Elimination on the Jacobian Matrix has been performed [18]. These techniques are used to exploit the sparsity of the Jacobian matrix.

1.2.3 Relaxation Methods

Relaxation methods are iterative approaches generally used as an alternative to the solution of a system of linear equations required for direct methods. Linear relaxation can, in fact, be used to solve the set of linear equations in place of solving exactly. However, compared with efficient sparse-matrix manipulation, this approach is often too complex to justify its use [13]. Direct methods are also inherently more reliable. Consequently, linear relaxation is not presented further here.

1.2.3.1 Relaxation at the Nonlinear Equation Level

At the nonlinear equation level, each iteration of a relaxation approach involves the solution of a set of decoupled NL equations. If exact solution of the NL equations was possible, the NL Gauss-Jacobi (GJ) and NL Gauss-Seidel (GS) relaxation methods could be expressed as:

Nonlinear Gauss-Jacobi:
repeat {
  forall (j **in** N) {
    solve $g_j(x_1^k, ..., x_j^{k+1}, ..., x_N^k) = 0$ for x_j^{k+1};
  }
}
until ($\|x^{k+1} - x^k\| \leq \epsilon$)

Nonlinear Gauss-Seidel:

repeat {
  forall (j **in** N) {
    solve $g_j(x_1^{k+1}, ..., x_j^{k+1}, ..., x_N^k) = 0$ for x_j^{k+1};
  }
}
until ($\|x^{k+1} - x^k\| \leq \epsilon$)

where the final statement concerns error bounds defining convergence. In fact, for a ball sufficiently small about the solution point, conditions of convergence parallel those of relaxation in the linear case [13]. That is, defining $g'(\hat{x})$ as the Jacobian of $g(\hat{x})$ where $g(\hat{x}) = 0$, and writing:

$$g'(\hat{x}) = L(\hat{x}) + D(\hat{x}) + U(\hat{x}) \tag{1.7}$$

where $L(\hat{x}), U(\hat{x}), D(\hat{x})$ are the strict lower, upper and diagonal matrices respectively, construct the matrices:

$$M_{GJ}(\hat{x}) = -D(\hat{x})^{-1}(L(\hat{x}) + U(\hat{x})) \tag{1.8}$$

and:

$$M_{GS}(\hat{x}) = -(D(\hat{x}) + L(\hat{x}))^{-1}U(\hat{x}) \tag{1.9}$$

There exists a region of convergence for the method if the eigenvalues of the respective matrices lie within the unit circle. The rate of convergence is linear, in comparison to quadratic for the direct methods approach described in Section 1.2.2. [15]

When computing x_i, GS based relaxation uses the present iteration approximation for all x_j s.t. $j < i$ whereas GJ uses the previous approximation for all x_j. GS consequently takes fewer iterations to convergence in most cases but does not lend itself to parallel evaluation as effectively as the GJ methods [13].

Iterated Timing Analysis (NL) The concept of iterated timing analysis (ITA) relies upon the fact that the convergence properties described above and moreover the rate of convergence is not affected by using a single step of Newton Raphson in place of solving the inner computation loop exactly [15]. This vastly decreases required computation at each iteration, thereby improving feasibility. This technique is exploited in the SPLICE program [14].

Through the construction of a *signal-flow graph*, required computation is further reduced by decoupling the time-step interdependence. Changing nodes can schedule the relevant fanouts

for a future time-step rather than processing unnecessarily large blocks or the entire circuit at every time-step.

Implementation of this technique has exhibited speedups of 10-200 times over conventional analog circuit simulators for large digital circuits [13].

One-Shot Timing Analysis (NL) Instead of iterating on the relaxation loop, in one-shot timing analysis a single pass is made with the inner NR loop taken to convergence. Considering the descriptions of the relaxation techniques given previously, this effectively implies that the property: $\| x^{k+1} - x^k \| \leq \epsilon$ is not guaranteed to hold. As a result, the convergence and error properties of the LMS method used to linearize the capacitors are not guaranteed. Most of these methods can be shown to be non self-consistent [13].

Feedback loops introduce one step of timing error at each time-point as the relaxation step is not iterated. To ensure that only feedback loop configurations induce timing error, efficient use of the signal-flow graph becomes critical in determining equation processing order. For this method the signal-flow graph is not just a tool for improving simulation time as for ITA. Also, the Gauss-Seidel approach only considers the influence of the upper triangular portion of the Jacobian matrix, the Gauss-Jacobi considering only the diagonal. If the technique is not self-consistant, accuracy of simulation cannot be improved by reducing the time-step. Consequently, for such methods circuits with either tight feedback coupling or floating capacitors are not handled well.

The only known self-consistant one-shot timing analysis technique is the *Implicit-Implicit-Explicit* method [13]. Although it has been demonstrated useful in circuits with floating capacitors, ITA has become the more practical choice [14].

One-shot timing analysis can obtain speedups of at least one order of magnitude over iterated timing analysis in circuits where it is applicable.

1.2.3.2 Relaxation at the Differential Equation level

Waveform Relaxation (DE) Waveform Relaxation (WR) at the DE level deals with variables in function spaces [21]. Intuitively, an equation in the set of as yet unsolved DE's is solved over all time relative to the existing approximate solution over all time for variables on which it depends. In an obvious extension of the NL relaxation case, GS-WR relies on the new variable solutions already equated in the present iteration whereas GJ-WR does not. The iterations continue until all of the computed waveforms change by less than a threshold amount as determined by the chosen

metric. The choice of time-step for solving other equations does not influence the present solution as each equation is considered separately over time. This decoupling is the major advantage of this approach.

In an actual implementation, blocks of the circuit (a subset of the equations, rather than a single one) are considered simultaneously using a direct methods approach. Scheduling of the blocks is used to improve simulation time. Similar to the use of signal-flow graphs in ITA, scheduling is an algorithmic improvement of WR and not a requirement for accuracy.

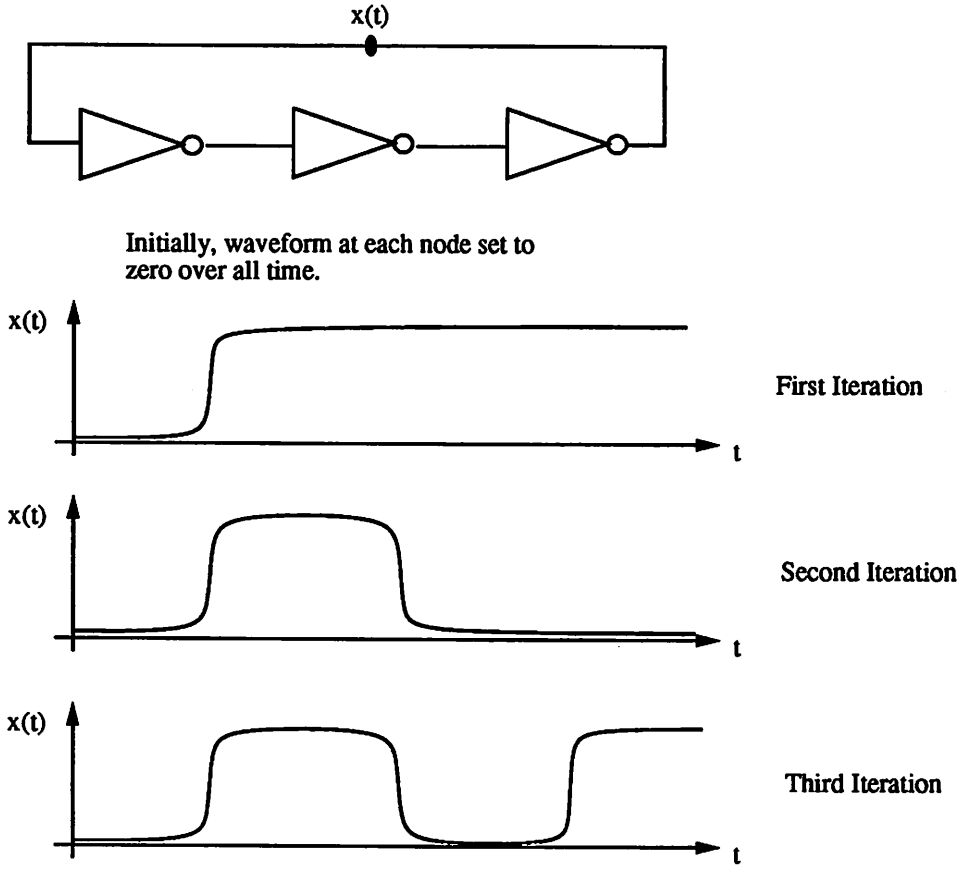


Figure 1.4: *Waveform Relaxation applied to Ring Oscillator*

As WR analyzes every equation over all time at each iteration, the technique can become wasteful in cases of strong feedback. Consider the example of a ring oscillator depicted in Figure 1.4. To reduce overcomputation, the time interval can be windowed into sections, each of which is iterated to convergence before progressing to the next interval.

With these algorithmic practicalities, implementation of WR in RELAX has shown speed

up of an order of magnitude over conventional simulators for MOS digital circuits.

1.3 Speeding Up the Analysis

1.3.1 At The Problem Level

1.3.1.1 Restricting the Topological / Model Generality

The design of circuit-specific simulators is an important facet of the simulation problem. For example, it is generally overkill to use a powerful analog circuit simulator, such as SPICE, on digital circuits with little analog behaviour (little or no capacitive signal coupling). The questions to be answered by the results, such as correct logic functionality and delay times, do not demand the high accuracy of waveforms achieved using analog simulators. Moreover, such a simulator takes an excessively long time to analyse a digital circuit as every transition from a 0 to a 1, or vice-versa, requires very small time-steps.

In digital circuit analysis, device models can be much cruder than the approximate models for analog circuits. This flexibility lends itself to efficient representation of models using low-order polynomial sections (Ref. Section 1.3.3.2). Circuit-specific simulators such as CAzM [7] which uses third-order spline modelling, have shown significant speedup over SPICE for digital MOS circuits.

Computational reduction can be also achieved if the circuit can be shown to be a series-parallel topology. For example, without the presence of feedback loops, a signal-transition graph can give an equation processing order such that no timing errors are incurred if one-shot timing analysis is used.

Although such speedup techniques have value in their area of application, this project is concerned with examining techniques valuable to general purpose simulators. Reduction of the circuit complexity will only be used as a tool for demonstrating that certain approaches do not reduce the analysis time of even a simplified problem. This serves to illustrate their inapplicability to the general simulation problem.

1.3.2 At The Differential Equation Level

1.3.2.1 Time Step Improvement

There has been a thorough examination of the problem of using higher-order integration methods to increase the length of the time-step [12][16]. It was found that the extra computation required at every time-point outweighed the time-step improvement. Moreover, if a method such as Variable-Order Gear was used, a large portion of the computation time became devoted to determining the time-step size and the required method-order optimally. As a result, the Trapezoidal method was found to be the most efficient and is the LMS method presently used in SPICE.

This approach has been examined and found barren from the point of view of speedup potential. The representation of the models using reduced-order polynomial sections does not bring this approach back into consideration as the time variation of a DE solution remains quite arbitrary (Ref. Section 3.2.1). Consequently, time-step improvement is not a major focus of this investigation.

1.3.3 At The Non-Linear Equation Level

1.3.2.2 Exploiting Latency

Speedup can be achieved by sectioning the circuit into minimally interconnected partitions, then assessing which of these need be processed at each time-step. In much the same way as a computer determines which sections of code to shift in and out of its cache memory, a block of the circuit can made *latent* until it is 'called' for processing because one of its controlling inputs is changing. This speedup technique is not affected by model representation so is not considered.

1.3.2.3 Models of Known Low Order Representation

Instead of having a set of computationally complex equations or empirically determined data points, models can be represented as a set of low-order polynomials each of which apply over only a portion of the device characteristics. Piecewise-constant [20], linear (Katzenelson) [5] and cubic simulators [7] have all been implemented, an example of each of these modelling schemes for a diode-like characteristic is shown in Figure 1.5.

As the order is decreased, the region of applicability of each polynomial section decreases but so too does the computation at each iteration. In each implementation, these modelling schemes have demonstrated speedup over SPICE for MOS digital circuits.

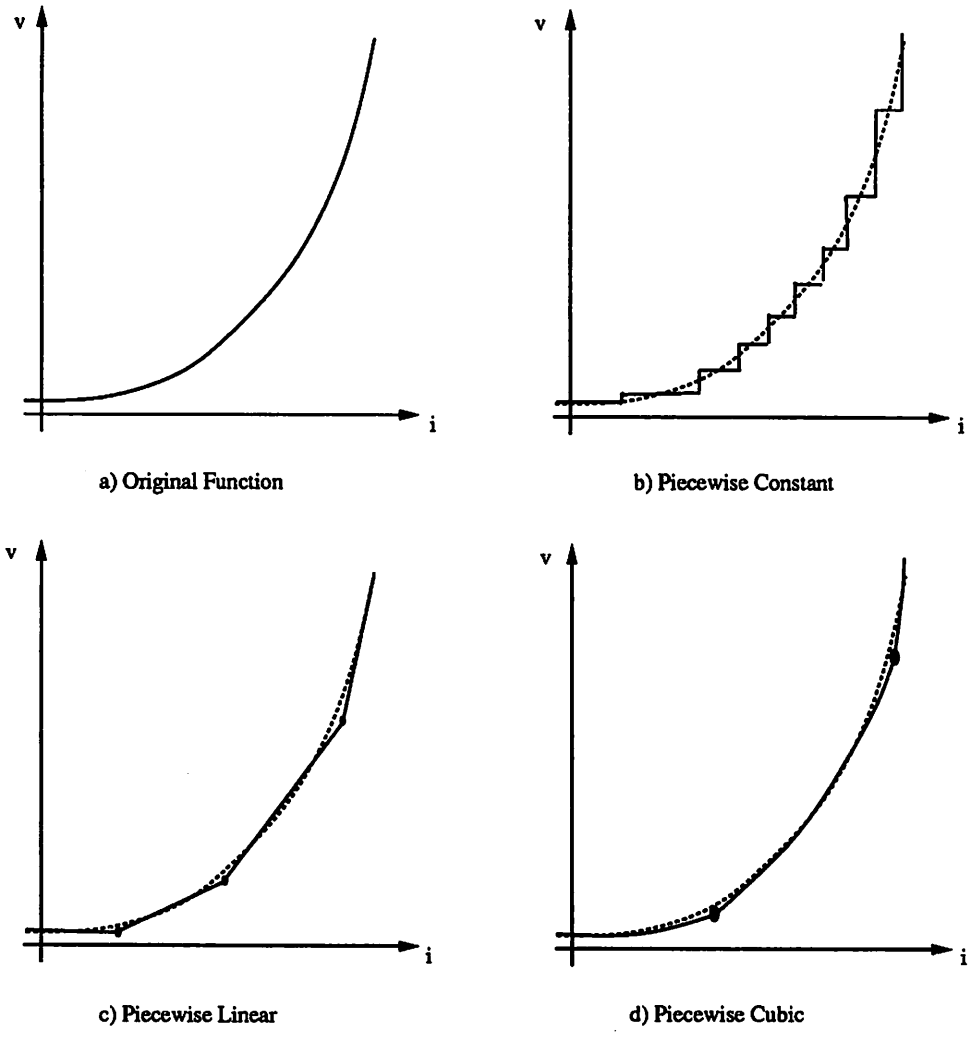


Figure 1.5: *Piecewise-Polynomial Modelling Schemes*

Both the piecewise-constant and linear schemes have analysis techniques which require stepping from region to region until a solution is found (Ref. Section 1.4.1.). Consequently, if the model must be very accurate and regions are small, the simulator can become slow. This is the reason for their applicability to digital circuit analysis where modelling errors are not so tightly bound.

On the other hand, circuit analysis techniques for conventional models can be applied to the higher-order descriptions. Indeed, the third-order spline-fitted model approach of CAzM [7] achieves its speedup over SPICE by evaluating the Jacobian matrix more rapidly. It is examining the possibility of simplification at higher levels of the simulator hierarchy due to this modelling

technique which forms the basis of this report.

Ohm's law does not hold for the piecewise-constant modelling as the description is not continuous. Consequently, as higher-order schemes cannot easily be shown as an extension of this approach, it will not be examined further in this report. Certain properties of piecewise-linear simulator techniques are very desirable so an examination has been made as to whether the technique can be generalized to higher-order modelling. To this end, the the Kazenelson algorithm is presented in more detail later in this introduction.

1.3.4 At The Linear Equation Level

1.3.4.1 Better Sparse Matrix Manipulation

In the direct methods approach, computation time can be divided into derivation and solution phases. Provided that the average degree of the nodes in the connectivity graph of the circuit does not increase (in practice, the average degree has been found to be about 2.3 [14]), the time required to set up the linear equation form is proportional to the number of nodes. However, using sparse matrix techniques, the solution phase increases as approximately $n^{1.2}$, where n is the node count [16].

To be satisfied that the search for the faster simulator should not just concentrate on the improvement of sparse matrix manipulation, it is necessary to show that the solution phase does not dominate computation time for reasonable sized circuits. The plots of Figure 1.6 and Figure 1.7 provide the required justification.

The graphs of Figure 1.6 were generated from repetitive block networks with two connections per node for the ladder network, three for the delta (where the repititive block is a triangle). The average is therefore close to that expected of practical circuits. Both resistive and MOS pass gate circuits are shown in the results which illustrate the difference in run-times between an old [16] and state-of-the-art sparse matrix package [11] in SPICE. Note that with the better sparse matrix manipulation, the dramatic run-time reduction is not noticeable over all circuit sizes. In fact, over the range from 10 to 100 nodes, there is little benefit despite the obvious improvement for circuits of more than 1000 nodes. This suggests that the linear solution phase of the analysis is not dominant for small circuits. In fact, with the better matrix package, solution cost only reaches about 50% of the setup cost at circuit sizes of 10^4 nodes. In the range $1 \rightarrow 10^4$, one percent speedup in the derivation phase therefore results in at least 0.67 percent overall speedup. i.e. For any circuits in this size range, improvements in the speed of execution for the derivation phase translate to

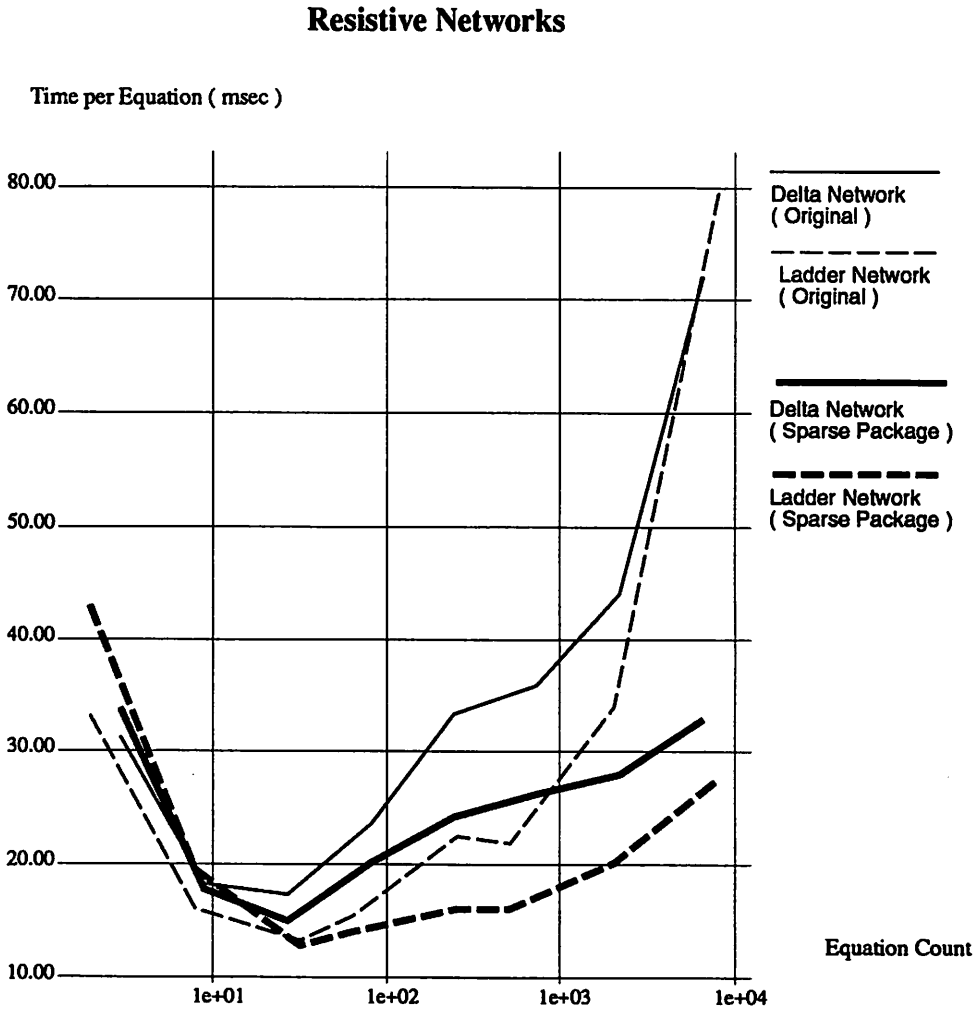


Figure 1.6: *Computation Time for Direct Methods Approach - Resistive Network*

significant savings in the overall run-time of the simulator. The aim of this research is to examine the possibility for reduction of the complexity of this part of the simulation problem through the use of low-order-polynomial model descriptions.

1.4 The Reduced-Order Model

Although this search for more efficient solutions to the circuit simulation problem is concerned with exploiting the properties of low-order representations for circuit models, it does not focus on the derivation and representation of such models. Consequently, the following is a very brief outline of techniques used in existing reduced-order polynomial simulators. This also serves

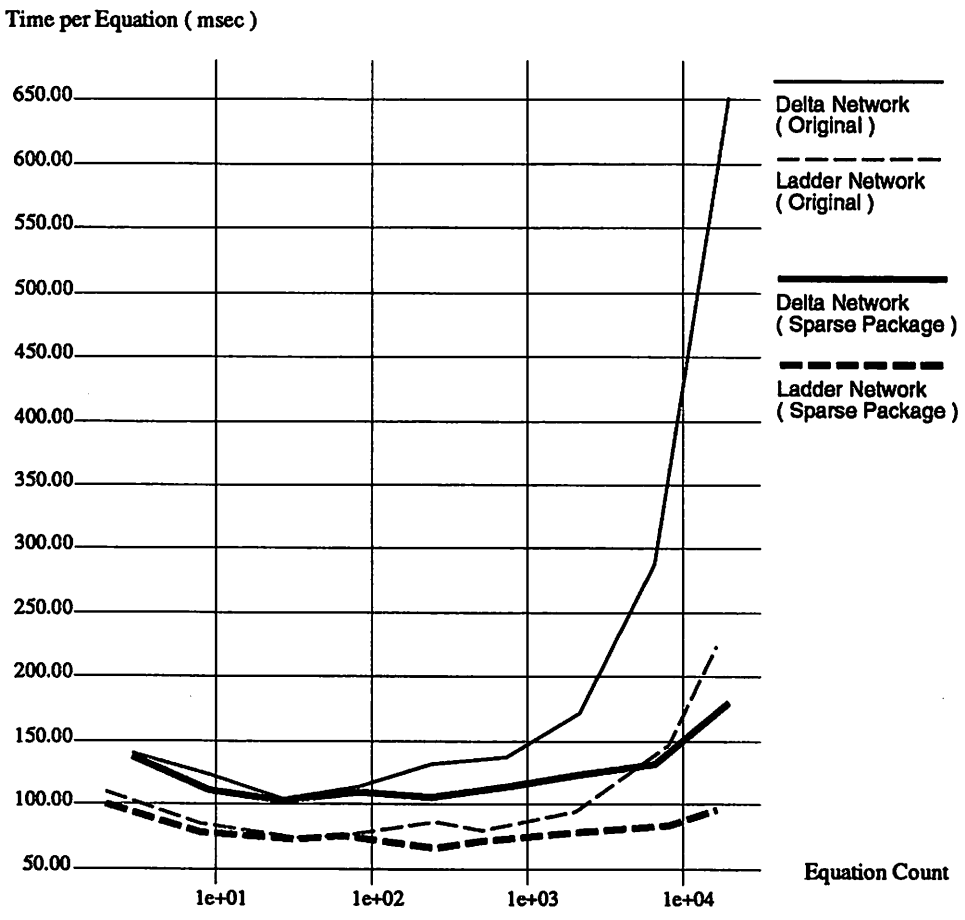


Figure 1.7: *Computation Time for Direct Methods Approach - MOS Pass Gate Network*

as an illustration of the basic properties of these representations.

1.4.1 The Katzenelson Algorithm

The Katzenelson algorithm [10] is a method for solving a circuit when the device models are represented in piecewise-linear form (Ref. Figure 1.5). Each linear polynomial section only applies over a restricted range as determined by required model accuracy. Consequently, for the overall network, the relation domain is partitioned into a set of polytopes (*regions*) within each of which the circuit is described by a set of linear equations. Basing an iterative approach upon the solution of these equations is equivalent to applying the NR technique. Unfortunately, as the model descriptions are not first-derivative-continuous, there is no guarantee of convergence.

In the Katzenelson approach, a set of linear equations are solved to give a point x^*. The set of linear equations are determined by the region in which the present iteration solution, x^k, lies. If x^* lies in the same region, then a solution has been found. Otherwise, the intersection of the line from x^k to x^* with the present region boundary is taken as x^{k+1}. The fundamental premise upon which this approach is based is that the path described in the space X is the inverse image of the straight line along vector $f(x^k) - y_0$, where $f(x^*) = f(x_0) = y_0$ at the point of solution. At each boundary-crossing, the progression towards or away from the desired solution $f(x^*)$ is uniquely determined at the next step by the present position provided that it is not the corner point of a polytope.

The efficiency of the algorithm is based upon the ability to determine easily these intersection points and not having to completely reconstruct the set of linear describing equations. To do this, two important points have to be noted:

- The domain-space for a circuit as described by the set of sparse tableau equations has a lattice structure. That is, region boundaries are orthogonal and parallel to axes-planes.
- The mapping must be continuous on every boundary.

The first fact implies that unless the vector passes through a corner point of the region, intersection with the boundary can be detected by tracing along the direction:

$$x' = x^k + \beta(x^* - x^k) \qquad 0 < \beta < 1 \tag{1.10}$$

and stopping when the first element of x' violates a inequality constraint for the region.

The second point is relevant to determining the modification to the Jacobian matrix required at each boundary crossing. Consider describing the polytope regions as the set of inequalities in x:

$$K = \{x | C_i x + g_i \geq 0\} \tag{1.11}$$

where C_i is a matrix, g_i is a vector and the linear mapping in each polytope is described as:

$$y = A_k x + f_k \tag{1.12}$$

where A_k is a matrix and f_k is a vector. Now, as every mapping must be continuous, at every boundary where $c_{kj}x + g_{kj} = 0$, between regions k and j, the following must hold:

$$A_k x + f_k = A_j x + f_j \tag{1.13}$$

Hence, there is only a rank-one update for A_k at every boundary crossing provided that the intersection is not with a corner point of the region.

Moreover, it has been shown [4] that given:

- All A_k are non singular
- The solution curve hits no corners
- The determinant of A_k in every unbounded region is of the same sign and the solution path starts from a point in an unbounded region such that $\| y_0 \| > M$ where M is the largest attainable value of $\| y \|$ in the bounded regions
- There does not exist multiple solutions in any individual polytope

then:

- No region except for the starting region can be reentered.
- Each bounded region entered by the solution curve must also be exited unless a solution x_0 exists in that region.
- A solution x_0 will be found if one exists.

There have been examinations of how to improve the algorithm to handle the analysis if the listed conditions do not hold [4]. Furthermore, techniques for computation of the boundary crossings have been generalized for the case where the polytopes are not guaranteed to form a lattice structure [5]. This allows application of the Katzenelson approach to more compact circuit equation representations, such as modified nodal analysis. However, these results will not be presented in detail here as the modifications do not change the basic premise of the approach and consequently are not of fundamental importance to the following study.

As the Katzenelson approach finds the solution by stepping from region to region, the average size of the polygons relates directly to run-time. That is, a smooth trade off between model accuracy and simulator speed can be achieved by varying the size of the piecewise linear segments. Similarly, there exists a tradeoff between memory requirements and precision. However, the critical point to note about the method is that it only becomes viable because there is a well-defined incremental change between the linear equation description when moving from one region to an adjacent one. If the equations had to be completely reformulated at each boundary crossing, the accuracy which could be obtained for run-times competitive with SPICE-like simulators would be far from sufficient.

1.4.2 The Low-Order Polynomial Model

If the interpolation between data points is a polynomial of order greater than or equal to two, the model description can be made first-derivative-continuous everywhere. Convergence properties of the standard NR type approaches are consequently maintained as Lipshitz continuity is satisfied. The derivative with respect to any of the dependent variables can be computed efficiently directly from the known polynomial coefficients and variable values at the present iteration. With the third-order modelling scheme presently used in simulators such as CAzM [7], this is the only property of the representation which is exploited.

If the polynomial description is of odd order, then equal weighting is given to all end point conditions. That is, the same amount of extra information is required at all end points. This is not so for even-order modelling. Consider, for example, the parabola between two points - the points plus the derivative at only one of them is required to fully specify the function. Odd-order polynomial modelling is therefore usually favoured.

Increasing the order of the model representation generally decreases the number of regions required for a description within the same error bounds. Consequently, although derivative computation becomes more expensive, one would expect region boundaries are traversed less frequently on the average in an iterative procedure such as NR. This implies less calls to the stored models to obtain new polynomial coefficients are required. However, much of the investigation into the feasibility of reduced-order-model simulators has concentrated on third-order interpolation, such as that obtained using multi-dimensional splines. This is a consequence of the third-order polynomials being the highest-order polynomials which can be solved explicitly in reasonable time for an iterative algorithm.

1.5 Report Organization

The examination of applicability of polynomial modelling to analysis techniques is presented in detail in the following two chapters. The first of these chapters investigates approaches to solving the problem at the NL equation level, the second investigating the DE description. Organization of these chapters is correlated to the simulator hierarchy detailed in Figure 1.1.

More concisely, the possibility of an improved technique for extraction of linear equations, theoretical and practical arguments against generalization of the Katzenelson algorithm to polynomial models other than linear, and the practicality of global explicit solutions or explicit

solutions in relaxation techniques are examined in Chapter 2. In Chapter 3, NL circuit relations are proven to be of the same order as device models even in the case of nonlinear capacitors. It is shown that the model descriptions cannot provide improved estimates on LMS method accuracy and stability, nor allow time-steps to be computed for solution to the next time-point to remain in the present polytope. Closed-form solution techniques are proven impractical and the application to WR is demonstrated to be of limited potential.

Each chapter closes with a summary of the applicability of reduced-order modelling at that level of analysis. Final conclusions on the predicted usefulness of this modelling technique to the speed-up of circuit simulators is presented in Chapter 4.

Chapter 2

The Nonlinear Equation Level (NL)

2.1 Formulating a Linear System of Equations (NL.LE)

At the Nonlinear Equation Level, the problem to be solved is that of finding a point in $\Re^n$ which satisfies a set of relations where those relations are of known polynomial order in hyperplane-bounded regions in the space. It is shown in Section 3.1.1 that the equations describing any circuit will possess these required properties even if nonlinear capacitors are present.

2.1.1 Application to Newton Raphson techniques

Existing reduced-order model (ROM) simulators exploit the ability to reconstruct the Jacobian matrix quickly. However, this generally involves extracting the entire set of applicable polynomial descriptions at every iteration from the model database. It will be shown that by using the intersection of the NR vector with polytope boundaries the requirement for rapid memory access can be reduced.

Techniques for finding the intersection point of the solution path with the polytope boundaries in the Katzenelson algorithm may be used explicitly to find the intersection of the NR vector with bounding planes. In the piecewise-linear approach the update to the Jacobian matrix is rank one provided that the solution path, linear in each polytope, does not intersect a corner of the polytope. By analogy, if the NR vector passes through a single bounding plane, only one equation in the set of polynomial branch equations need be modified. That is, crossing a bounding hyperplane corresponds to violating a single regional constraint for an individual device given the present polynomial descriptions for the entire circuit. Only the equations relating to that branch voltage /

current need be updated.

One approach which exploits this property would be to stop at boundary crossings of the NR vector, update the nonlinear equation description and begin the next iteration there. Unlike the piecewise-linear approach, this would require the re-evaluation of the Jacobian at this point and computation of the actual value of the function at the crossing (Ref. Figure 1.5). However, there are major theoretical drawbacks to such a technique:

1. The analogy with the Katzenelson theorem is not complete (Ref. Section 2.1.2), so this approach can not guarantee the properties inherent in that method.

2. Although at first glance this may appear to be a form of damped NR, the step size is fixed by model descriptions and the topological arrangement of the circuit, not by enforcing the reduction of some suitable norm. This implies that even if damping is applied to determine an upper bound on the step length, rate of convergence properties of NR are not retained.

3. If the NR vector intersects a corner point, the algorithm is not guaranteed to cross all the hyperplanes which form that corner (Figure 2.1.). Consequently, to update the set of nonlinear equations, the NR vector at the boundary point must be examined to find which region is about to be entered.

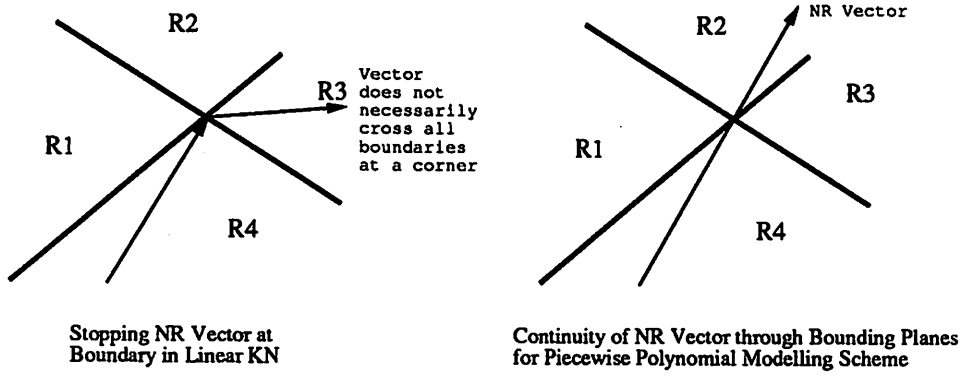


Figure 2.1: *The Corner-Point Problem*

Instead of stopping at the boundary points, the convergence conditions of Newton Raphson can be maintained by taking the same step as determined for arbitrary model descriptions. However the polytope boundaries crossed by the vector can be determined by using the same techniques as exploited in the Katzenelson algorithm for intersections of vectors with hyperplanes. Once these have been found, the explicit relation of these planes to individual device models can be

exploited to update the set of nonlinear equations efficiently. Corners are no longer a problem as the continuation of the NR vector is guaranteed to pass through all the planes which intersect at that point, as illustrated in Figure 2.1. If the vector actually terminates on a plane or corner, the set of equations derived for the present polytope (without considering the final bounding plane intersection) are valid as the device models are derivative continuous. The critical test as to whether this method is practical is obviously the comparison between the time to completely reconstruct the Jacobian and that to compute the bounding plane intersections and update selectively.

2.1.2 Generalizing the Katzenelson Algorithm

The Katzenelson algorithm has found its niche in some existing simulators for the efficient computation of dc operating point under simplified model conditions. In particular, it has been found useful on circuits for which standard Newton Raphson techniques don't converge in reasonable time, if at all. As the Katzenelson algorithm utilizes circuit models based on piecewise-linear representation, it is appropriate to examine whether some of the highly desirable properties of the method can be maintained for a piecewise-polynomial modelling scheme.

The following analysis of the applicability of such an approach is presented in two parts. An examination of whether the Katzenelson approach for the piecewise-linear case has desirable properties for transient computation constitutes the first part. In the second subsection the difficulty of constructing an efficient theoretical parallel between piecewise-linear and piecewise-polynomial approaches is presented. It is consequently demonstrated that a generalization of the Katzenelson algorithm probably has little practical applicability even to finding dc operating points, let alone speeding up transient analysis.

Application of Standard Katzenelson to Transient Computation

The most desirable property of the Katzenelson algorithm is its well-publicised ability to find solutions in cases where more conventional numerical techniques fail [4] [5]. However, it is important to note a proof that the algorithm will find a solution (indeed, all solutions) relies strongly upon the Jacobian determinants in the infinite regions being all of the same sign [4]. Futhermore, the starting point of a path to the solution must be within an infinite region. Allowing arbitrary models, the path can be quite circuitous demonstrated by the example in Figure 2.2. By modifying the starting position to be within a totally bounded region (region bounded in all dimensions), it is straightforward to construct a non-convergent path for this example. The algorithm can also be

made to fail by denying the Jacobian determinant sign rule for the infinite regions. Consequently, neither condition can be relaxed if the guarantee of finding any existing solution is to be maintained.

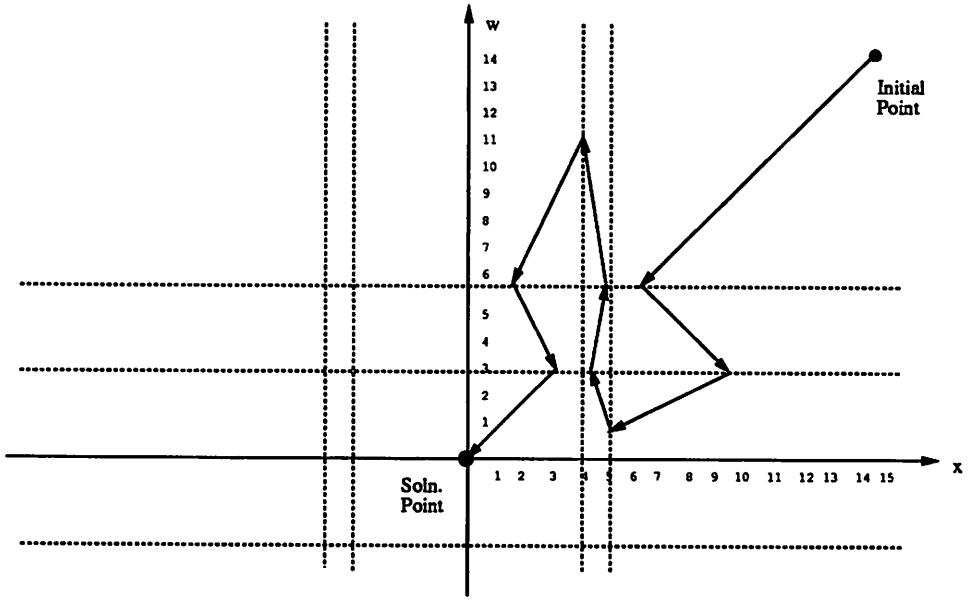


Figure 2.2: *Example Illustrating Possible Circuitous Nature of the Katzenelson Algorithm*

The "guaranteed solution" property of the Katzenelson Algorithm is of paramount importance in any constructive argument for its use. If this property is not true for each time-point in the transient analysis, there is no theoretical justification for simulation with piecewise-linear models having better convergence properties than simulation with standard models and NR techniques.

For transient analysis, maintaining the guaranteed solution property by starting from an unbounded region of the full circuit description at every time-point is clearly impractical. Each step of the Katzenelson algorithm is determined by region boundaries and backtracking away from a desired solution is allowed, so a rate of convergence is not definable. Consequently, the termination condition cannot be based upon a rate of convergence and must be related explicitly to the number of regions examined before finding a solution. To make this number practical, a restricted subspace must be isolated which contains the solution to the next time-point. (Adjacent regions outside the restricted subspace can be assumed infinite and numerically modified provided continuity is maintained.)

Construction of a "predicted solution subspace" is an extremely difficult task, as presented in Section 2.2.2. In fact, its size cannot be computed nor even accurately estimated in sufficient time for transient analysis purposes. The size of the subspace must consequently be an educated guess, perhaps using a predictor extrapolation based on a previous time-step and "distance-traversed-in-space" relationship. Strong overestimates would only result in very large execution times for the Katzenelson algorithm at each time-step.

A major problem presented by this approach is whether the functions in the adjacent regions outside the subspace can be manipulated in such a way that the infinite-region Jacobian determinant property is satisfied for the subspace. It has been proven in Appendix A that for a lattice structure solution space, continuity of the function is maintained if and only if in every region infinite in the same direction of dimension x_j, $\frac{\partial f_i}{\partial x_j}$ is the same. Consider setting the derivatives in the infinite regions in order to satisfy the Jacobian property. Obviously, the singly infinite regions can be analysed first as only a single column of the matrix is free to be altered. The required property of the singly infinite regions in each dimension could be satisfied independently. However, by the continuity argument, assigning derivatives for just these regions fixes the derivatives for the entire space. For example, consider the region infinite in all dimensions (known to exist because of the lattice structure). The entire Jacobian matrix in this region is a construct of the assignments given to derivatives in the singly infinite regions. The sign of the Jacobian determinant in this region need not be the desired one as the required derivative computation in each dimension has been uncorrelated. Consequently, the infinite dimensions cannot be considered independently which makes this construction impractical for transient analysis. From an extension of this argument, it is clearly not possible to fix the infinite region derivatives progressively as the respective regions are entered. In the case of a non-lattice structure, attempting to satisfy the infinite region Jacobian determinant sign property is even less practical.

The above presentation is true for a set of piecewise-linear continuous functions. The fact that these functions are generated from a set of circuit relations and piecewise-linear device models does not reduce the complexity of the problem if an arbitrary interconnection of components is allowed.

Generalization to Piecewise-Polynomial Models

A fully parallel development of a Katzenelson type algorithm for general piecewise-polynomial device models is beyond the scope of this investigation. However, a reasonably cursory

analysis of the problem is sufficient to show that even if such theory were developed it would be of little practical use.

In the piecewise-linear case, if a polytope with singular Jacobians or multiple solutions does not exist, the path generated by the Katzenelson algorithm can not reenter any region except that in which it started. This property is used to determine a suitable run time given the number of polytopes in the space and to detect cycling. In a polynomial modelling scheme of greater than first-order, it is not possible to ensure that less than one solution exists per region by model construction. Furthermore, in a polytope the Jacobian determinant can assume both positive and negative sign. Examples to illustrate these points are shown in Figure 2.3.

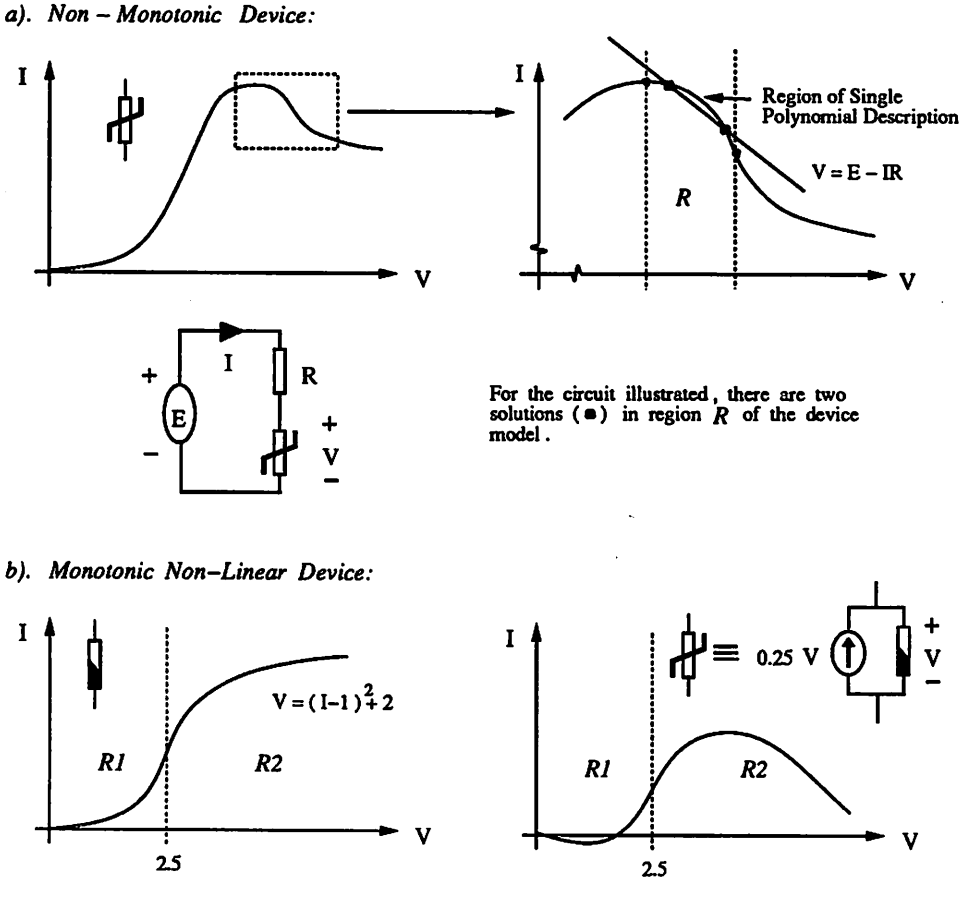


Figure 2.3: *Violation of the Single-Solution-per-Region Property*

In the partitioning scheme chosen for this example, the functional description of the model

is monotonic and first derivative monotonic in each region. Violation of the single solution per region property in this case implies that such a violation is possible for any chosen partitioning scheme (it is not practical to assign model regions dynamically according to device interconnection in a circuit). If device models are not overall monotonic, construction of a counter-example is trivial (first example). However, even in the case of a fully monotonic model, if arbitrary controlled sources are permitted the property can be denied (second example).

The existence of polytopes in which the sign of the Jacobian determinant is not unique allows the solution path to have multiple entry points into that region. It is extremely time-consuming to examine whether a region actually has this property (Ref. Section 2.2.1). Consequently, detection of cycling and determination of run-time based on number of regions in the space is not straightforward as for the piecewise-linear case. As demonstrated in the example, if circuits were designed to avoid this phenomena, it would require the exclusion of arbitrarily connected controlled sources and non-monotonic device models. Circuit restrictions of this form help reduce the problem to one which is well-conditioned even for standard NR analysis.

The uniqueness of the solution path (a mapping of a linear function between previous and present time-point conditions) is ensured as long as the Jacobian at any point is non-singular. This is a basic result of bifurcation theory. In the piecewise-linear case, if the Jacobian is non-zero in all polytopes, only crossings of the region boundaries can result in splitting the path. Furthermore, the continuation is known to be unique if the path intersects a single plane face, even if the sign of the determinant changes, as a region cannot be reentered. It is consequently only possible that a path splits at the intersection of that path with a polytope corner point. The perturbation theory used in these cases to determine the direction of extension is greatly simplified as the analysis need only extend to examining each of the regions which share that corner. However,singularity points for the Jacobian can exist inside polytopes through allowing piecewise-polynomial models. As the determinant is continuous across hyperplanes, polytope boundaries are not explicitly related to determining branching points of the solution path. In fact, fully generalized perturbation theory must be used at any point of Jacobian singularity. This is certainly a significant computational complexity increase over the piecewise-linear case.

Even if the Katzenelson algorithm could be generalized, its theoretical basis would rely upon accurately tracing the solution path. This path is not explicitly computable (Ref. Section 2.2). This implies an iterative approach such as NR. To get an accurate path trace, NR could be used with a very small increment along the linear path between last time-point and present time-point conditions. This, however, defeats the purpose of the method as this approach is not dissimilar

to using NR directly with very small time-steps. Alternatively, one might traverse a single NR vector to its intersection with a polytope boundary then attempt to find the correct intersection of the solution path with this plane. This approach, however, is not valid for the following reasons:

- The region boundaries are not necessarily the critical points for path splitting.
- Singularity points of the Jacobian could be overlooked which would invalidate the theory on which the algorithm is based.
- The plane of intersection of the NR vector and the solution path need not be correlated (Ref. Figure 2.4).

Consequently, the implementation of a generalized form of the Katzenelson theorem for piecewise-polynomial models could not be competitive with existing simulators.

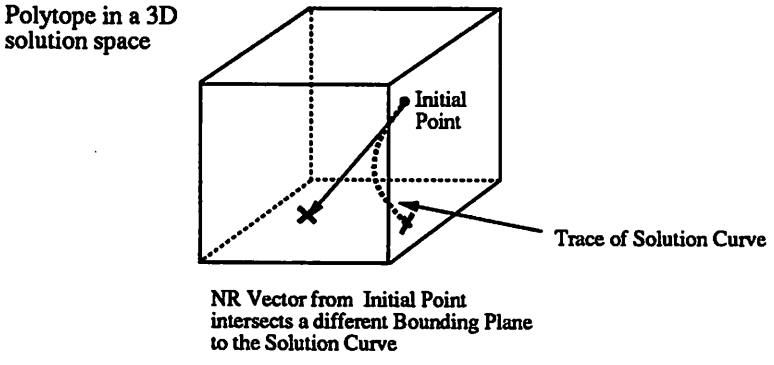


Figure 2.4: *Using NR to find Plane of Intersection with Solution Path*

2.2 Solving Without the Need For Global Linearization (NL.DE)

Consider device models as represented by piecewise-low-order-polynomials of less than third-order such that for the single-variable device, roots are explicitly computable in reasonable time. The question then to be asked is whether this property can be exploited to avoid the Newton Raphson linearization step. In the following analysis, the undesirably high complexity of such a scheme is exposed for the even the restricted case of all circuit elements being two-terminal devices.

It is easily demonstrated that for a set of polynomial equations extracted from a circuit topology, the ability to solve them simultaneously is a property which depends upon linearity. Take the example of Figure 2.5. The set of equations to describe this arrangement are the following:

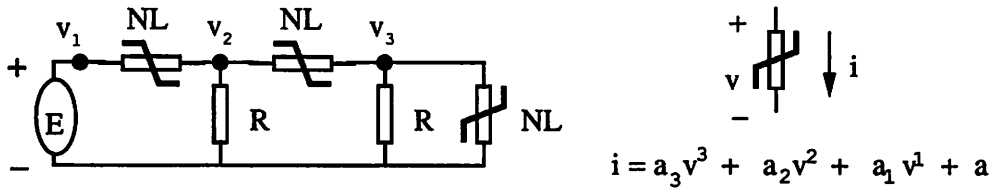


Figure 2.5: *Simple Two Terminal Device Nonlinear Circuit*

$$a_3(v_1-v_2)^3+a_2(v_1-v_2)^2+a_1(v_1-v_2)^1-a_3(v_2-v_3)^3-a_2(v_2-v_3)^2-a_1(v_2-v_3)^1-\frac{v_2}{R}=0 \quad (2.1)$$

$$a_3(v_2-v_3)^3+a_2(v_2-v_3)^2+a_1(v_2-v_3)^1-a_3v_3^3-a_2v_3^2-a_1v_3^1-\frac{v_3}{R}=0 \quad (2.2)$$

Given the source voltage E, find v_2, v_3. Even if the branch voltages were taken as variables to avoid multiplicative terms such as $v_2^2v_3$, a convenient linear algebraic representation does not exist for this case [8]. The only possible form is that in which the variable vector contains every branch voltage raised to powers 1, 2, and 3 as separate variables. The resulting removal of variable independence prevents explicit solution of this system by any conventional means. Consequently, in general an explicit solution for a node in a network can only be obtained after a series of circuit collapsing steps (Section 2.2.1). The only other approach which avoids the requirement for global linearization is one based upon relaxation techniques (Section 2.2.2).

2.2.1 The Explicit Solution

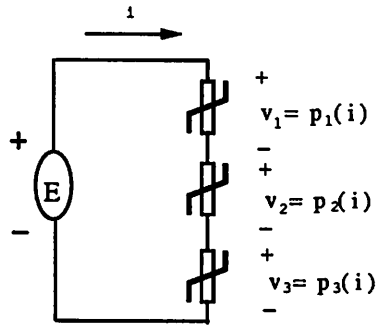
The incentive for investigating a network collapsing technique is illustrated in the examples of Figure 2.6. A single polynomial of the same order as the model descriptions can be extracted for the cases of:

- a pure series connection of elements whose voltage characteristics are expressed as a polynomial of current.
- a pure parallel connection of elements whose current characteristics are expressed as a polynomial of voltage.

This is essentially the basis of a series / parallel reduction technique for a graph of the circuit. Such an approach would only be applicable to the restricted class of series / parallel networks, i.e. circuits without feedback elements. However, this is the most basic form of network reduction. Its lack of

practicality, presented in the next two subsections, illustrates that extracting explicit solutions by circuit reduction techniques has very limited potential.

a) Series Connection

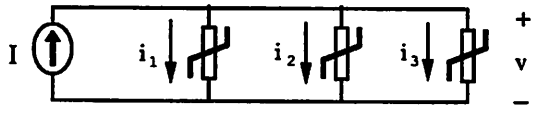


$E = v_1 + v_2 + v_3$

$E = p_1(i) + p_2(i) + p_3(i)$

=> Explicitly Solvable for i

b) Parallel Connection



$i_1 = p_1(v)$

$i_2 = p_2(v)$

$i_3 = p_3(v)$

$i_1 + i_2 + i_3 = I$

$p_1(v) + p_2(v) + p_3(v) = I$

=> Explicitly Solvable for v..

Figure 2.6: *Series / Parallel Circuit Reduction*

Single Low-Order Polynomial Models

To retain the ability to solve the network explicitly, the polynomials describing the collapsed sections of circuit cannot be order greater than three. This requirement explicitly relates the desired model description (whether $i = p^3(v)$ or $v = p^3(i)$) to the topology of the circuit. In the case of parallel combination the former representation is demanded while a series combination requires the latter. A simple example is sufficient to justify this necessity. Consider the series combination of two elements, one device with terminal voltage described as a polynomial of current the other with current in terms of voltage (Figure 2.7).

Assume second-order polynomial models of the form:

$$i = a_2 v_2^2 + a_1 v_2 \tag{2.3}$$

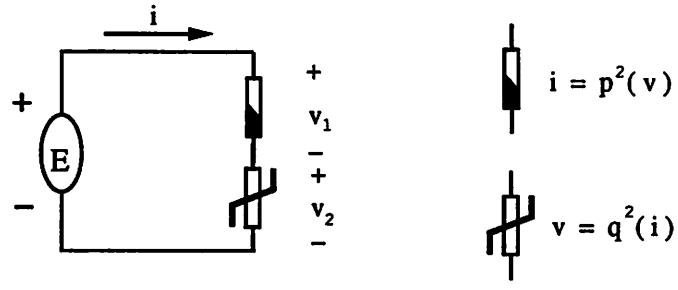


Figure 2.7: *Dependence of Model Description Upon Explicit Solvability Property*

$$v_2 = E - v_1 = b_1 i^2 + b_1 i \tag{2.4}$$

Collapsing this network corresponds to substituting the second relation into the first. This produces a fourth-order polynomial in i. The implication of this result is that to achieve independence between model description and circuit topology, all devices must have representations both as $i = p^3(v)$ and $v = p^3(i)$. In general, if $i = p^3(v)$ is a good match to characteristics this does not imply that there exists a good polynomial description of the form $v = p^3(i)$, and vice versa.
(Consider $v = i^2$ on the interval $[0, 1]$ for which $i = \sqrt{v}$ on $[0, 1]$.)

Even the existence of device models with polynomial descriptions $i = p^3(v)$ and $v = p^3(i)$ is not sufficient to guarantee extraction of a polynomial for a node in the network which can be solved explicitly. Collapsing series elements produces a polynomial of voltage in terms of current, parallel elements reduce to a polynomial of current in terms of voltage. However, the placement of these subcircuits in the topology determines the requirement for $i = p^3(v)$ or $v = p^3(i)$ for their collapsed representation. The extraction of a polynomial of practically solvable order is then dependent upon the construction of an inverse low-order polynomial approximation. In fact, rather than attempting to describe models in both forms, such a scheme would also be used at the device level.

The equations extracted through the use of local polynomial approximations to describe network sections are only applicable over a limited range. Consequently, even though results are computed explicitly, any procedure using this technique would have to be iterative to guarantee accuracy. The formulation of such an iterative algorithm implies that polynomial approximations must remain valid for all devices. This requirement can only be satisfied by computing a solution for all nodes in the network attached to or controlling a nonlinear element so that local polynomial approximations can be updated if required. That is, even if the simulation question being asked required transient behaviour for only a few nodes of interest, in general the subset of nodes for

which a solution must be computed is unlikely to be significantly smaller than the set of all nodes. Although the region of applicability would be improved, there is significantly greater complexity in formulating a polynomial approximation over derivative determination (linearization). When combined with the generally low number of NR iterations required to converge at each time-point of the transient analysis in practical cases, this suggests that the explicit solution approach would be considerably more computationally expensive.

Piecewise Continuous Low-Order Polynomial Models

In a practical simulator, the benefit of a piecewise-continuous polynomial modelling scheme to the "explicitly solvable" approach is further reduced. This is a consequence of range of applicability of the low-order polynomial description being a property of model construction and present iteration point. Consider the curve of Figure 2.8. This depicts the region of applicability of a polynomial if a local approximation is made in comparison to the function extracted from model representation. In this case, even though the device description may be of the desired form for the topology ($v = p^3(i)$ or $i = p^3(v)$), the proximity of the iteration point to the region boundary restricts the usefulness of the approximation.

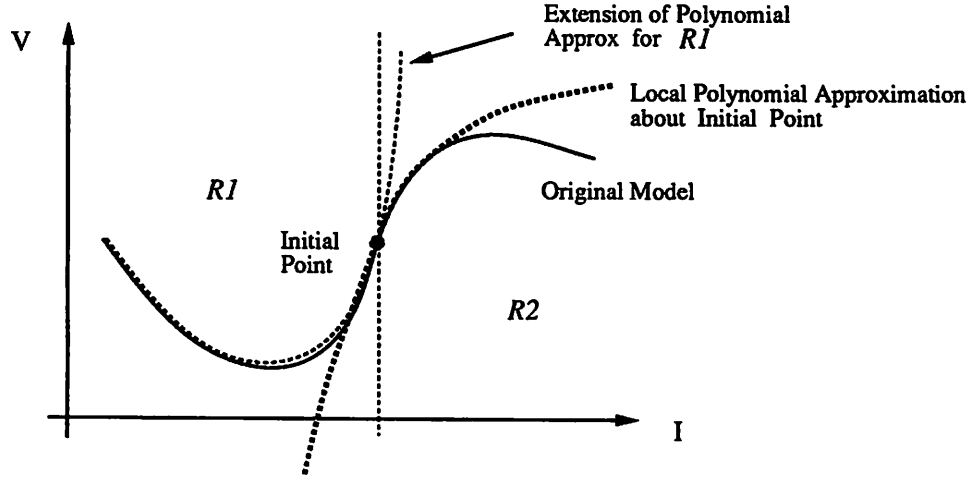


Figure 2.8: *Model and Local Approximation Polynomial Descriptions*

A polynomial description applicable in one region is in no way guaranteed to be even a close approximation when extended into adjacent regions. Consequently, if a solution point is found outside the present polytope through explicit use of the device polynomials, there is no generally provable relation between that and the desired solution. In fact, such a result can not even be utilized

to determine in which region the solution actually lies. This prevents formulation of an iterative procedure from this basis.

It is not sufficient when checking the validity of a solution to examine only the results at the desired nodes. If device polynomial descriptions are used directly for some elements, every node controlling those elements must be computed. The validity of the final solution depends upon every polynomial approximation used in the relevant collapsed subcircuits remaining valid. Furthermore, nonlinearity prevents the use of superposition techniques. If the describing polynomial for a device is invalid, all of the dependent collapsed subcircuit descriptions must be recomputed from scratch.

The practical consequence of these results is that a low-order polynomial description must be generated for the function about the present iteration point. The polynomial extracted from the piecewise-polynomial descriptions of the models does not have the desirable properties of symmetry or bounded accuracy about a point through its construction. This does not permit the formulation of a convergent iterative approach using this form of device description. Consequently, a piecewise-continuous modelling scheme exploiting an explicit solution-property inhereted from local-polynomial descriptions cannot offer speedup over standard NR linearization.

2.2.2 The Relaxation Approach

At the NL equation level, obtaining an advantage over standard relaxation techniques for networks with arbitrary device descriptions (other than through ease of linearization) depends upon the ability to solve circuit relations explicitly. A set of polynomials of known order is closed under addition so the set of equations obtained from KCL and constitutive branch equations is the same order as the model representations. In fact, if all devices are expressed with current a polynomial of voltage (except for voltage sources, voltage controlled voltage sources (VCVS) and CCCS), the set of variables required to maintain a complete description of known order is the same as for the MNA form after linearization. The principle of relaxation allows each equation to be taken separately and analysed for a single variable with all other variables fixed as described in Section 1.2.3. In this way, every equation is reduced to a single variable polynomial of less than third-order which can be solved directly.

Techniques exploited to force better convergence of relaxation based approaches are equally applicable regardless of the mathematical representation of the models. For example, requiring capacitors to ground at each node to ensure diagonal dominance of the linearized matrix can be used to improve convergence of all approaches. This is a consequence of the convergence

within a region about a solution point being depending upon properties of the Jacobian matrix (Ref. Section 1.2.3.). Convergence rates thus derived actually assume solution of the equations in each iteration to within a small bounded error, except in the case of Iterated Timing Analysis.

The problem of ordering the equations in such a way as to optimize the number of relaxation iterations is equivalent to those adopted in Gauss-Jacobi-Newton or Gauss-Seidel-Newton approaches. Consequently, these ordering techniques are not examined further in this report.

Nonlinear Relaxation and Iterated Timing Analysis

The problem with this approach is clearly a consequence of the possibility of the solution being outside the present polytope. For an arbitrary network, it is not generally possible to bound the time-step and so ensure that the solution for the next time-point remains within the region (Ref. Section 3.1.2.). The model descriptions for all devices cannot be exploited outside their region of validity. Futhermore, an overall solution obtained outside this polytope cannot be utilized to determine in which region the solution actually lies. (The reasons for this are similar to those outlined in Section 2.2.1) However, as only a single nonlinear relation is numerically solved at a time, it is possible to progressively update the set of describing relations.

Even in the case of a single variable piecewise-polynomial relation, the solution of the describing polynomial for the present region cannot suggest a direction towards a valid solution outside the polytope. A counter example to moving in the direction of an invalid solution is depicted in Figure 2.9.

For the single variable piecewise-polynomial relation, the solution lies either to the left or right of the present point. Furthermore, the length of the time-step has been determined such that over that interval, a polynomial in time of the same order as the integration method used is a sufficiently accurate description of the waveforms. For example, if the TR method is used, the variation with time over each interval is a parabolic approximation. Given that variables can be described as polynomials of known order and that a single equation is being solved, it is appropriate to ask whether in this particular case the time-step to a region boundary can be determined. The approach would involve replacing the variable being solved by a boundary value, all other variables by a second-order time description and then solving for t. The ability to do so would allow the direction towards the solution to be determined.

Consider the form of the polynomial relation to be solved for current summation at a node. It is constructed from models described as $i = p^3(v)$. With respect to node voltages there

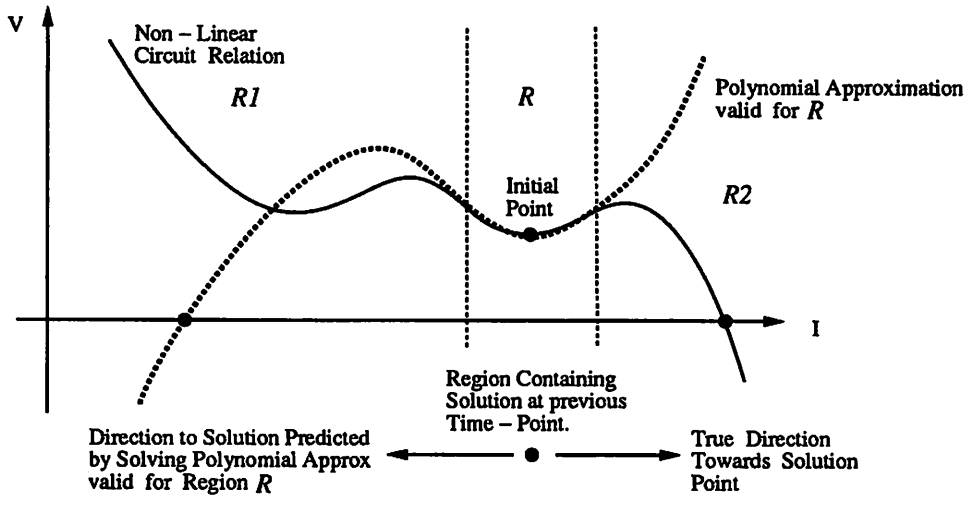


Figure 2.9: *Counter Example to Moving in Direction of Invalid Solution*

may exist terms $(e_i - e_j)^3$ producing the products e_i^3. If the node voltages can be expressed as second-order polynomials of time, a sixth-order polynomial in t can be produced. Even restricting the models to second-order, polynomials of greater than fourth-order (not explicitly solvable) are readily constructible. The presence of capacitors, linear or nonlinear (Ref. Section 3.1.1) implies that there can exist coefficient terms in the overall equation of the form $\frac{a}{h}$, where h is the time-step. To compute the time to the region boundary, h is replaced by t (thereby maintaining a valid description for TR step size less than h). This has to be multiplied through to produce a polynomial in t which could now be of fifth-order, e.g., $t.(p^2(t))^2 \rightarrow p^5(t)$. The time-step to a region boundary cannot be computed explicitly. Nevertheless, the time-step computed was only to be used to determine whether the desired solution is beyond a certain boundary. Without a necessity for accuracy, convergence of NR applied to this polynomial in time could similarly be used as an indicator. Approaching a solution inside the time-step, h, implies this boundary is crossed, a solution outside implying the other is (assuming the present region has been determined to contain no solution). The direction towards the solution need only be determined once. Analysis would proceed by progressively adjusting the circuit-describing relations appropriate the to regions in the specified direction until a solution is found. Practically, it may be found quicker not to construct a polynomial in time to determine direction. An alternative approach could involve just examining the "left" adjacent region, then the "right", next left, next right and so on until a solution is found.

The usefulness of the explicit solution approach clearly depends upon the average time required to compute a valid explicit solution versus convergence of Newton Raphson. Standard NR

techniques have shown adequate convergence in an average of not much more than three iterations. Given the simplicity of extracting derivatives from polynomials, the computational complexity of finding the roots of the third-order polynomials would be predictably larger than application of standard NR. One should note that explicit solution of third-order polynomials involves radicals which are frequently obtained through NR techniques in practical applications. The explicit solution technique clearly becomes impractical if the average number of regions examined for an explicit solution is somewhat greater than one.

The discussion of the computational trade-off between NR and explicit solution techniques is, however, almost a moot point. The factor implying this is the proof that ITA has the same rate of convergence as NL relaxation for an initial guess sufficiently close to the solution [15]. This proof demonstrates that accurate solutions to the inner loop relations are not critical to relaxation performance in transient analysis. ITA has also been shown to execute significantly faster than full NL relaxation in practice, especially when selective trace techniques are also used [14]. This is sufficient to essentially eliminate any need to experimentally examine the trade-off between convergent NR and explicit solution techniques.

Timing Analysis

Timing analysis requires the accurate computation of the solutions to each equation (Ref. Section 1.2.3.). The time complexity of finding these solutions explicitly is exactly equivalent to that presented previously. However, in this case the trade-off between NR convergence and the explicit solution method is crucial.

2.3 Practicality of Application to the NL Level

The applicability of low-order polynomial modelling has already been demonstrated at the NL equation level. The increased ease of constructing of a set of linear equations is exploited in existing simulators, such as CAzM. This approach might be improved further through selective updating of the set of nonlinear circuit relations. Thorough investigation of such a technique would involve empirical examination of the trade-off in complexity between determining intersections of NR vectors with polytope boundaries and full reconstruction of the Jacobian matrix at every NR iteration. Unfortunately this appears to be the only unexplored avenue which offers any promise of speed-up at the NL equation level. Possibilities for improving NL analysis through utilization of

device descriptions have been examined in the application of a generalization of the Katzenelson algorithm to transient analysis, and the elimination of the need for linearization. Both approaches proved unfruitful.

Use of the Katzenelson algorithm in transient analysis does not maintain many of the desirable properties of the technique even in the case of piecewise-linear device descriptions. For example, it is impractical to manipulate infinite region derivatives to ensure that the method is guaranteed to find an existing solution. Furthermore, if higher-order modelling is allowed, the inability to exactly trace the solution path and the loss of the significance of region boundaries (points of singularity for the Jacobian matrix being critical) prevents a generalization of the method being practical, even for dc analysis.

Knowledge that the circuit relations are polynomials of the same order as the models did not prove valuable. As these polynomials are multi-variable in general, finding explicit solutions to avoid application of NR involves either collapsing the circuit or use of relaxation techniques. Collapsing is impractical as there is no guarantee that the solution lies within the present polytope (region of valid NL description) and circuit topology fixes requirement for the form of device/ subcircuit description ($v = p(i)$ or $i = p(v)$). In NL relaxation, as each equation is solved as single-variable-dependent, the trade-off between complexity of explicit solution techniques and convergent NR is somewhat less apparent. However, it has been shown that ITA which only uses a single NR step towards the solution of the NL equations in the inner computational loop converges as rapidly as NL relaxation. There is therefore little justification in the added complexity of obtaining the exact result provided by explicit solution techniques.

Chapter 3

The Nonlinear, First Order Differential Equation Level (DE)

3.1 Formulating a System of Nonlinear Equations (DE.NL)

3.1.1 LMS Methods

Handling Nonlinear Capacitors

Many of the results of the previous chapter are derived assuming that the set of nonlinear equations extracted from circuit relations are polynomials of order not greater than the model descriptions. As a set of polynomials of known order is closed under addition, this assumption is obviously true in the case of non-derivative-dependent nonlinear components. However, for general circuit analysis such components may be present (the collector-emitter capacitance of a BJT, for example). It must therefore be shown that a representation exists for these elements such that the known order property is maintained without a need for NR linearization. Any model linearization implies a need for iterative convergence even if solutions to the polynomials resulting from non-derivative-dependent components are found explicitly.

Consider a nonlinear capacitor represented in the form $q = p^3(v)$, where q is charge. This implies $i = \frac{d(p^3(v))}{dv}\frac{dv}{dt} = g^2(v)v'$ and after application of the TR integration method this becomes:

$$i_{n+1} = (2\frac{v_{n+1} - v_n}{h_{n+1}} - v'_n)g^2(v_{n+1}) = f^3(v_{n+1}) \tag{3.1}$$

Consequently, current at the present time-point, i_{n+1}, is a polynomial of the present-time-point

voltage, v_{n+1}, of the same order as the model. This is true regardless of the LMS method used. Similarly, in the case of nonlinear inductors, a polynomial description of flux in terms of current, $\phi = p^3(i)$, should be used. These representations maintain the property of known polynomial order for KCL and separate branch equations of the MNA representation.

Accuracy and Stability

Performance of integration methods used for solving entirely arbitrary differential equations is assessed by their stability and accuracy when applied to exponential waveforms. For a method of n^{th}-order accuracy, when applied to an exponential with a real exponent, the result is exact for any truncation of the Taylor series expansion of the exponential below the $(n+2)^{th}$ term. If the device models are represented in piecewise-polynomial form, an examination of whether the generality of the differential equations has been reduced is important. A positive result could allow a more problem-specific definition of accuracy and stability, thereby improving time-step estimation in circuit simulation.

A counter example to disprove this proposal exists even if models are restricted to second-order polynomials and inputs vary linearly with time. A multiplier can be designed as shown in Figure 3.1 from basic op-amp analog adders and second-order nonlinear devices. Through the use

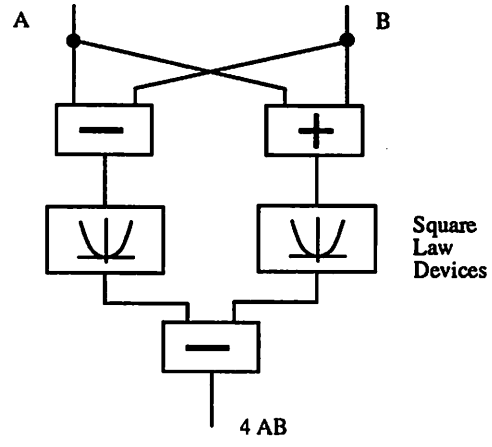


Figure 3.1: *Multiplier Construction Using Second Order Nonlinearities*

of multipliers, delay lines and amplifiers, it is straight forward to construct a circuit which generates a truncated Taylor series expansion (any order) for an exponential (real or imaginary exponent) given a linearly-time-varying input to the network. Numerical accuracy of an integration method is determined by the dominant error term of the truncated series. Consequently, even in this over

simplified case, the generality of the circuit waveforms has not been reduced significantly. This implies that piecewise-polynomial modelling cannot provide an insight into improved time-step estimation.

3.1.2 Time Step Restriction

The problem with an explicit solution approach is the possibility that the solution may lie outside the polytope of valid description (Ref. Section 2.2.1 & 2.2.2). Consequently, the existence of an efficient analytical approach for bounding the time-step in order to remain within the region is desirable. Such a technique could be utilized to determine whether a solution should be found explicitly or by using an iterative technique . (An iterative technique, such as NR, would be used if the time-step required to remain within the polytope is significantly smaller than that bounded by integration method accuracy.)

The complexity of computing the maximum time-step, h_{n+1}, needed to remain within the present polytope is large for even simple circuits. Consider the diode / capacitor example of Figure 3.2. By utilizing the TR integration method:

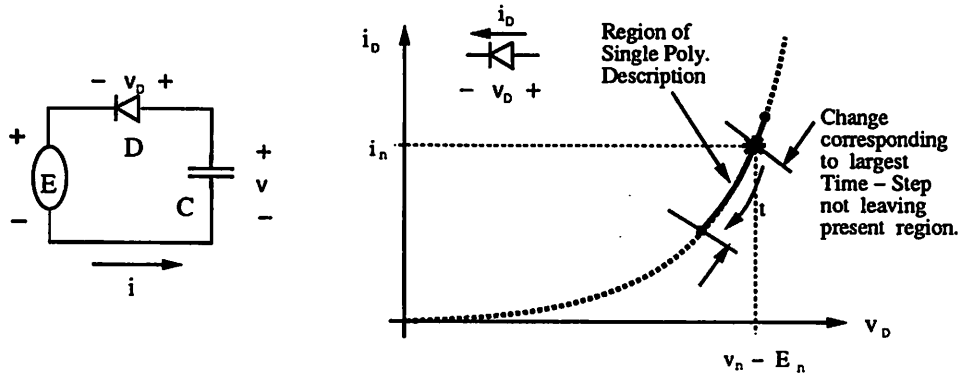


Figure 3.2: *Computing Restricted Time Step for Simple Circuit*

$$v'_{n+1} = 2\frac{v_{n+1} - v_n}{h_{n+1}} - v'_n \tag{3.2}$$

$$\Rightarrow i_{n+1} = \frac{2C}{h_{n+1}} v_d - \frac{2C}{h_{n+1}}(v_n - v_s) - i_n \tag{3.3}$$

To compute the time-step to the boundary, assign to v_d a boundary value and substitute the corresponding value of i_{n+1} and an expression for source voltage variation (in terms of h_{n+1}; that is, time for which $h_{n+1} = 0$ implies the previous time-point). This relation is then solved for h_{n+1}. A

result less than the time-step bound for accuracy of the integration method implies that the boundary corresponding to the value of v_d substituted will be crossed. Futhermore, the boundary value is attained for that value of time-step. A result greater than the bound for accuracy implies that the specified boundary is not crossed in this time-step. However, it gives no information as to whether the other polytope boundary is crossed. (Of course, in this case, if the time-step bounds are close, crossing of the other boundary is unlikely. However, as the performance of the LMS method is not guaranteed beyond the accuracy bound, this is a statistically variable property.) To be certain that the solution lies within the polytope, all boundary conditions must be examined.

In the chosen example, if the source voltage variation is a polynomial variation of less than third-order, the final expression can be solved explicitly. However, even using the fact that within the time-step bound for accuracy all waveforms can be assigned a low-order polynomial description, in general the resultant equations are not solvable explicitly (Ref. Section 2.2.2).

The time expressions obtained for the network would need to be solved by numerical techniques. In an N dimensional space, up to $2N$ boundary conditions would need to be assessed. Each boundary examination would be of somewhat greater complexity than finding a NR solution to the set of circuit relations for the present polytope. (The order of the equations may be greater than the order of model representations.) Consequently, in general circuit simulation, it would not be practically possible to determine a bound on h_{n+1} to ensure that the solution to the next time-point lies within the present polytope.

3.2 Solving Without Formulating Global System of Nonlinear Equations (NL.NL)

3.2.1 Decoupled LMS Methods

Waveform Relaxation (WR)

Inside an iteration of waveform relaxation, each circuit relation is solved separately over a specified time interval. That is, a relation of the form:

$$f_n(\dot{x}(t), x(t), u(t)) = 0, \qquad 0 \leq t \leq T \tag{3.4}$$

is solved for variable $x_n(t)$, all other variables having specified variation for all $\{t : 0 \leq t \leq T\}$. Essentially, each analysis of this form is equivalent to solving a node in a multi-input, single-unknown circuit where all inputs are represented in piecewise-polynomial form

(Ref. Figure 3.3). The construction of known waveforms as piecewise-polynomials is a con-

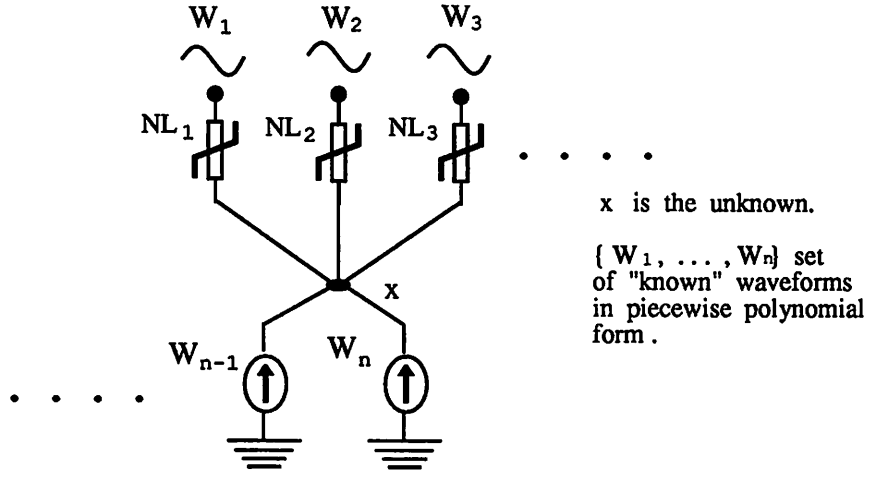


Figure 3.3: *Diagramatic Representation of Step in WR*

sequence of the LMS method used. Between time-steps taken in the solution of each node, a polynomial approximation of the same order as the LMS method is valid. (To complete the analogy, a piecewise-polynomial representation of actual network inputs would be required.) Clearly, any advantage to be obtained as a consequence of model representation will be seen in the tradeoff between standard NR and an explicit solution approach.

The problem of determining in which polytope the solution at the next time-point lies is similar to that associated with ITA (Ref. Section 2.2.2). Although the equation produced through application of the LMS method can be solved explicitly, (Ref. Section 3.1.1), a result outside the valid region of description is not useful. Moreover, for an arbitrary node in the circuit, traversal of a region boundary is more likely in WR analysis than for a step taken in ITA. (In ITA, the step length is not necessarily determined by the variation of the waveform being computed so it tends to be shorter.) An examination of the complexity for determining boundary crossings is therefore relevant.

Unlike ITA, a single polynomial to be solved for boundary conditions cannot necessarily be constructed through the substitution of polynomial functions in time for the "fixed" variables. (That is all variables other than $x_n(t)$ from Equation 3.4). As the time-step of adjacent waveforms may not be strongly correlated to those of the node or branch being examined, several polynomial sections may have to be analysed (Ref. Figure 3.4). The construction of these sections from the time-steps of adjacent nodes reduces the computational independence of the waveform being examined.

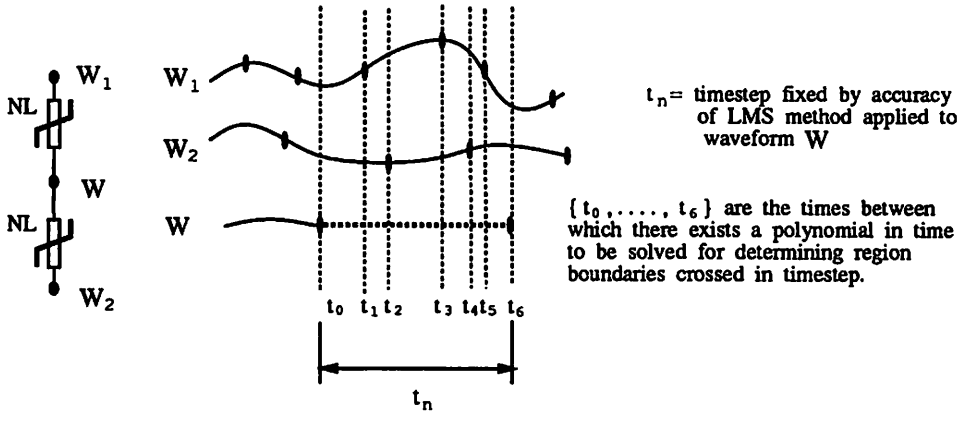


Figure 3.4: *Dependence of Polynomial Description in Time upon Time-Steps of Local Nodes*

Futhermore, analysis of each section for boundary crossings has the same time complexity as that used to compute a single time-step of an equation in ITA. This reduced independence of waveform analysis and significant computational cost of locating boundary crossings makes the determination of the time-step-to-boundary infeasible. If an explicit solution is to be found, the approach to take is that of examining the "left" adjacent region, "right" adjacent region, and so on until a valid solution is found (Ref. Section 2.2.2). However, for each region a third-order polynomial must be solved. This implies that an average of much greater than one polytope passed through per time-step will bias computational complexity considerably in favour of standard NR.

Of all the techniques examined in this report, it is probably the application of reduced-order modelling to WR techniques for which the trade-off between explicit solution and standard numerical analysis is least well defined. However, as for any iterative technique taken to convergence, e.g., ITA , the demand for exact solutions on each iteration is significantly reduced. Implementation of efficient Waveform-Newton simulators (in which only a single NR step is taken to define the next time-point solution for a waveform of the present iteration) has shown this to be true for WR techniques [21]. Given that NR need not be executed to within the tight bounds on accuracy required for a global solution at a time-point, the computational complexity of obtaining an exact solution is not warranted.

3.2.2 The Closed-Form Solution to O.D.E's:

Extracting closed-form solutions to circuit relations is not possible in general. However, the ability to determine such a solution eliminates the need to apply integration methods. It is

consequently relevant to examine whether such a solution technique is plausible in the case of reduced-order modelling.

One approach to constructing a closed-form solution is through the summation of a set of basis functions. Impractical in general, this modelling scheme might be expected to allow a reduced set of basis functions to be specified. Extraction of a finite basis set could permit adequate approximation of the differential equation solutions through determination of the relevant coefficients. The following discussion proves the non-existence of a set of practical cardinality.

A set of basis functions is a complete set of orthonormal functions, $\{\phi_n\}$ which is closed under multiplication . For example, in the case of a periodic waveform of period T, the set of functions: $\{exp(j2n\pi(\frac{1}{T})t : n \in N\}$ is a basis [19] in that all such waveforms can be represented as a *Fourier Sum*:

$$x(t) = \sum_{n=-\infty}^{\infty} c_n exp(j2n\pi(\frac{1}{T})t) \tag{3.5}$$

The completeness of the set implies that for any function, $x(t)$, the restricted sum, $\sum_{n=1}^{M} c_n\phi_n$ approaches $x(t)$ uniformly.

i.e. For any $\epsilon > 0, \exists M_x \in N s.t. \forall m > M_x, \|x(t) - \sum_{n=1}^{m} c_n\phi_n\| < \epsilon$.

However, for arbitrary continuous functions (such as those produced in general simulation), there does not exist M independent of $x(t)$ such that for all such functions the above relation holds.

If the simulator uses reduced-order models, then within the bounds of a polytope in the solution space, the waveform description at a node can be a polynomial in time of arbitrary order (Ref. Section 3.1.1). A corollary of the *Stone-Weierstrauss Theorem* [17], a theorem from elementary topology which relates the ability to uniformly approximate continuous functions by elements of a subalgebra in the space of continuous functions, states:

Every continuous function on a closed bounded set X in $\Re^n$ can be uniformly approximated on X by a polynomial (in the co-ordinates)

This is equivalent to stating that the closure of the set of all polynomials is the set of continuous functions over that interval. (In this case, the closed bounded interval is the time over which the circuit functionality is to be examined.) The argument for the non-existence of a basis set of practical size proceeds by contradiction. Assume that there exists a finite subset, cardinality M, of a set of basis functions, $\{\phi_n(t)\}$, such that a weighted summation of elements of that subset can be used to uniformly approximate any polynomial, $p(t)$, to within error $\frac{\epsilon}{2}$.

i.e. For any $\frac{\epsilon}{2} > 0, \exists M \in N s.t. \|p(t) - \sum_{n=1}^{M} c_n\phi_n(t)\| < \frac{\epsilon}{2}$.

However, by corollary, for any continuous function $f(t)$, there exists a polynomial $p(t)$ such that

$\|f(t) - p(t)\| < \frac{\epsilon}{2}$. Hence, $\exists M \in N s.t. \|f(t) - \sum_{n=1}^{M} c_n \phi_n(t)\| < \epsilon$. This implies that the this finite subset of the basis could be used to approximate any continuous function uniformly. As this subset is not closed under multiplication, it is not a subalgebra and this is a denial of the Stone-Weierstrauss Theorem. Therefore, such a set does not exist.

The complexity of constructing a closed-form solution from a set of basis functions is consequently no more difficult in the general case than for reduced-order modelling. If a closed-form solution is to be extracted, it must be related to the polynomial form of the differential equations explicitly.

Allowing Nonlinear Capacitors: In general, a differential equation (generated by the summation of currents at a node) will be of the form:

$$p_1(v_1)\frac{dv_1(t)}{dt} + p_2(v_2)\frac{dv_2(t)}{dt} + ... + q_1(v_1) + q_2(v_2) + ... + f_1(t) + f_2(t) + ... = 0 \quad (3.6)$$

where input sources are represented as polynomials in time, $f_i(t)$, and polynomials $p_i(v), q_i(v)$ are extracted from model relations. Consider now the simple example depicted in Figure 3.5 of a nonlinear capacitor in series with a linear resistor driven by a voltage source of polynomial variation. For the above circuit, the describing relations are:

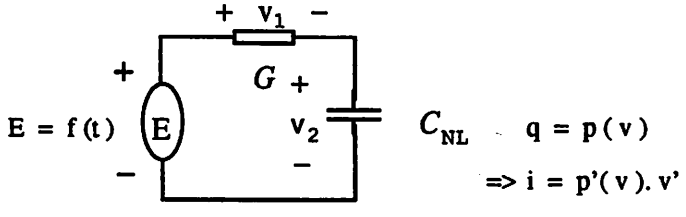


Figure 3.5: *Simple Nonlinear Capacitor Example*

$$Gv_2(t) - p'(v_1).v_1'(t) = 0 \quad (3.7)$$

$$v_1(t) + v_2(t) - g(t) = 0 \quad (3.8)$$

where $g(t)$ is a polynomial. This reduces to the solution of a differential equation of the form:

$$v_1'(t) = \frac{G(v_1(t) + g(t))}{p'(v_1)} \quad (3.9)$$

This equation does not have a general solution over all time. In fact, the only closed-form expression known for an arbitrary differential equation of the type:

$$u' = \frac{p(u,t)}{q(u,t)} \quad (3.10)$$

is that in the asymptotic limit which assumes one of the following forms:

$$u'(t) \sim at^b e^{g(t)}$$

$$u'(t) \sim at^b (log(t))^{\frac{1}{c}}$$

where $g(t)$ is a polynomial and a, b, c are constants [1]. This is clearly not a useful result as the differential equations describing the circuit remain valid over a only a restricted time interval (until polytope boundary is crossed). Furthermore, these asymptotic expressions are not even guaranteed to bound the solution close to time $t = 0$. Closed-form expressions are consequently implausible if nonlinear capacitors are not linearized.

All Capacitors Linear or Linearized: Linearization of nonlinear capacitors in a network implies that any closed-form solution is an approximation even within the polytope of valid description. The linearization implies that the general circuit relation DE is reduced to:

$$a_1 \frac{dv_1(t)}{dt} + a_2 \frac{dv_2(t)}{dt} + ... + q_1(v_1) + q_2(v_2) + ... + f_1(t) + f_2(t) + ... = 0 \qquad (3.11)$$

However, differential equations of the type:

$$\frac{du}{dt} = p(u, t) \qquad (3.12)$$

do not have closed-form solutions unless $p(u, t)$ is linear in u. That is, a simple closed-form solution can only be found in the case of linearization of all devices. Techniques such as *quasilinearization* can be used to obtain closed-form upper and lower bounds on the solution to the nonlinear DE . However, except in very particular situations, such as for the solution to the *Riccati Equation* :

$$\frac{du}{dt} = u^2(t) + a(t) \qquad (3.13)$$

these bounds are neither simple to compute nor provably tight.

There is no evidence to suggest that a general technique to finding a closed-form expression satisfying the type of DE's described above will ever be developed. However, even supposing it were, such a function would definitely not be a low-order polynomial. An iterative technique such as NR would be required to determine when the solution crosses a region boundary (for determination of the region of applicability of the solution). Given the complexity of checking all boundary conditions (Ref. Section 2.2.2 & 3.1.2), such an approach could never compete with existing simulators.

3.3 Practicality of Application to the DE Level

The form of the model representation does not appear to have a significant effect upon the solution techniques at the differential equation level. In the construction of NL equations, the time-step cannot be bounded to ensure that the solution at the next time-point lies within the present region. Futhermore, it is not possible to guarantee accuracy and stability with larger time-steps in LMS methods even though differential equations are of a known form. Allowing model representations in piecewise-polynomials of greater than first-order precludes closed-form descriptions of waveform solutions. In fact, the modelling technique does not even permit the construction of a smaller set of basis functions than would be used in general simulation. Both the inability to reduce the set of basis functions and construct a more appropriate description for LMS accuracy are a consequence of the fact that if second-order circuit elements are permitted, waveforms of arbitrary polynomials in time can be generated.

It is only in its application to waveform relaxation that the known polynomial form of the differential equations is even theoretically applicable. After extraction of NL equations using LMS methods, a solution at the next time-point could be found explicitly. However, unlike its application to NL relaxation, it is not practical to determine the direction towards a solution which is not in the present region. Regions to the left and right of the present interval of validity (as a polytope is an interval in a single variable piecewise-polynomial relation) must be checked exhaustively. Given that the results need not be very exact (each waveform solution being the basis of an iterative procedure), little practical advantage over NR techniques is expected.

Chapter 4

Conclusions:

Low-order piecewise-continuous polynomial device modelling schemes should become more evident in the future development of circuit simulators. The decreasing cost of memory and implementation of efficient data handling techniques have made the past trade-off between memory space and simulator speed less significant. Simulators such as CAzM have clearly demonstrated the speed-up potential in using a rapid table look-up approach as the basis for construction of the Jacobian matrix. However, the benefit of this form of model representation appears to be almost entirely restricted to this straightforward application.

In a piecewise-polynomial model environment, the differential and nonlinear equations produced can be shown to be of the same order as device representations. However, for anything of higher order than piecewise-linear, closed form solutions are not plausible. The set of basis functions sufficient to describe generated waveforms is not a finite restriction on the set required for simulation with arbitrary models. Futhermore, it is not possible to develop better accuracy bounds on existing LMS method for the purpose of increasing time-step length. Explicit solution techniques at the nonlinear equation level can not be used to obtain a general solution, even if all models are a single polynomials. Use of the polynomial form of the relations to eliminate the need for NR is only possible in relaxation techniques such as ITA or WR. In both these cases (but more so for ITA than WR), the reduced need for exact solutions on each iteration strongly suggests that any trade-off in time between finding explicit solutions and applying NR would not be favourable. There does not exist a practical generalization of the Katzenelson algorithm from piecewise-linear to piecewise-polynomial models even for dc analysis.

The only approach which appears to warrant further investigation through implementation is a slight modification of existing piecewise-polynomial model simulators. Instead of fully

reconstructing the Jacobian matrix at every iteration, it may be possible to reduce complexity by updating the nonlinear circuit equations selectively. Such an updating scheme would be based on the boundaries crossed by the NR vector at each iteration. Overall, however, modelling schemes using piecewise-polynomial models of greater than second order promise little help in the development of improved nonlinear time-domain transient circuit analysis techniques.

Bibliography

[1] Richard Bellman, "Methods of Non-Linear Analysis", *Academic Press*, 1970

[2] j. Burns, A. R. Newton and D. O. Pederson, "Active Device Table Look-Up Models for Circuit Simulation ", *Proc. of the International Symposium on Circuits and Systems*, 1982, pp. 250-253

[3] C. de Boor, "A Practical Guide to Splines", *Springer-Verlag*, 1978

[4] M. J. Chien and E. S. Kuh, "Solving Piecewise-Linear Equations for Resistive Networks", *Circuit Theory and Applications*, Vol. 4, pp. 3-24, 1976

[5] Leon O. Chua and Robin L. P. Ying, "Finding All Solutions of Piecewise-Linear Circuits", *Circuit Theory and Applications*, Vol. 10, pp. 201-229, 1982

[6] W.M.Coughran,Jr., R.Grosse and D.J.Rose, "CAzM: A Circuit Analyzer with Macromodeling", *IEEE Transactions on Electron Devices*, Vol. ed-30, no. 9, pp. 1207 - 1213, Sept. 1982

[7] D.J.Erdman, S.W.Kenkel, G.B.Nifong, D.J.Rose and R.Subrahmanyan, "CAzM: A Numerically Robust, Table-Based Circuit Simulator", *Circuit Theory and Applications*, Vol. 10, pp.201-229, 1982

[8] Kenneth Hoffman and Ray Kunze, "Linear Algebra, 2nd edition", *Prentice / Hall International*, 1986

[9] P. Hsi and C. H. Lee, "Modified Cubic B-Spline Interpolation", *Proceedings of the IEEE*, vol. 69, no. 12, Dec. 1981, pp. 1590-92

[10] Jacob Katzenelson, "An Algorithm for Solving Nonlinear Resistor Networks", *The Bell System Technical Journal*, pp. 1605-1620, Oct. 1965

[11] K. S. Kundert and A. Sangiovanni-Vincentelli, "Sparse User's Guide - A Sparse Linear Equation Solver ", *Users Guide, U.C.B. E.E.C.S. Dept.*, 1988

[12] L.W.Nagel, "SPICE2: A Computer Program to Simulate Semiconductor Circuits", *U.C. Berkeley Department Memorandum no. ER-M520*, 1975

[13] A. Richard Newton and A. L. Sangiovanni-Vincentelli, "Relaxation-Based Electrical Simulation", *IEEE Transactions on Computer-Aided Design*, Vol. CAD-32, No. 4, pp. 308-330, Oct. 1984

[14] A. Richard Newton, "The Simulation of Large Scale Integrated Circuits", *PhD Dissertation, UC Berkeley*, 1978

[15] J.M.Ortega and W.C.Rheinboldt, "Iterative Solution of Non-Linear Equations in One and Several Variables", *New York, Academic Press*, 1970

[16] Thomas L. Quarles, "Analysis of Performance and Convergence Issues for Circuit Simulation", *U.C. Berkeley Department Memorandum no. ER-M89/42*, 1989

[17] H. L. Royden, "Real Analysis, 3rd edition", *Macmillan Publishing Co.*, 1988

[18] A. L. Sangiovanni-Vincentelli "Circuit Simulation", *Computer Design Aids for VLSI Circuits"*, The Netherlands: Sijthoff and Noordhoff, pp. 19-112, 1980

[19] Henry Stark and Franz B. Tuteur, "Modern Electrical Communications, Theory and Systems", *Prentice / Hall International*, 1979

[20] C. Visweswariah and R. A. Rohrer, "Piecewise Approximate Circuit Simulation", *ICCAD proceedings*, 1989

[21] J. White and A. L. Sangiovanni-Vincentelli, "RELAX2: A New Waveform Relaxation Approach for the Analysis of LSI MOS circuits", *Proc. 1983 Int. Symp. Circuits Syst.*, May 1983

Appendix A

A.1 Guaranteed Convergence of Katzenelson

Much of justification for describing devices with piecewise linear models for transient analysis is associated with the guarantee of finding a solution which the Katzenelson algorithm can offer. However, as detailed in the introduction, this guarantee is only provable in the case where the determinants of all Jacobian matrices in the infinite regions have the same sign.

The following set of lemmas are used to construct a proof that to maintain continuity over region boundaries, adjustment of derivatives to satisfy the determinant sign property in the singly infinite regions alone is not sufficient to guarantee that property overall. Consequently, the task of satisfying this property for a localised subset of the entire solution space at each time-point is impractical. This severely reduces the desirability of using such a method in transient analysis (Ref. sect. 2.1.2).

Uncorrelated adjustment of Jacobian matrix entries in singly infinite regions which are unbounded in different dimensions is only plausible if those dimensions are orthogonal. Lattice structure spaces are therefore the only type under consideration here, each dimension being an axial direction.

Lemma 1 *If there exist totally bounded regions* [1] *then for each dimension of the space, there exists at least one singly infinite region of that dimension. Furthermore, multiply infinite regions do not share a face with a totally bounded region.*

Proof: Consider a totally bounded region R in the space. Take a point x in R. As bounding hyperplanes extend over the entire space and are mutually orthogonal, any vector v_j originating

[1] A *totally bounded region* is defined to be one such that any straight line within the region is of finite length.

from x parallel to any axis, x_j say, intersects only those planes perpendicular to x_j. The solution space is constructed from a circuit which is a finite combination of elements. Each element is described by a model with a bounded region of validity so the space of totally bounded regions in the solution space is itself totally bounded. Consequently, vector v_j must eventually extend into a region, R_j, infinite in dimension x_j. As R is bounded in all dimensions and hyperplanes are infinite, R_j is bounded in all dimensions except x_j. ie. R_j is a singly infinite region. As x_j is arbitrary, it follows that for each dimension in the space, there exists at least one singly infinite region of that dimension. By lattice structure, if there exists a totally bounded face to region R', then R' is either singly infinite or totally bounded. The second part of the statement follows by contraposition. □

Lemma 2 *If there exists two singly infinite regions of the same dimension and direction, R_1 and R_2, then there exists a path from R_1 to R_2 which traverses only singly infinite regions of that dimension without passing through corners. Furthermore, all these regions share the same hyperplane with the totally bounded regions.*

Proof: Prove the second statement first. Suppose the totally bounded faces of R_1 and R_2 lie on the hyperplanes K_1 and K_2 respectively. By the lattice structure, K_1 and K_2 are infinite planes perpendicular to the infinite dimension of R_1 and R_2, x_j. Hence, K_1 and K_2 are parallel. If $K_1 \neq K_2$, this implies that either K_1 intersects R_2 or K_2 intersects R_1 contradicting the bounding properties of the planes. Hence,$K_1 = K_2 = K$ so there exists a single hyperplane bounding R_1 and R_2. To prove the first statement, fix dimension $x_i \neq x_j$. As R_1 and R_2 are singly infinite regions, there exist planes: $K_{1a_i}, K_{1b_i}; K_{2a_i}, K_{2b_i}$ perpendicular to x_i which bound R_1 and R_2 respectively. Consequently, a set of hyperplanes:

$\{K, (K_{1_1}, K_{2_1}), ..., (K_{1_{j-1}}, K_{2_{j-1}}), (K_{1_{j+1}}, K_{2_{j+1}}), ..., (K_{1_N}, K_{2_N})\}$

can be chosen such that these set of planes bound a subspace containing R_1 and R_2. This subspace is singly infinite in dimension x_j and so all regions within this subspace are also singly infinite, by the second statement of the lemma. Construction of a path from R_1 to R_2 through only singly infinite regions without crossing corner points is now trivial. □

Lemma 3 *All singly infinite regions of the same dimension and direction have the same functional derivative in that axial direction. ie. All singly infinite regions of the same dimension and direction share a common column in the Jacobian matrix.*

Proof: In each region, the piecewise linear property implies that $\frac{\partial f_i}{\partial x_j}$ is constant. However, the function is continuous over region boundaries so $\frac{\partial f_i}{\partial x_j}$ is the same along any boundary parallel to the x_j axis. Hence, any two adjacent singly infinite regions of dimension x_j have the same partial derivative in that direction. By Lemma 3, it is possible to trace a path between any regions R_1 and R_2 singly infinite in the same direction and dimension x_j which only passes through other singly infinite regions of the same dimension. It follows that all regions singly infinite in dimension x_j and of that direction have the same value of $\frac{\partial f_i}{\partial x_j}$. □

Lemma 4 *All regions infinite in a particular dimension and direction have the same functional derivative along that axis.*

Proof: Consider the fact that a region unbounded in n dimensions can be adjacent to only regions of $n-1, n$ and $n+1$ infinite dimensionality. Consider region R infinite in orthogonal dimensions x_i and x_j. By Lemma 1, if there exist non-empty totally bounded regions in the space R must be adjacent to regions singly infinite in the dimensions x_i and x_j respectively. Consequently, by continuity, the functional derivatives in R along the dimensions x_i and x_j are the same as for the singly infinite regions. Now, in general, want to prove that given a region R infinite in multiple dimensions including some direction of x_j, there exists a path starting in R not passing through region corners which extends into regions singly infinite in the same direction of x_j. Consider, an n dimensional infinite region unbounded along axis x_j must share a planar face with an $n-1$ dimensional infinite region also unbounded in dimension x_j. If not, there exist no totally bounded regions. From a trivial extension of Lemma 2 and the previous statement, a path with the disired properties can be constructed. Consequently, using the same argument as for Lemma 3, it follows that all regions infinite in dimension x_j and of that direction have the same value of $\frac{\partial f_i}{\partial x_j}$. □

Lemma 5 *Allow $\frac{\partial f_i}{\partial x_j}$ to be changed in the regions singly infinite in dimension x_j. The function remains continuous over all space if for all regions infinite in the x_j dimension, $\frac{\partial f_i}{\partial x_j}$ is the same as for the singly infinite regions of that dimension and direction.*

Proof: Consider regions infinite in dimension x_j. Before changing $\frac{\partial f_i}{\partial x_j}$, the function is known to be continuous as the device models are continuous. Suppose $\frac{\partial f_i}{\partial x_j}$ is changed to $\frac{\partial f_i'}{\partial x_j}$ in all such regions. Hence for any x in these regions:

$f_i(x)_{new} = f_i(x)_{old} + (\frac{\partial f_i'}{\partial x_j} - \frac{\partial f_i}{\partial x_j})x_j = f_i(x)_{old} + Ax_j$

Hence, if: $f_i(x_1)_{old} - f_i(x_2)_{old} = \delta$,

$\Rightarrow f_i(x_1)_{new} - f_i(x_2)_{new} = \delta + A(x_{1_j} - x_{2_j})$

Hence, as A is finite $x_1 \rightarrow x_2$,

$\Rightarrow (f_i(x_1)_{new} - f_i(x_2)_{new}) \rightarrow 0$.

So by definition, f_i remains continuous.

However, i is arbitrary so the statement is generally true. □

Theorem 1 *Consider a lattice structure for a piecewise linear continuous function where the derivatives in the infinite regions are to be altered. Continuity is maintained iff in every region infinite in the same direction of dimension* x_j, $\frac{\partial f_i}{\partial x_j}$ *is the same.*

Proof: Direct result of Lemmas 4 and 5. □