MIT/LCS/TR-277

EFFICIENT MODELING

FOR

SHORT CHANNEL MOS CIRCUIT SIMULATION

RHH

Mark Griffin Johnson

This blank page was inserted to preserve pagination.

Efficient Modeling for Short Channel MOS Circuit Simulation

Mark Griffin Johnson

This research was performed at the MIT Laboratory for Computer Science, Cambridge, Massachusetts. It was supported in part by the Advanced Research Projects Agency of the Department of Defense, monitored by the Office of Naval Research under contract N00014-75-C-0661, and in part by a National Science Foundation Fellowship.

Efficient Modeling for Short Channel MOS Circuit Simulation

by

Mark Griffin Johnson

Submitted to the Department of Electrical Engineering and Computer Science
on August 13, 1982 in partial fulfillment of the requirements for the
Degree of Master of Science

Abstract

Existing circuit models for short-channel MOS transistors represent a compromise between computation speed and ease of use. Empirical models are very fast to evaluate, but their parameters must be fitted from experimental measurements. Theoretical models require longer computation time, but they may be used to predict the performance of new, unmeasured MOS technologies since their parameters are not curve-fitted from experimental data.

This thesis combines the best features of both types of model, yielding a fast circuit simulator whose input parameters need not be extracted from experimental measurements. A nonlinear optimization algorithm is used to "compile" the parameters of a theoretical model into parameters for an empirical model, providing the superior user-interface of theoretical models without sacrificing simulator execution speed. Results produced by a prototype model compiler are presented, showing the modeling error to be approximately 5 percent.

Thesis Supervisor: Stephen A. Ward

Title: Associate Professor of Computer Science and Engineering

Key words and phrases: MOS Transistor Modeling, Numerical Optimization, Nonlinear Parameter Estimation.

ACKNOWLEDGMENTS

I would like to thank Edward Feustel, Edmund Reese, and Joe McAlexander; without their advice, this thesis would never have been possible.

I would also like to thank my thesis advisor, Steve Ward, for helping in the selection of this thesis project and for his continued support.

Finally, I thank Angelo Miele, who introduced me to this field.

Table of Contents

Chapter 1: Introduction

One of the major challenges in integrated circuit engineering is to minimize the number of design iterations required to fully debug a chip. Error tracing is made difficult by the small size of the chip, since micro-probes or electron microscopes are required to monitor the behavior of internal nodes. When errors are finally located and corrections are proposed, they are both expensive and time-consuming to implement: the entire chip must be refabricated, even if the modification affects only one transistor. This process may cost up to $20,000 and require 6 weeks to complete.

The best way to avoid design revisions is to debug a chip before it is ever fabricated; usually this is done by computer simulation. Simulation makes circuit behavior easy to examine, so malfunctions can be rapidly located and repaired. Proposed modifications are then simulated to see if they remove the error. An iterative process of redesign and re-simulation is performed until the simulated circuit behavior is acceptable.

This thesis is concerned with the specific task of MOS circuit simulation; that is, accurately predicting the analog waveforms produced in a network of MOS transistors under specified input excitations. Simulation accuracy is impossible unless the nonlinear characteristics of the individual transistors are accurately modeled. Providing acceptable modeling accuracy at low (simulation) cost is the goal of this work.

High performance is obtained in modern MOS technologies by scaling down the transistor dimensions [11, 29]. Unfortunately, as they become smaller, transistors deviate significantly from the behavior predicted by classical gradual-field theories. These "short channel effects" drastically complicate the physical theory of transistor behavior, and hence increase the complexity of transistor models. Two distinct approaches to short channel transistor modeling have been successfully employed in circuit simulators: theoretical models (*e.g.*, [13]) and empirical models (*e.g.*, [19]). The terms "theoretical" and "empirical" are new; I use them to call attention to the

structural difference between two general classes of MOS models.

Theoretical models are created by performing a complete solid-state electrodynamic analysis of the MOS transistor structure. Each identifiable physical effect is explicitly accounted for, in an attempt to eliminate all possible sources of error. The resulting equations are then directly used as a model for circuit simulation.

The parameters of a theoretical model are simply the physical attributes of a particular MOS technology (*e.g.*, oxide thickness, substrate doping, junction depth, etc.). These constants are carefully measured and closely controlled in the semiconductor fabrication process. Because the model parameters are so accurately known, theoretical models are extremely easy to use: physical constants of the technology are plugged directly into the circuit simulator.

New MOS processes are also readily simulated with theoretical models, by inserting target values of the physical fabrication parameters into the simulator. Circuits can be accurately simulated even before the first wafer has been processed, allowing chip design and technology development to proceed in parallel. The price paid for this extreme flexibility is slow simulation. Each separate physical effect is modeled individually; some effects require transcendental function calculations or nested iterations. When all of these calculations are combined, the final modeling routine becomes extremely complex, requiring a very large number of computational operations to predict transistor behavior. The high cost of theoretical models has led to the development of an alternative strategy, which I will refer to as "empirical" modeling.

Empirical models are not rooted in solid-state physics; rather, they are mathematical expressions which provide a good curve-fit to the behavior of MOS transistors. Whereas theoretical models strive for accuracy at any cost, empirical models strive for acceptable accuracy at very low cost. Since individual transistors on the same physical wafer are subject to parameter variations, modeling accuracy beyond

a certain level is illusory.

Instead of separately accounting for each individual physical effect, empirical models often lump several effects together, and they occasionally ignore some effects entirely. Accuracy is thus traded against model evaluation speed, accepting engineering approximations in return for fast simulation. As R. C. Foss has said [19], "A ±30 percent representation of a 10 percent effect gives 3 percent accuracy."

Since their equations are not constrained to be physically motivated, empirical models can employ any convenient mathematical expressions which give reasonable accuracy. Simple polynomials are often used, because they can provide acceptable curve-fits while permitting fast evaluation. The coefficients of these polynomials are the "fudge factors" which are adjusted to obtain a good fit. Because the "fudge factors" are not physical constants of the fabrication process, their values are unpredictable, and must be extracted from measurements made on actual transistors. New, as-yet-unfabricated technologies cannot be simulated with empirical models, because no test transistors exist for parameter extraction.

A hybrid modeling scheme which combines the flexibility of theoretical models with the fast simulation speed of empirical models is presented in this thesis. Model parameters are (easily obtainable) physical fabrication constants, yet high speed empirical equations are implemented in the circuit simulator. This is achieved by "compiling" the physical constants for a theoretical model into the parameters necessary to drive an empirical model which is used by the circuit simulator.

The model parameter compilation process is analogous to a high-level computer language compiler. A slow translation program is invoked once, generating a fast object program which is subsequently invoked many times. In the MOS modeling domain, the "object program" is a set of parameters which will drive a fast empirical model (permitting fast circuit simulation). The "translation program" uses the input theoretical model parameters (the "source program") to simulate several test transistors, *mimicking the measurements one would make to perform a manual*

parameter extraction. These measurements are fed into a multiple-parameter nonlinear curve fitting program, which generates optimal (best-fit) parameters to drive the empirical model.

In some cases, it might be possible to analytically compute the empirical model parameters, given the theoretical model parameters (*e.g.*, when the empirical model is a subset of the theoretical model). However, a more general solution is desirable, which would permit compiling between any two arbitrarily selected models. Parameters could then be compiled for several different empirical models, to evaluate which one gives the best tradeoff between speed and accuracy. To allow these sorts of experiments, a curve-fitting program was constructed.

Of course, transistor modeling is only one part of the work performed by circuit simulators, so it is possible that a drastic speedup in transistor modeling might result in only slightly improved total simulation speed. Figure 1.1 shows the total time required to simulate the same circuit using two different MOS transistor models. One model (Berkeley) is theoretical, and the other model (Mosaid) is empirical. Data for this figure is taken from the SPICE 2G.5 simulator, running on the VAX[1] 11/780 computer under the V7 UNIX[2] operating system. For this particular circuit, simulation time was reduced by 35%, simply by switching to an empirical model.

The present modeling effort is concentrated on the steady-state (DC) current-voltage characteristics of MOS transistors. Although capacitances and their model parameters have been omitted for simplicity, there is no fundamental limitation which prevents the compilation technique from extracting them. Indeed, since most capacitance parameters do not interact with DC parameters, two separate compilations could be employed, thus reducing the dimensionality of the parameter-space and

[1]VAX is a registered trademark of Digital Equipment Corporation.
[2]UNIX is a registered trademark of Bell Telephone Laboratories.

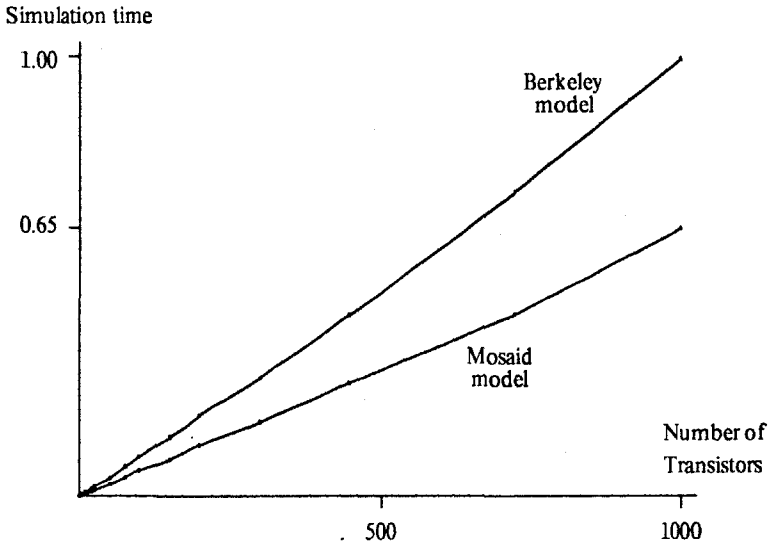

Figure 1.1 Circuit simulation time for different MOS models

improving compilation speed.

The present effort emphasizes the compilation aspects of transistor modeling; old, well-known models were used in the compilation experiments, rather than developing entirely new models. However, the technique presented here is a general one, allowing parameters for virtually any MOS model to be compiled into parameters for virtually any other model. (For reasons of time, this work concentrated on the Berkeley and Mosaid models). The same method could be applied, for example, to generate table entries for a totally table-driven empirical model [41], or to compute optimal coefficients for a cubic-spline interpolation scheme.

The use of numerical minimization for parameter extraction has applications besides the proposed compilation strategy. If an MOS technology has been in existence for some time, actual test transistors are available for measurement, and this data can be fed to the minimization routine. Optimal-fit parameters can then be extracted directly from the physical data [7, 39]. However, this technique does not

permit the designer to easily perturb the simulation (*e.g.*, to see the effects of over-etched polysilicon), because the empirical model parameters are not directly related to physical quantities.

This thesis is divided into five chapters. Chapter 1 discusses the idea of using a two-level modeling technique, and proposes the technique of constructing a parameter compiler. MOS models are the topic in chapter 2. The distinctions between theoretical and empirical models are discussed, and the two example models used in the parameter compilation experiments are described. Chapter 3 presents numerical algorithms for nonlinear minimization; they are subsequently used in the parameter compilation programs. Chapter 4 discusses the process of model parameter extraction. Manual extraction techniques are presented, and then fully automated procedures are described. Results obtained from prototype versions of model parameter compilers are presented, and the computational benefits of the compilation strategy are outlined. Chapter 5 presents some concluding remarks, along with suggestions for extending and improving this work.

Chapter 2: DC Models for MOS Transistors

2.1: Introduction

This chapter explores the problem of constructing a circuit model of the MOS field-effect transistor. The model is nothing more than a set of equations which predicts the device's current-voltage behavior. A familiar example of a model is Ohm's Law, which states that the current through a resistor is linearly proportional to the voltage across its terminals:

$$I = V \left(\frac{1}{R} \right) \qquad (2.1.1)$$

A simple, intuitive model of an idealized transistor is presented, to show the general form of model equations and to exhibit typical I-V characteristics. Model refinements are then introduced, which attempt to account for the non-ideal behavior observed in real transistors. As refinements are added, the complexity of model equations is increased, and the distinction between theoretical and empirical models becomes clearer. The equations for a typical theoretical model and for a typical empirical model are presented; they are subsequently used in the parameter compilation experiments described in chapter 4.

2.2: The MOS Field-Effect Transistor

A cutaway view of an n-channel MOS transistor is shown in Figure 2.1. The transistor has four terminals, called the Drain, Source, Gate, and Bulk (or substrate). The drain and source terminals are n-type diffused regions, while the bulk is p-type silicon; these pn junction diodes are reverse biased under normal operating conditions.

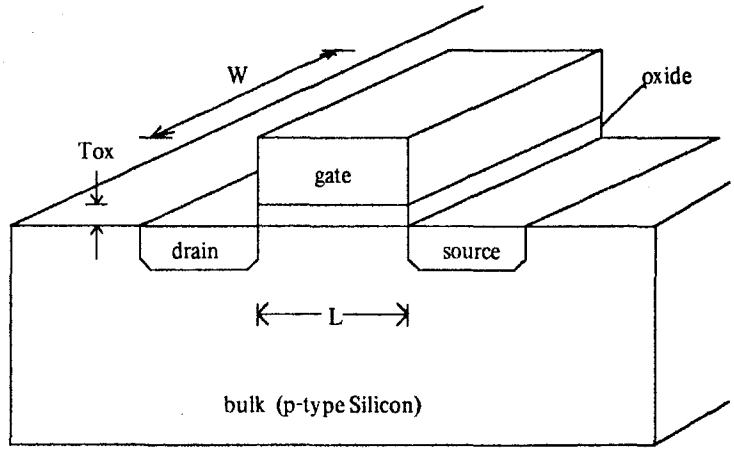

Figure 2.1 MOS Field Effect Transistor

The gate terminal is typically made of polycrystalline silicon, and is separated from the rest of the device structure by a thin insulating layer of silicon dioxide[1]. No DC current can flow into the gate terminal because of this insulator, so (unlike the bipolar junction transistor) the MOS transistor is a voltage-controlled device, as shown in the circuit model of Figure 2.2. Although the drain and source terminals are symmetric, a labeling convention is adopted such that $V_{DS} \geq 0$.

Current-voltage characteristics of MOS transistors can be (roughly) divided into three separate regions of operation, labeled in Figure 2.3 as the Triode, Saturation, and Cutoff regions. Transistors operating in the cutoff region have zero drain current I_{DS}; current begins to flow only when the gate-to-source voltage exceeds a threshold value called V_T. N-channel devices with $V_T \geq 0$ volts are often called "enhancement" transistors, while transistors with thresholds below 0 volts are called "depletion"

[1]Gate dielectric materials other than silicon dioxide are occasionally used in exotic devices.

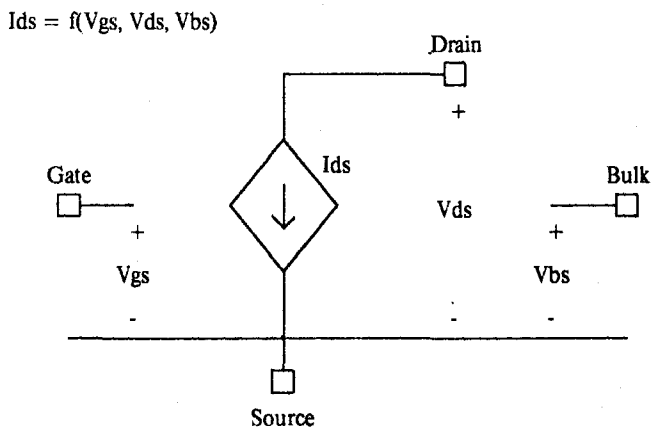

Figure 2.2 Circuit model of MOS transistor

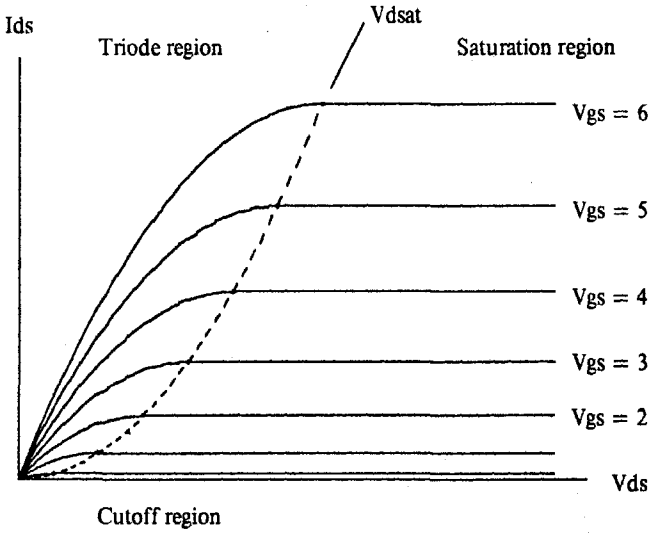

Figure 2.3 Typical current – voltage characteristics

devices.

In the triode region, drain current is an increasing function of drain voltage V_{DS}. However, as drain voltage is increased, the current eventually levels off ("saturates") and the transistor enters the saturation region. The transition from triode to saturation

occurs at a value of drain voltage known as V_{Dsat}. Figure 2.3 shows the locus of V_{Dsat} values.

The n-channel transistor depicted in Figure 2.1 is built on a p-type substrate; an n-type substrate (with p-type source and drain) can also be used, resulting in a p-channel transistor. P-channel devices operate exactly the same way that n-channel transistors do, except that the algebraic signs of all voltages and currents are reversed. Thus a p-channel, enhancement transistor has $V_T \leq 0$ volts, while a p-channel depletion device has $V_T > 0$.

2.3: A Simple First-Order Model

A very simple model of MOS transistor operation is developed in this section. The discussion will focus on n-channel devices, although p-channel transistors are readily modeled by a change of algebraic sign. Results arising from solid-state physics will be referenced, but used without derivation, in order to treat the transistor as a circuit-level lumped element. Many solid-state electrodynamic analyses of the MOS transistor are available in the literature [13, 21, 32, 47]; the interested reader is referred to them for more detail.

When a positive gate-to-bulk bias V_{GB} is applied to the MOS transistor, the induced electric field across the gate oxide attracts negatively-charged carriers (electrons) to the silicon dioxide surface, as shown in Figure 2.4. These electrons ionize the holes present at the surface, and begin to deplete the p-type bulk of majority carriers. If the bias voltage is made sufficiently large, the number of electrons at the surface exceeds the number of holes, and the surface is said to be *inverted*. So many electrons are available that the surface is effectively n-type, not p-type. The inversion layer at the surface forms an n-type "channel" from the (n-type) drain to the (n-type) source, allowing current to flow.

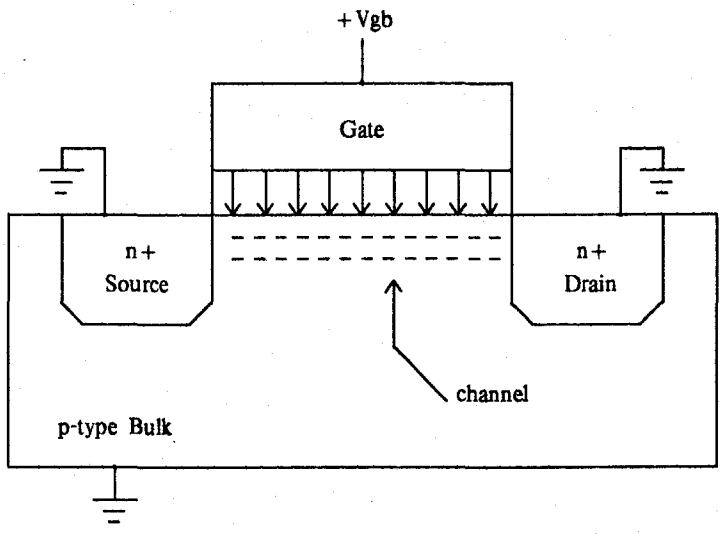

Figure 2.4 Field induced channel

If the source is shorted to the bulk ($V_{GS} = V_{GB}$), and the drain-to-source voltage is kept very small, the total charge Q_C in the channel region is given by[2]

$$Q_C = \frac{-\varepsilon_{ox}}{T_{ox}} (V_{GS} - V_T) WL \qquad (2.3.1)$$

Channel current is related to channel charge through the transit time T_{tr}

$$Q_C = -I_{DS} \; T_{tr} \qquad (2.3.2)$$

[2]This discussion is after Muller and Kamins [33], pp. 350-354.

Transit time is simply the channel length divided by the drift velocity

$$T_{tr} = \frac{L}{-\mu_n E} = \frac{L^2}{\mu_n V_{DS}} \tag{2.3.3}$$

The channel current is therefore

$$I_{DS} = \mu_n \frac{W}{L} \frac{\varepsilon_{ox}}{T_{ox}} (V_{GS} - V_T) V_{DS} \tag{2.3.4}$$

If the drain-to-source voltage is now increased, the channel potential is not constant, but increases from source to drain. This effect can be approximated by considering the entire channel to be at its "average" potential ($V_{GS} - (V_{DS}/2)$). This leads to a new channel charge Q_C

$$Q_C = \frac{-\varepsilon_{ox}}{T_{ox}} (V_{GS} - V_T - V_{DS}/2) W L \tag{2.3.5}$$

and a new drain current

$$I_{DS} = \mu_n \frac{W}{L} \frac{\varepsilon_{ox}}{T_{ox}} (V_{GS} - V_T - V_{DS}/2) V_{DS} \tag{2.3.6}$$

At constant V_{GS}, equation (2.3.6) predicts the drain current to be a parabolic function of drain voltage; see Figure 2.5. The maximum of the parabola occurs at $V_{DS} = (V_{GS} - V_T)$. Beyond this drain voltage, equation (2.3.6) predicts unreasonable behavior, since the incremental conductance (slope) is negative. In fact, the drain current saturates at this maximum value, becoming independent of V_{DS} due to a physical mechanism known as channel pinchoff [33].

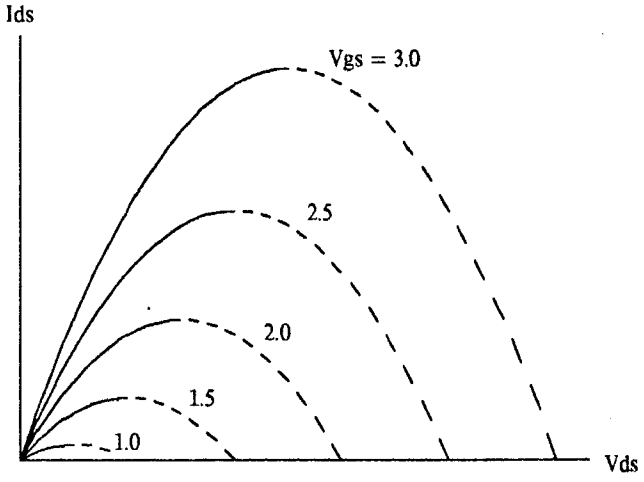

Figure 2.5 Idealized current – voltage characteristics

Equation (2.3.6) is valid in the triode region ($V_{DS} \leq V_{Dsat}$); the saturation voltage is given by

$$V_{Dsat} = V_{GS} - V_T \tag{2.3.7}$$

At drain voltages above V_{Dsat}, the transistor enters the saturation region, where the drain current is independent of V_{DS}:

$$I_{DS} = \mu_n \frac{W}{L} \frac{\varepsilon_{ox}}{2T_{ox}} (V_{GS} - V_T)^2 \tag{2.3.8}$$

The constants in the preceding equations (*e.g.*, μ_n, ε_{ox}, T_{ox}) are physical

parameters of the fabricated transistors.

An equivalent formulation would replace the leading term of (2.3.6) with a single constant "K", yielding the familiar Shichman-Hodges model [40]:

$$I_{DS} = \begin{cases} 0 & (a) \text{ if } V_{Dsat} < 0 \\ K \frac{W}{L} (V_{GS} - V_T - V_{DS}/2) V_{DS} & (b) \text{ if } V_{DS} \leq V_{Dsat} \\ \frac{K}{2} \frac{W}{L} (V_{GS} - V_T)^2 & (c) \text{ if } V_{DS} > V_{Dsat} \end{cases} \quad (2.3.9)$$

Equation (2.3.9) models all three regions of operation: (a) cutoff, (b) triode, and (c) saturation.

2.4: Sources of Error

In the first-order model of the preceding section, the source was assumed to be shorted to the bulk, and V_T was independent of bias voltage. Measurements on actual transistors show that V_T varies with the source-to-bulk voltage V_{SB}; see Figure 2.6. This phenomenon, known as the "body effect", is incorporated into the Shichman-Hodges model by introducing the empirical parameters V_{T0}, φ, and γ:

$$V_T = V_{T0} + \gamma \left[(\varphi + V_{SB})^{1/2} - (\varphi)^{1/2} \right] \quad (2.4.1)$$

A more serious source of error in the Shichman-Hodges model arises from the assumption that the channel voltage can be approximated by its average value $(V_{GS} - (V_{DS}/2))$. This approximation can be removed by writing an expression for the voltage drop across an infinitesimal length dy of the channel, at a distance y from the

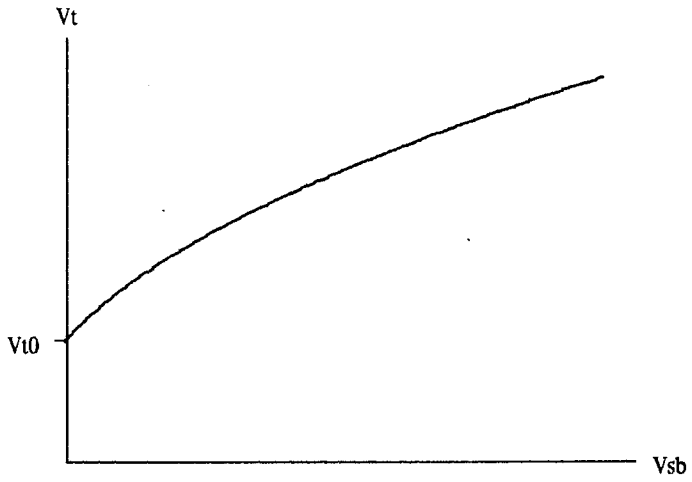

Figure 2.6 The body effect

source terminal. Manipulation yields

$$I_{DS}\, dy = -\mu_n W\, Q_C(V_C)\, dV_C \tag{2.4.2}$$

which can be integrated to give the drain current I_{DS} [33]:

$$I_{DS} = \mu_n \frac{W}{L} \{ C_{ox} (V_{GS} - V_{FB} - V_{DS}/2) V_{DS} - \frac{2}{3} (2\varepsilon_s q N_a)^{1/2} [(2\varphi_F + V_{DS} + V_{SB})^{3/2} - (2\varphi_F + V_{SB})^{3/2}] \} \tag{2.4.3}$$

Equation 2.4.3 is valid in the triode region; current is assumed to be constant in the saturation region. The transition voltage is given by

$$V_{Dsat} = V_{GS} - V_{FB} - \frac{\varepsilon_s q N_a}{C_{ox}^2}\{ [1 + \frac{2 C_{ox}^2}{\varepsilon_s q N_a} (V_{GS} - V_{FB} + V_{SB})]^{1/2} - 1\} \quad (2.4.4)$$

Notice that equation 2.4.3 explicitly contains terms for the "body effect" mentioned above, with coefficients that are physical constants of the fabrication process. It is a refinement of the simple theoretical model developed in section 2.3.

By an appeal to channel pinchoff it was assumed that the incremental output conductance ($\partial I_{DS} / \partial V_{DS}$) in the saturation region is zero. Actual transistors exhibit finite output conductance, which increases with increasing gate voltage. This effect, often called "channel length modulation", is shown in Figure 2.7. The empirical Shichman-Hodges model accounts for channel length modulation by including an output conductance parameter V_E, similar to the Early voltage of bipolar transistors:

$$I_{DS} = \frac{K}{2} \frac{W}{L} (V_{GS} - V_T)^2 [1 + \frac{V_{DS} - V_{Dsat}}{V_E}] \quad (2.4.5)$$

Theoretical models have also been developed to account for finite output conductance [20]; one such model will be shown in section 2.6.

2.5: Second Order Effects

As MOS fabrication technology improves, it becomes possible to build smaller transistors. When a transistor's dimensions are decreased, assumptions made in the derivation of the model equations (*e.g.*, that threshold voltage is independent of device size) become less valid. This section discusses several examples of such second-order effects.

Transistors with short channels ($L \leq 9\mu m$) or narrow channels ($W \leq 9\mu m$) have different threshold voltages than wide, long devices [1, 2, 8, 10, 48]; see Figure 2.8.

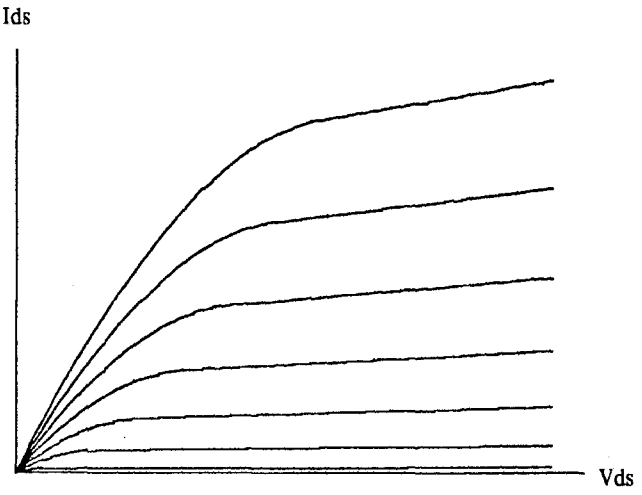

Figure 2.7 Channel length modulation

Short-channel thresholds are lower because the depletion regions around the source and drain partially ionize the channel. As channel length is decreased, this additional ionization becomes an appreciable fraction of the total channel charge. Narrow channel transistors, on the other hand, have higher thresholds than wide devices [4]; this results from field implants diffusing into the channel region and raising the average bulk doping density [45].

Another effect that becomes more pronounced in short channel devices is the nonlinear velocity-field relationship. For low electric fields, equation (2.3.3) is applicable, and carrier velocity is proportional to the transverse electric field ($E = -V_{DS}/L$). However, at high fields, velocity becomes limited by carrier interactions with lattice phonons and eventually reaches a maximum value. This effect limits the drain current (and hence the gain) of short channel transistors, as shown in Figure 2.9. Two transistors with identical (W/L) ratios are plotted; one has $L = 20$

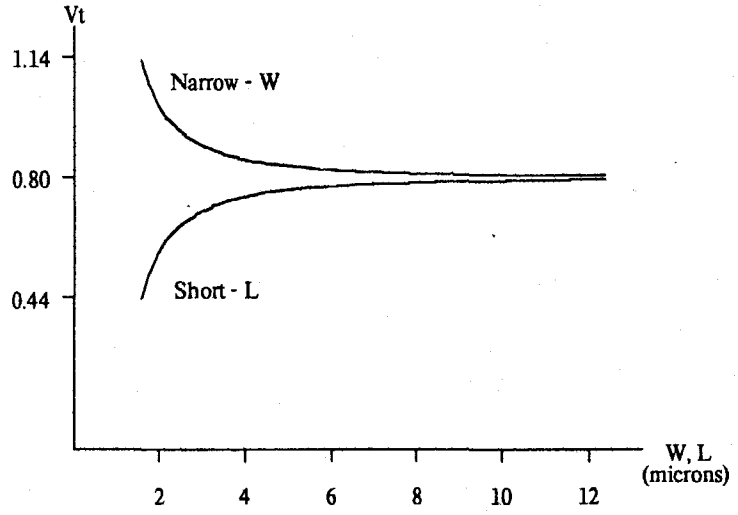

Figure 2.8 Narrow– and short–channel effects

microns (solid curves), and the other has L = 5 microns (broken curves).

In the derivation of equation (2.4.3), it was assumed that surface mobility is constant. Experimentally, mobility is found to be a function of the vertical electric field across the gate oxide. As gate voltage V_{GS} rises, mobility decreases, so the final current of an actual transistor will be less than that predicted by (2.4.3). Two different models for this effect will be presented in the following sections.

2.6: Berkeley Model

The example theoretical model, used for the parameter compilation experiments in chapter 4, is presented in this section. It was developed at the University of California at Berkeley, and is installed as the built-in model in the SPICE circuit simulator [44]. The symbol C_{ox} is used to represent the gate oxide capacitance per unit area ($= \varepsilon_{ox}/T_{ox}$).

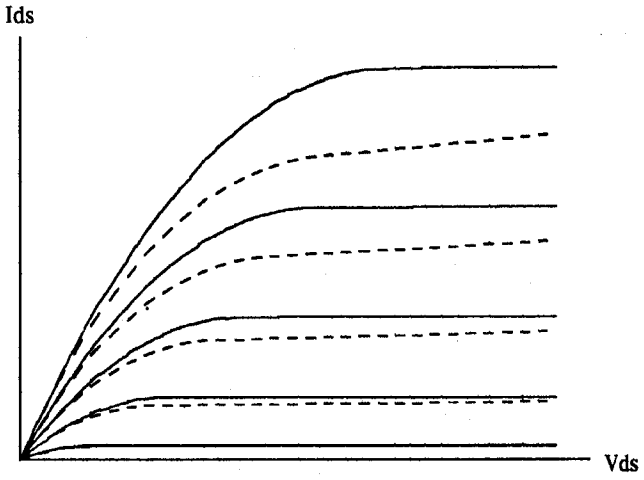

Figure 2.9 Velocity saturation

Threshold Voltage:

$$V_T = V_{BIN} + \gamma_S (2\varphi_F + V_{SB})^{1/2} \tag{2.6.1}$$

$$V_{BIN} = \varphi_{MS} - \frac{q\ Nss}{C_{ox}} + 2\varphi_F + (\eta - 1)(2\varphi_F + V_{SB}) \tag{2.6.2}$$

(Body effect coefficient):

$$\gamma_S = \frac{1}{C_{ox}} (2q\,\varepsilon_s\, NSUB)^{1/2} (1 - \alpha_S - \alpha_D) \tag{2.6.3}$$

$$\alpha_S = \frac{XJ}{2L_P} \left[\left(1 + 2\frac{W_S}{XJ}\right)^{1/2} - 1 \right] \tag{2.6.4}$$

$$\alpha_D = \frac{XJ}{2L_P} \left[\left(1 + 2\frac{W_D}{XJ}\right)^{1/2} - 1 \right] \tag{2.6.5}$$

(Depletion region width):

$$W_S = X_D \, (2\varphi_F + V_{SB})^{1/2} \tag{2.6.6}$$

$$W_D = X_D \, (2\varphi_F + V_{SB} + V_{DS})^{1/2} \tag{2.6.7}$$

$$X_D = \left(\frac{2\varepsilon_S}{q \, NSUB \, NEFF} \right)^{1/2} \tag{2.6.8}$$

$$\eta = 1 + \frac{\pi}{4} \frac{\varepsilon_S}{C_{ox} W} \tag{2.6.9}$$

Narrow channel effects are accounted for by equation (2.6.9), in which η increases as channel widths decrease. This serves to raise the built-in voltage (equation 2.6.2), increasing the threshold.

The input parameter *XJ* controls the modeling of short channel effects. Depletion regions around the source and drain serve to ionize carriers in the bulk and lower the threshold of short channel transistors. This is modeled using Dang's trapezoidal approach [10], where the width of the depletion regions at the source and drain are given by equations (2.6.6) and (2.6.7) respectively.

Surface Mobility:

$$E_x = \frac{C_{ox}}{\varepsilon_S} (V_{GS} - V_T) \tag{2.6.10}$$

$$\mu_{eff} = \begin{cases} \mu_0 \left(\dfrac{UCRIT}{E_x} \right)^{UEXP} & (a) \text{ if } E_x > UCRIT \\ \mu_0 & (b) \text{ if } E_x \leq UCRIT \end{cases} \tag{2.6.11}$$

Mobility remains constant, at its low-field value of μ_0, until the vertical electric field E_x exceeds *UCRIT*. Above *UCRIT*, mobility decreases exponentially according to equation (2.6.11a).

Drain Current:

$$V_x = \min (V_{DS}, V_{Dsat}) \tag{2.6.12}$$

$$I_{DS} = \begin{array}{l} \mu_n \frac{W}{L_{eff}} \{ C_{ox} (V_{GS} - V_{BIN} - \eta V_x/2) V_x - \\ \frac{2}{3} \gamma_S [(2\varphi_F + V_x + V_{SB})^{3/2} - (2\varphi_F + V_{SB})^{3/2}] \} \end{array} \tag{2.6.13}$$

This formula is essentially identical to the theoretical prediction given in equation (2.4.3), except for the inclusion of short channel effects (via γ_S) and narrow channel effects (via η). When V_{DS} exceeds V_{Dsat}, V_x becomes constant and the drain current saturates.

Saturation Voltage:

$$VMAX = \frac{I_{Dsat}}{W C_{ox} L_{eff} (V_{GS} - V_{BIN} - \eta V_{Dsat} - \gamma_S [V_{Dsat} + 2\varphi_F + V_{SB}]^{3/2})} \tag{2.6.14}$$

$$L_{eff} = L_P + \frac{X_D^2 VMAX}{2\mu_{eff}} - X_D [\frac{X_D VMAX}{2\mu_{eff}} + (V_{DS} - V_{Dsat})]^{1/2} \tag{2.6.15}$$

$$L_P = L - 2L_D \tag{2.6.16}$$

Expressions (2.6.14) and (2.6.15) are nonlinear equations in the two unknowns L_{eff} and V_{Dsat} [44]. Velocity saturation (Figure 2.9) is accounted for by the parameter *VMAX*. Finite output conductance in the saturation region is introduced through the channel length modulation term L_{eff}.

The input parameters, most of which are physical constants of the fabrication process, are summarized in Table 2.10. (Only "*NEFF*" is an empirical adjustment). The actual Berkeley model implemented in SPICE is slightly more complicated, since it includes prethreshold conduction. However, this feature was not used in the compilation experiments, so the modeling equations are omitted here.

The model program is relatively expensive to evaluate; including the calculations of the small-signal conductances, it requires 257 double-precision floating multiply operations, 125 divisions, 20 square roots, 6 exponentiation operations, and 4 logarithm evaluations.

2.7: Mosaid Model

The Mosaid model was developed to provide reasonable modeling accuracy while permitting high speed simulation. It is based primarily on the Shichman-Hodges equations (2.3.9), which are very fast to evaluate. Empirical parameters are introduced to account for short channel effects, while avoiding nested iterations (such as equations 2.6.14 and 2.6.15).

Effective Gate Length: $$L_{eff} = L - 2L_D \tag{2.7.1}$$

Parameter	Units	Description
N_{ss}	Meters^{-2}	Density of surface states
$2\varphi_F$	Volts	Surface potential
T_{ox}	Meters	Gate oxide thickness
NSUB	Meters^{-3}	Substrate doping density
XJ	Meters	Junction depth
L_D	Meters	Lateral diffusion & poly over-etch
NEFF	—	Total channel charge coefficient
μ_0	Meters^2/Volt-Second	Low-field surface mobility
UCRIT	Volts/Meter	Critical field for mobility degradation
UEXP	—	Exponent of mobility reduction
VMAX	Meters/Second	Scattering-limited carrier velocity

Table 2.10 Berkeley model parameters

Threshold Voltage:

$$V_T = V_{T0} + \gamma \left[(\varphi + V_{SB})^{1/2} - (\varphi)^{1/2} \right] - K_D \frac{V_{DS}}{L_{eff}} \qquad (2.7.2)$$

The Shichman-Hodges model (Equation 2.4.1) is used to account for the body effect, with the addition of short channel threshold reduction through the parameter

K_D. Narrow channel threshold effects (Figure 2.8) are not modeled.

Effective Gate Drive:
$$V_e = V_{GS} - V_T \tag{2.7.3}$$

Effective Gain Factor:
$$K_{eff} = \frac{KP}{1 + \theta V_e} \tag{2.7.4}$$

In this formulation, effective mobility is decreased even at low vertical electric fields, unlike the two-region model implemented in the Berkeley model (equation 2.6.11). This equation has been found to yield acceptable accuracy while permitting fast evaluation [9, 10, 47].

Drain Current:

$$I_{DS} = \begin{cases} 0 & (a)\ \text{if} V_e \leq 0 \\ \dfrac{K_{eff} W}{L_{eff} + 2K_{eff} R_S V_{DS}} (2V_e - V_{DS}) V_{DS} & (b)\ \text{if} V_{DS} < V_{Dsat} \\ \dfrac{K_{eff} W}{L_{eff} - C_M (V_{DS} - V_{Dsat})} V_{Dsat}^2 & (c)\ \text{if} V_{DS} \geq V_{Dsat} \end{cases} \tag{2.7.5}$$

Saturation Voltage:
$$V_{Dsat} = \frac{2V_e}{1 + [1 + (4K_{eff} R_S V_e / L_{eff})]^{1/2}} \tag{2.7.6}$$

Velocity saturation effects (Figure 2.9) are modeled by postulating a feedback term R with the dimensions of resistance, such that the gate drive V_{ee} is reduced as drain current increases:

$$V_{ee} = V_{GS} - V_T - I_{DS}R$$

To make the R term geometry-independent, it is replaced by R_S/W in the Mosaid model. The reduced gate drive is substituted into the Shichman-Hodges model (equation 2.3.9b), giving the drain current (equation 2.7.5b). [KP in the Mosaid model is equal to $K/2$ in the Shichman-Hodges model].

The saturation voltage V_{Dsat} is found by setting $\partial I_{DS}/\partial V_{DS}$ equal to zero, which gives equation (2.7.6)

Channel Length Modulation:
$$\alpha = \frac{V_{Dsat}}{8} \tag{2.7.8}$$

$$C_M = C_{M0}\left[1 - e^{-\alpha (V_{DS} - V_{Dsat})}\right] \tag{2.7.9}$$

The original Mosaid model used a constant value of C_{M0} in 2.7.5(c); this gives rise to a slope-discontinuity at V_{Dsat}. In the linear region the slope ($\partial I_{DS}/\partial V_{DS}$) tends toward zero at V_{Dsat}, while in the saturation region the slope remains finite at V_{Dsat}. Since the SPICE circuit simulator uses first derivatives in its Newton-Rhapson computations, a discontinuous slope can cause simulation nonconvergence, so the fix described by equations (2.7.8) and (2.7.9) was inserted. Parameters for the Mosaid model are summarized in Table 2.11.

The Mosaid model is comparatively inexpensive to evaluate; it requires only 36 double-precision floating multiply operations, 12 divisions, 6 square roots, and 2 exponentiation calculations.

Parameter	Units	Description
L_D	Meters	Lateral diffusion & poly over-etch
V_{T0}	Volts	Zero-bias threshold
γ	$Volts^{1/2}$	Body effect coefficient
φ	Volts	Surface potential
K_D	Meters	Short channel threshold coefficient
KP	$Amps/Volts^2$	Gain constant
θ	1/Volts	Mobility degradation coefficient
R_S	Ohm - Meters	Velocity saturation feedback constant
C_{M0}	Meters / Volt	Channel length modulation constant

Table 2.11 Mosaid model parameters

Chapter 3: Minimization Algorithms

After a mathematical model of transistor behavior is selected, parameter values must be determined which give the best fit to a particular set of measurements. Serious modeling errors can occur if the parameter values are poorly chosen. To insure that the best possible modeling accuracy is maintained across the entire range of transistor operation, it is desirable to create a set of "optimally good" parameter values.

Let P be an n-vector such that P_k is the value of the k-th model parameter. Suppose there exists a function $f: R^n \rightarrow R^1$ such that $f(P)$ is a measure of the modeling error incurred when the parameters P are used. Then the optimum parameter values exist at the point P where $f(P)$ is smallest.

The problem of finding the minimum value of a function has been extensively studied [12, 16, 18, 30]. This chapter presents five of the better-known minimization algorithms. Together with a suitable function $f(P)$, they can be used to construct programs for compiling optimal sets of model parameters. Detailed derivations of the individual algorithms are not given, nor are convergence properties proven, since the emphasis is on their application and not their design. In each case, the original papers are referenced for the reader who wishes to study the algorithms in greater detail.

It should be noted that these algorithms are designed to find *local* minima; no computationally tractable algorithm is known for finding the global minimum of an arbitrary function.

3.1: Notation

To simplify the discussion of the algorithms, a common notation has been used throughout the chapter. The function $f(x)$: $R^n \rightarrow R^1$ is to be minimized, and the n-vector x_j is the starting point for the jth iteration. A displacement Δx_j is made from the present point x_j to the next point x_{j+1}:

$$x_{j+1} = x_j + \Delta x_j \tag{3.1.1}$$

The displacement can be written in the form

$$\Delta x_j = -\alpha_j p_j \tag{3.1.2}$$

where p_j, an n-vector, denotes the search direction and α_j, a scalar, is the stepsize.

In general, the symbol

$$g = \nabla f(x) \tag{3.1.3}$$

will be used for the gradient of f, where g is an n-vector. Also,

$$G = \nabla^2 f(x) \tag{3.1.4}$$

is the $n \times n$ matrix of second partial derivatives. For scalar functions, the symbol $f^{(k)}(x)$ will be used to indicate the k-th derivative of f, evaluated at x.

3.2: Hooke and Jeeves Algorithm

This algorithm is based on the extremely simple idea of varying one coordinate at a time. From an initial point x_{00}, a step of length δ is taken along coordinate 1. f is evaluated at (x_{00}), $(x_{00} + \delta e_1)$, and $(x_{00} - \delta e_1)$, where e_1 is a unit vector along

coordinate 1. Whichever of these three points gives the lowest value of f is then named x_{01}, and the process is repeated for all n unit vectors, finally producing x_{0n} which completes iteration 0.

When a displacement is made that reduces the function value, further steps along the same direction will probably continue to decrease f. The Hooke and Jeeves algorithm [26] attempts to exploit this property, using a successful direction of descent as a first try for further exploration. For this reason, the algorithm is also called "pattern search" in the literature. Instead of starting iteration 1 at the endpoint of iteration 0, the displacement vector is doubled in length:

$$x_{10} = x_{00} + 2(x_{0n} - x_{00}) = 2x_{0n} - x_{00} \tag{3.2.1}$$

The algorithm's progress on minimizing a function of two variables is shown in Figure 3.1.

As the algorithm proceeds toward a minimum, the stepsize δ eventually becomes too large and must be reduced. It is replaced when an iteration fails to reduce the function value: $f(x_{jn}) \geq f(x_{j0})$. A simple geometric formula is used:

$$\delta^* = \rho\,\delta \qquad (\rho < 1) \tag{3.2.2}$$

The algorithm terminates when the value of δ is reduced below some minimum stepsize ε_{HJ}. The implementation of this algorithm described in chapter 4 uses

$$\begin{aligned} \varepsilon_{HJ} &= 10^{-4} \\ \delta_0 &= 0.1 \\ \rho &= 0.4 \end{aligned} \tag{3.2.3}$$

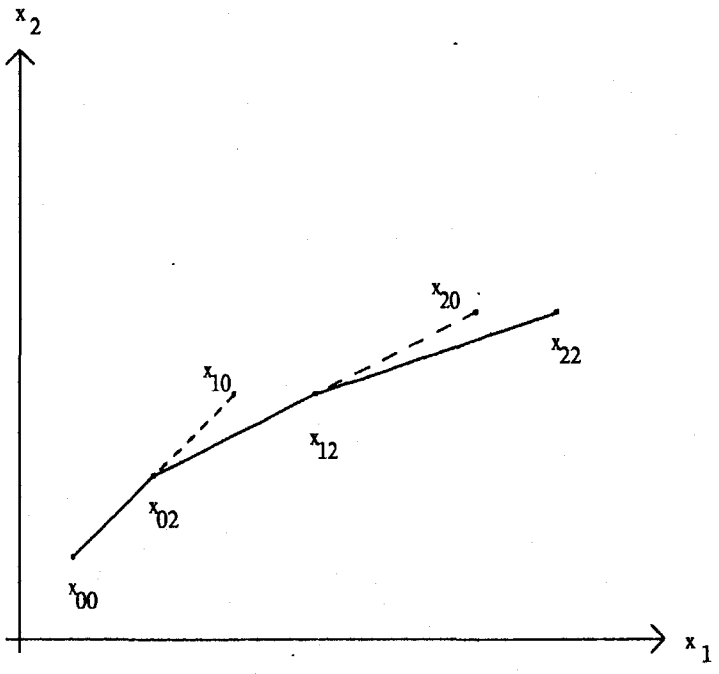

Figure 3.1 Hooke and Jeeves algorithm

3.3: Simplex Algorithm

This algorithm is based on topological generalization of a triangle[1]. Just as a triangle is a two-dimensional object having three vertices, a *simplex* is an n-dimensional object having $n + 1$ vertices. A tetrahedron, for example, is a simplex in 3-dimensional space.

From a starting simplex of $n + 1$ points, the point at which f is highest is reflected through the hyperplane formed by the other n points. If the function value at the new point is lower, the simplex has been improved and the procedure can be repeated. During the minimization procedure the simplex changes in size and shape, adapting to

[1]This algorithm for function minimization should not be confused with Dantzig's "simplex algorithm" for linear programming.

the local contour of f. The algorithm is given below using the notation of Fidler and Nightingale [14]:

x_H is the vertex which gives the highest value of $f(x)$
x_S is the vertex which gives the second highest value of $f(x)$
x_L is the vertex which gives the lowest value of $f(x)$
x_0 is the centroid of all vertices x_k other than x_H:

$$x_0 = \frac{1}{n} \Sigma x_k \qquad (k \neq H) \tag{3.3.1}$$

Three different operations may be performed on the simplex.

(1) The vertex x_H may be *reflected* through the centroid to give x_R:

$$x_R = (1 + \alpha) x_0 - \alpha x_H \tag{3.3.2}$$

(2) The simplex may be *expanded* along a favorable direction:

$$x_E = \gamma x_R + (1 - \gamma) x_0 \tag{3.3.3}$$

(3) The simplex may be *contracted*:

$$x_C = \beta x_R + (1 - \beta) x_0 \tag{3.3.4}$$

These operations are used to find the minimum value of f in Algorithm 3.2 [14, 35].

The algorithm is terminated when the standard error falls below a given threshold ε_S:

$$\frac{\Sigma[f(x_j) - \bar{f}]^2}{n} < \varepsilon_S^2 \tag{3.3.5}$$

Since the vertices of the initial simplex may be widely separated, the algorithm essentially has $n + 1$ independent starting points. The impact of an especially poor starting point is therefore reduced.

[1] The distinguished vertices x_H, x_S, x_L, and x_0 are computed.

[2] x_R is computed according to (3.3.2).

[3] If $f(x_R) \leq f(x_H)$, but $f(x_R) \geq f(x_L)$, an improvement has been made, so x_H is replaced by x_R, and another iteration can begin at step [1].

[4] If $f(x_R) < f(x_L)$, the reflected point has a lower function value than all other vertices. This direction appears to be favorable, so the expansion (3.3.3) is attempted. x_H is replaced by whichever of x_R or x_E gives the lower function value, followed by a return to step [1].

[5] If $f(x_R) < f(x_H)$, but $f(x_R) > f(x_S)$, only a minor improvement has been made, and a contraction is performed using equation (3.3.4) in case the reflection has overshot a better point. x_H is replaced by whichever of x_R or x_C gives the lower function value, and the algorithm returns to step [1].

[6] If $f(x_R) > f(x_H)$, no improvement has been made by reflection. The implication is that the minimum probably lies within the simplex, so the simplex is shrunk about the lowest point x_L: $x_k = (x_k + x_L)/2$

Algorithm 3.2 The simplex method

Chapter 4 describes an implementation of the simplex algorithm which uses

$$\begin{aligned} \epsilon_S &= 10^{-11} \\ \alpha &= 1.0 \\ \beta &= 0.5 \\ \gamma &= 2.0 \end{aligned} \qquad (3.3.6)$$

3.4: Newton's Iteration

The Hooke and Jeeves and simplex algorithms operate on function values only; no derivative information is required. If the first and second derivatives of f are available, more powerful algorithms may be applied, giving faster and more reliable convergence. These algorithms use a first-order Taylor series expansion of f about x_1. In the one-variable case,

$$f(x_2) \approx f(x_1) + [x_2 - x_1]\, f^{(1)}(x_1) \tag{3.4.1}$$

A point x_2 is not a minimum unless

$$\frac{df}{dx}(x_2) = 0 \tag{3.4.2}$$

Maximum points and inflection points also satisfy (3.4.2); local explorations about x_2 are usually performed to verify that it is indeed a minimum.

An approximate solution to (3.4.2) can be found by differentiating (3.4.1) and setting the result equal to zero.

$$0 = \frac{df}{dx}(x_1) + [x_2 - x_1]^2 \frac{d^2f}{dx^2}(x_1) \tag{3.4.3}$$

Equation (3.4.3) has the solution

$$x_2 = x_1 - \frac{f^{(1)}(x_1)}{f^{(2)}(x_1)} \tag{3.4.4}$$

which is known as *Newton's iteration.*

In the multivariable case $f: R^n \rightarrow R^1$, the first-order Taylor approximation is

$$f(x_2) \approxeq f(x_1) + g^T[x_2 - x_1] \tag{3.4.5}$$

where g is the gradient of f at x_1. The derivative of equation (3.4.5) is set equal to zero, giving a minimum at

$$x_2 = x_1 - G^{-1} g \tag{3.4.6}$$

where G is the matrix of second partial derivatives at x_1. In the terminology of (3.1.2), the search direction p_1 is given by $(G^{-1} g)$, and the stepsize α_1 is equal to one.

Equation (3.4.6) is used to calculate the displacement $\Delta x_j = -G_j^{-1} g_j$, which provides the starting point for iteration $(j + 1)$. Iterations are performed in this fashion until convergence is achieved. The algorithm is considered to have converged when the gradient is approximately zero (see equation 3.4.2):

$$g^T g \leq \varepsilon_{NI} \tag{3.4.7}$$

Newton's iteration has the desirable property of *quadratic convergence*: if f is a quadratic function, the minimum point is achieved in a number of iterations at most equal to the number of variables [27]. Since an arbitrary function can be closely approximated by a quadratic over a small interval, the algorithm will converge equally rapidly on arbitrary functions, providing the starting point is close to the minimum. This property assures rapid convergence in the final stage of computation.

Newton's iteration (3.4.6) has limited usefulness as a practical minimization algorithm, because it requires the user to supply formulae for calculating the n first derivatives in the vector g, and the (n^2) second derivatives in the matrix G. Frequently

this is inconvenient (or impossible), as for example when the number of coordinates n is large. Variations of the basic Newton iteration have been developed which do not require second derivatives; they are much more useful than the original formula (3.4.6). These *quasi-Newton* algorithms use approximations of G or G^{-1} to eliminate the need for second derivatives.

3.5: Quasi-Newton Algorithms

Huang [27] has presented a class of quadratically convergent algorithms which replace Newton's iteration (3.4.6) by the approximation

$$p_j = H_j g_j, \qquad \Delta x_j = -\alpha_j p_j \tag{3.5.1}$$

where the $n \times n$ matrix H characterizes a particular algorithm. (In all cases, H is computed without using second derivatives). The initial matrix H_0 is taken to be the identity matrix. Several well-known and highly successful minimization algorithms are members of Huang's class, including the conjugate-gradient algorithms and the variable metric algorithms.

Once the search direction p_j is known, the stepsize α_j must be chosen. For good progress toward the minimum of f, the value of α_j should be selected which minimizes the *scalar* function $u(\alpha)$:

$$u(\alpha) = f(x_j - \alpha p_j) \tag{3.5.2}$$

Very efficient algorithms are known for univariate minimization, so the optimum α can quickly be found. This determines the displacement Δx_j and hence the new point x_{j+1}, and another iteration can begin. The algorithm terminates when the gradient is sufficiently small, similar to the Newton termination criterion (3.4.7).

3.6: Conjugate-Gradient Algorithm

A particularly appealing member of Huang's class is the conjugate-gradient algorithm, studied by Polak and Ribiere [37]. It uses the approximation

$$H_j = I + \frac{p_{j-1}(g_{j-1} - g_j)^T}{g_{j-1}^T g_{j-1}} \tag{3.6.1}$$

where I is the identity matrix. Substitution into (3.5.1) yields

$$p_j = g_j + \gamma p_{j-1} \tag{3.6.2}$$

where the scalar γ is given by

$$\gamma = \frac{(g_{j-1} - g_j)^T g_j}{g_{j-1}^T g_{j-1}} \tag{3.6.3}$$

This substitution shows that the matrix H_j need not be stored directly; only the vectors p_{j-1}, p_j, g_{j-1}, and g_j are required. The Polak-Ribiere approach is therefore attractive for very large problems, where the storage cost of an $n \times n$ H-matrix would be prohibitive.

Equation (3.6.2) gives the search direction p_j, and the steplength α_j is computed by univariate minimization. Iterations proceed in this fashion until the gradient vector approaches zero:

$$g^T g \leq \epsilon_{CG} \quad (= 10^{-10}) \tag{3.6.4}$$

3.7: Fletcher's Switching-Policy Algorithm

This algorithm combines the well known Davidon-Fletcher-Powell [17] and Broyden-Fletcher-Goldfarb-Shanno [23] variable metric procedures, giving a hybrid algorithm which is superior to each [15].

The DFP algorithm is a member of Huang's class, characterized by

$$H_j = H_{j-1} + \frac{\delta\delta^T}{\delta^T\gamma} - \frac{H_{j-1}\gamma\gamma^T H_{j-1}}{\gamma^T H_{j-1}\gamma} \tag{3.7.1}$$

where

$$\begin{aligned} \delta &= x_j - x_{j-1} \\ \gamma &= g_j - g_{j-1} \end{aligned} \tag{3.7.2}$$

The BFGS algorithm is also a member of Huang's class:

$$H_j = H_{j-1} - \frac{\delta\gamma^T H_{j-1}}{\delta^T\gamma} - \frac{H_{j-1}\gamma\delta^T}{\delta^T\gamma} + \left[1 + \frac{\gamma^T H_{j-1}\gamma}{\delta^T\gamma}\right] \frac{\delta\delta^T}{\delta^T\gamma} \tag{3.7.3}$$

Fletcher's switching-policy algorithm uses either (3.7.1) or (3.7.3), depending on the outcome of a test: if

$$\delta^T\gamma \geq \gamma^T H_{j-1}\gamma \tag{3.7.4}$$

then the BFGS update formula (3.7.3) is used for iteration j. Otherwise, the DFP update (3.7.1) is used. Fletcher has presented performance data indicating that this hybrid algorithm is faster than either of its component parts. The hybrid approach, he

claims, avoids near-singularity in the H matrix, making the algorithm more stable [15].

If the H matrix is not positive definite, then stability and convergence to a minimum are much less likely. Fletcher's algorithm contains a safety test which prevents H from becoming nonpositive definite. Whenever

$$\delta^T \gamma \leq \varepsilon_1 \quad (= 10^{-16}) \tag{3.7.5}$$

the algorithm is "restarted" by setting H_j to the identity matrix. The algorithm is terminated when the gradient becomes sufficiently small:

$$g^T g \leq \varepsilon_F \quad (= 10^{-10}) \tag{3.6.4}$$

3.8: Line Searches

After the search direction p is determined, the line $(x - \alpha p)$ is searched to find the optimum value of α. In this section, two methods are discussed for performing a univariate search. Both algorithms proceed by finding an interval (a, b) which is known to contain the minimum point x. The size $|b - a|$ of this "bracket" is then decreased repeatedly until a minimum is found.

A simple criterion may be applied to find an initial bracket: at least one local minimum exists in the interval (a, b) if there is a point c within the interval, such that

$$(a \leq c \leq b) \quad \text{and} \quad [f(c) \leq f(a)] \quad \text{and} \quad [f(c) \leq f(b)] \tag{3.8.1}$$

A bracket satisfying (3.8.1) may be easily found using the algorithm of Davies, Swann, and Campey [6]; the procedure is illustrated in Figure 3.3. From the initial point $\alpha = 0$, a step of size δ is taken in the $+\alpha$ direction. Stepsizes are repeatedly doubled until

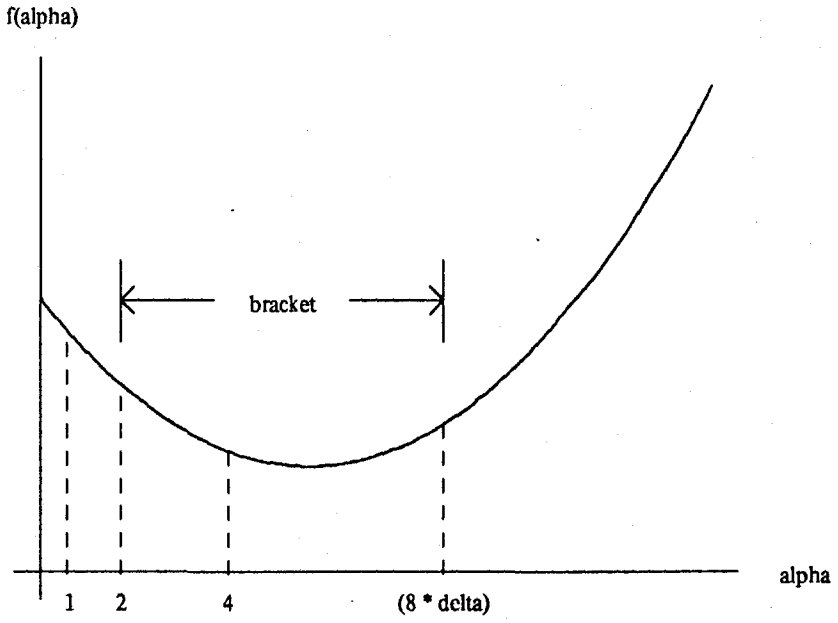

Figure 3.3 Establishing an initial bracket

the minimum is overshot and condition (3.8.1) is met, establishing the first bracket.

Once the initial bracket is found, a simple algorithm for converging to a minimum is the Golden Section search, illustrated in Figure 3.4. New points c and d are selected, which divide the original interval into three pieces: (a, c), (c, d), and (d, b). The function f is evaluated at c and d, and the reduced interval is given by

$$\begin{aligned} (a^*, b^*) &= (a, d) \qquad \text{if } f(c) \leq f(d) \\ (a^*, b^*) &= (c, b) \qquad \text{if } f(c) > f(d) \end{aligned} \tag{3.8.2}$$

The size of the bracket has been reduced by the fraction ad/ab (or cb/ab). To keep the algorithm unbiased, these fractions should be identical. Choosing c and d such that $cb = ad$ insures that the bracket will shrink by the same amount on either side.

Two function evaluations were used in the simplified iteration described above (at c and d). The Golden Section algorithm actually uses only one evaluation per

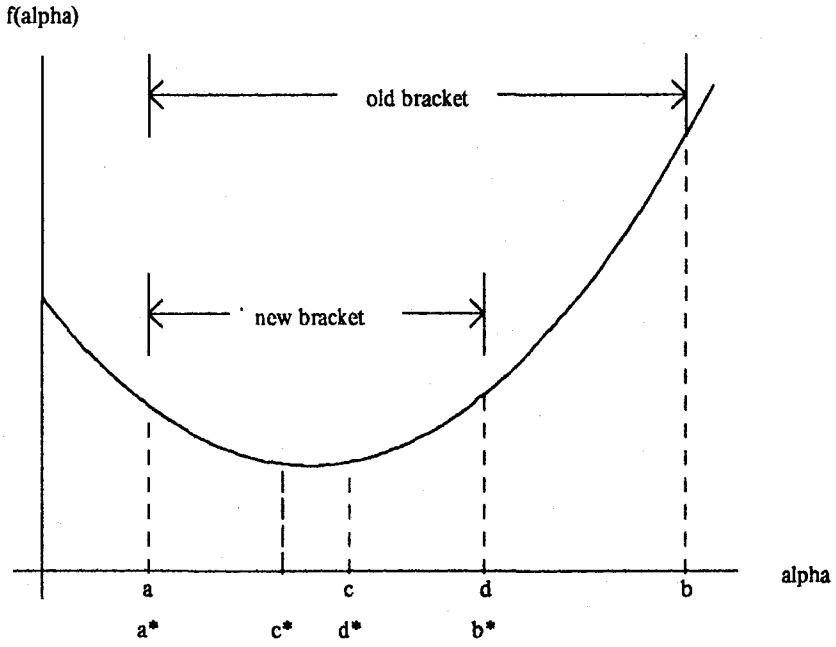

Figure 3.4 Golden Section search

iteration, through careful selection of the points c and d. If the new bracket is, for example, (a, d), then the old point c lies within the bracket, and $f(c)$ has already been evaluated. The point c should therefore be reused, so that only one new point needs to be evaluated in the next iteration.

If the ratio ad/ab is equal to the ratio ac/ad, then the point c can be reused and the resulting partition of the bracket will be unbiased, as shown in Figure 3.4. Normalizing to $ab = 1$, let $y = ad$ so that $(1 - y) = ac$. Then

$$\frac{y}{1} = \frac{1 - y}{y} \qquad (3.8.3)$$

This equation produces the Golden Section ratio used in Greek architecture (hence the algorithm's name). The solution is

$$y = \frac{(5)^{1/2} - 1}{2} = 0.61803 \qquad (3.8.4)$$

Two function evaluations are performed for the first bracket. Each subsequent bracket uses equation (3.8.4) to compute one new point, so only one function evaluation is required per iteration. After n iterations, the bracket has shrunk to $(0.61803)^n$ of its original size. The search is completed when the bracket has been reduced to a sufficiently small size

$$|b - a| \leq \varepsilon_2 \quad (= 10^{-15}), \qquad \text{or}$$
$$\frac{|b - a|}{b} \leq \varepsilon_3 \quad (= 10^{-3}) \qquad (3.8.5)$$

Table 3.5 compares the Golden Section algorithm to a simple binary search. On each iteration, binary search halves the bracket, but two function evaluations are used. The table shows the number of function evaluations required to shrink an initial bracket of size $|b - a| = 1$ down to various final sizes; binary search takes approximately 1.4 times as many iterations as the Golden Section search.

Line Search	Final bracket size					
Algorithm	1E-1	1E-2	1E-3	1E-4	1E-5	1E-6
Binary search	8	14	20	28	34	40
Golden Section	5	10	15	20	24	29

Table 3.5 Function evaluations required for bracket reduction

Another popular line search algorithm uses a quadratic interpolation scheme for finding the minimum [22]. Given an initial bracket (a, b) and a point c within the bracket, a quadratic is fitted to these three points. The minimum of the quadratic is given by [38]:

$$d = \frac{1}{2} \frac{[b^2 - c^2]f(a) + [c^2 - a^2]f(b) + [a^2 - b^2]f(c)}{[b - c]f(a) + [c - a]f(b) + [a - b]f(c)} \qquad (3.8.6)$$

A new bracket is then formed, taking the three lowest points found so far (which still maintain a bracket). Another interpolation is computed, and the process is repeated. Near the minimum, a quadratic interpolation will be very close to any arbitrary function, and the final stages of convergence will be very fast.

After evaluating both line search algorithms, the Golden Section method was selected for use in the minimization implementations described in chapter 4. When an initial bracket contained several minima (a common occurrence), Golden Section appeared to converge more rapidly than quadratic fitting; presumably this is because a quadratic is a poor approximation to a function with more than one minimum.

3.9: Numerical Differentiation

In many cases, analytic derivatives of $f(x)$ are not available, so they must be computed numerically. For example, the derivative $\partial I_{DS} / \partial VMAX$ for the Berkeley model does not exist in closed form (see equation 2.6.14). A finite difference formula can be used to estimate the derivatives:

$$\frac{\partial f}{\partial x_j} = \frac{f(x + \delta e_j) - f(x)}{\delta} \qquad (3.9.1)$$

If $f(x)$ is known, this forward-difference formula requires only one additional function evaluation to compute the derivative. A central-difference approach would need to

evaluate f at both $(x + \delta e_j)$ and $(x - \delta e_j)$; this is too expensive for use in a minimization algorithm. The size δ of the finite interval is chosen to reflect the scale of the component x_j:

$$\delta = \max (h\, x_j ,\ \delta^*) \tag{3.9.2}$$

where

$$\begin{aligned} h &= 2^{-16} \\ \delta^* &= 10^{-18} \end{aligned} \tag{3.9.3}$$

3.10: Marquardt's Algorithm

Suppose a physical process is modeled by the equation

$$y = M(V,x) \tag{3.10.1}$$

where y is the dependent variable, $V = (V_1, V_2, ..., V_k)$ are the independent variables, and $x = (x_1, x_2, ..., x_n)$ are adjustable parameters of the mathematical model M. A set of m experiments is made, obtaining values of y for different values of V. The problem is to find those values of x which give the experimental data the best fit to (3.10.1).

The best fit in the least-squares sense is obtained by defining the m residuals:

$$R_i(x) = M(V_i, x) - y_i \tag{3.10.2}$$

and minimizing f, the sum of squares of these residuals [28]:

$$f(x) = \Sigma R_i(x)^2 = R(x)^T R(x) \tag{3.10.3}$$

Special purpose algorithms for minimizing sum-of-squares functions such as (3.10.3) have been developed; the most popular of these methods is due to Marquardt [31].

Newton's iteration can be applied directly to (3.10.3), giving

$$G(x)\,\Delta x = -\,g(x) \tag{3.10.4}$$

The gradient $g(x)$ can be expressed in terms of the vector R of residuals:

$$g(x) = 2J(x)^T R(x) \tag{3.10.5}$$

where $J(x)$ is the $m \times n$ Jacobian matrix with

$$J_{ij}(x) = \frac{\partial R_i(x)}{\partial x_j} \tag{3.10.6}$$

The matrix G of second derivatives is given by

$$G(x) = 2J(x)^T J(x) + 2\,\Sigma \nabla^2 R_i(x)\, R_i(x) \tag{3.10.7}$$

where $\nabla^2 R_i(x)$ is the second derivative matrix of $R_i(x)$ [28].

Equation (3.10.7) shows that a substantial part of the matrix G is obtained by using only first derivatives. Near the solution, if the residuals are small or almost linear, then the second term of (3.10.7) is negligible. This leads to the approximation

$$G(x) \approxeq 2J(x)^T J(x) \tag{3.10.8}$$

which can be substituted into Newton's iteration (3.10.4), giving

$$J(x)^T J(x)\,\Delta x = -\,J(x)^T R(x) \tag{3.10.9}$$

Equation (3.10.9) is known as the Gauss-Newton algorithm.

In practice it has been found that (3.10.8) is sometimes a poor approximation, leading to *divergence* of the iterates. Marquardt suggests that a "damping term" (λI) should be added to the Gauss-Newton algorithm [31]:

$$(J^T J + \lambda I) \Delta x = - J^T R \tag{3.10.10}$$

where I is the identity matrix and $\lambda \geq 0$ is a variable parameter of the algorithm.

On any particular iteration, J and R are fixed, so that Δx can be considered a function of λ. When $\lambda = 0$, Marquardt's algorithm reverts to the Gauss-Newton formula (3.10.9). As $\lambda \rightarrow \infty$, then $\Delta x \rightarrow - J^T R / \lambda$, which is an incremental step along the direction of steepest descent of f. Note that divergence is impossible with suitably large values of λ.

At iteration j, Marquardt's algorithm solves (3.10.10) for Δx, using increasingly larger values of λ until a displacement Δx is obtained for which $f(x_j + \Delta x) < f(x_j)$. Typically λ is increased in powers of ten until a function decrease is obtained:

$$\lambda^* = 10 \lambda \tag{3.10.11}$$

The algorithm is halted when the displacement Δx is negligibly small

$$\frac{\Delta x}{\tau + |x_j|} \leq \varepsilon_M \tag{3.10.12}$$

or when the value of λ exceeds a preset threshold

$$\lambda \geq \Lambda \tag{3.10.13}$$

The implementation of this algorithm described in chapter 4 uses

$$\begin{aligned} \tau &= 10^{-3} \\ \varepsilon_M &= 10^{-3} \\ \Lambda &= 10^{11} \end{aligned} \qquad (3.10.14)$$

Chapter 4: Automatic Parameter Extraction

Parameters for empirical MOS models are usually generated by a tedious set of hand calculations, followed by an iterative procedure of "tweaking" the numbers until an acceptable curve fit is obtained. Although it has been used successfully for many years, this method suffers from at least three major drawbacks: (1) it is time consuming, (2) it is prone to errors of omission or oversight, and (3) its termination criterion is qualitative rather than quantitative. This chapter discusses the computer implementation of numerical algorithms for parameter extraction, designed to eliminate these drawbacks. Prototype versions of parameter compilers (using these numerical algorithms) are presented. The performance of these compilers, measured in terms of modeling accuracy and compilation speed, is detailed.

Since transistors have several different regions of operation (Cutoff, Triode, Saturation) and geometry-dependent behavior (short and narrow channel effects), many measurements are needed to adequately characterize device operation in all regimes. These regions are arbitrarily constructed for ease of explanation and understanding; transistor behavior is actually a continuum, with certain effects more pronounced in some areas than in others. This leads to difficulties in parameter extraction, because (erroneous) assumptions are often made that device behavior is localized within these regions. For maximum accuracy, the entire operating range must be considered when extracting parameters, because they interact in the final model. Unfortunately, manual parameter extraction methods assume that (some) parameters are independent, and that their effects are localized; this necessitates "tweaking" the final numbers to obtain acceptable accuracy.

4.1: Manual Methods

To illustrate the methods used in manual parameter extraction, a manual extraction will be performed for the Mosaid model of section 2.7. The parameters of long, wide channel transistors will be extracted first [44]; the model fit for these large devices should be very good, because they will be closely approximated by the first-order theory. Then small-geometry effects will be accounted for, taking care to isolate the effect of each parameter, so its value can be calculated independently from the others.

The first parameter to be measured will be the threshold voltage V_T of a long, wide channel device. Equation (2.7.5b) shows that if V_{DS} is very small, drain current is proportional to $V_{GS} - V_T - V_{DS}/2$. Measurements of drain current are taken at various values of gate voltage V_{GS}; current begins to flow at $V_{GS} = V_T + V_{DS}/2$. This is shown in Figure 4.1. Drain current is not precisely linear with gate voltage, because the surface mobility (μ_{eff}) begins to fall as V_{GS} rises. V_T is found by extrapolating to $I_{DS} = 0$ from the maximum-slope portion of the curve, to insure that degraded mobility is not inadvertently included in the threshold voltage.

Figure 2.6 shows that threshold voltage is a function of source-to-bulk voltage V_{SB}; by repeating the measurement of Figure 4.1 at several values of V_{SB}, the body-effect parameters φ and γ can be computed. The parameter V_{T0} is simply the measured value of V_T taken at $V_{SB} = 0$. Rearranging equation 2.4.1,

$$V_T - V_{T0} = \gamma (\varphi + V_{SB})^{1/2} - \gamma (\varphi)^{1/2} \qquad (4.1.1)$$

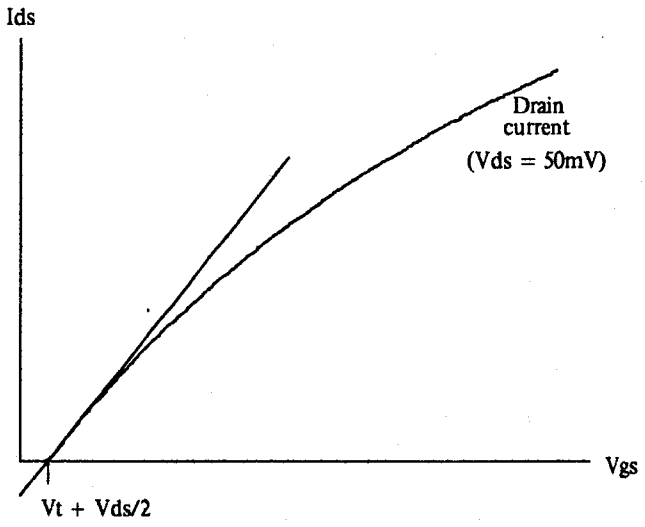

Figure 4.1 Measurement of V_T

Initially taking $\varphi = 0.6$ (a good approximation for silicon devices)[1], a change of variables can be made:

$$\begin{aligned} y &= V_T - V_{T0} \\ x &= (\varphi + V_{SB})^{1/2} \\ m &= \gamma \\ b &= -\gamma\,(\varphi)^{1/2} \end{aligned} \qquad (4.1.2)$$

Equation (4.1.1) is transformed into the familiar linear equation

$$y = mx + b \qquad (4.1.3)$$

[1] φ represents the Fermi potential of the silicon surface, $2\varphi_F$. It is proportional to the logarithm of the bulk doping density, and its value is usually close to 0.6 in typical silicon MOS transistors.

Using the trial value of φ, the measured data can be plugged into equations (4.1.2) and (4.1.3), extracting values for m and b with a simple linear regression. This computation is easy to perform, and is implemented on several hand-held scientific calculators [24].

The new value of φ is inserted into equation (4.1.2) in place of the initial guess, and another linear regression is performed. Regressions are iterated in this fashion until the values of φ and γ stabilize (say, to within 0.1%).

After extracting the model parameters for threshold, the gain parameters KP and θ are computed. Drain current measurements are taken on long, wide devices in the saturation region (where equation (2.7.5c) applies). Solving for KP,

$$KP = [1 + \theta(V_{GS} - V_T)] \frac{L}{W} \frac{I_{DS}}{(V_{GS} - V_T)^2} \tag{4.1.4}$$

which suggests the substitution

$$\begin{aligned} y &= \frac{L}{W} \frac{I_{DS}}{(V_{GS} - V_T)^2} \\ x &= \frac{L}{W} \frac{-(V_{GS} - V_T) I_{DS}}{(V_{GS} - V_T)^2} \\ m &= \theta \\ b &= KP \end{aligned} \tag{4.1.5}$$

The measurements are plugged into (4.1.5), and a linear regression is performed, giving θ and KP.

The channel-shortening parameter L_D can be found by measuring transconductance $(\partial I_{DS}/\partial V_{GS})$ as a function of mask channel length L. Two transistors with identical channel widths and different channel lengths are operated at

low drain voltage, and measurement of I_{DS} are taken at several values of V_{GS}. Transconductance is the maximum value of the slope of this curve; see Figure 4.1. Under these conditions,

$$\text{Slope} = \frac{\text{Const}}{L + \Delta L} \tag{4.1.6}$$

where $\Delta L = (-2L_D)$.

Measured values Slope_1 and Slope_2 are taken on devices with mask channel lengths L_1 and L_2. The two resulting equations (4.1.6) are divided, giving the ratio

$$\frac{\text{Slope}_1}{\text{Slope}_2} = \frac{L_2 + \Delta L}{L_1 + \Delta L} \tag{4.1.7}$$

Solving for ΔL,

$$\Delta L = \frac{L_2\,\text{Slope}_2 - L_1\,\text{Slope}_1}{\text{Slope}_1 - \text{Slope}_2} \tag{4.1.8}$$

At this point, model parameters V_{T0}, γ, φ, KP, θ, and L_D are known for long, wide transistors. The small-size parameters are extracted next, beginning with K_D. Rewriting equation (2.7.2),

$$K_D = \frac{L_{eff}}{V_{DS}} \{ V_{T0} - V_T + \gamma [(\varphi + V_{SB})^{1/2} - (\varphi)^{1/2}] \} \tag{4.1.9}$$

At a constant V_{DS}, threshold measurements are made for several values of channel length L_{eff}. Each data point is plugged into equation (4.1.9), giving a value

of K_D. These are averaged to produce the final K_D.

To extract a value of R_S, drain current measurements are taken in the triode region. The current equation (2.7.5c) is solved for R_S, giving

$$R_S = \frac{1}{2 K_{eff} V_{DS}} \left[\frac{K_{eff} W V_{DS}(2V_e - V_{DS})}{I_{DS}} - L_{eff} \right] \qquad (4.1.10)$$

Several I_{DS} measurements are taken on short channel transistors in the triode region, and values of R_S are then computed from (4.1.10). The final value of R_S is taken as the average of the results from (4.1.10). L_{eff}, K_{eff}, and V_e values used in these calculations should be computed from equations (2.7.1), (2.7.4), and (2.7.2) respectively.

C_{M0} can similarly be extracted from I_{DS} measurements made on short-channel devices in the saturation region. Equation (2.7.5c) is manipulated to yield

$$C_{M0} = \frac{1}{V_{Dsat} - V_{DS}} \left[\frac{K_{eff} W V_{Dsat}^2}{I_{DS}} - L_{eff} \right] \qquad (4.1.11)$$

V_{Dsat} is computed from equation 2.7.6 and known values of the other parameters. The final value of C_{M0} is taken to be the average of the individual values computed by equation (4.1.11).

Several simplifying assumptions were made in the parameter extraction recipe given above, resulting in increased modeling errors. For example, equation (4.1.6) ignores the R_S term in the denominator of equation (2.7.5b) by assuming that drain current is inversely proportional to $(L + \Delta L)$. More significantly, the values of V_{T0}, γ, φ, and K_D are derived from indirect measurements: first a quantity called "V_T" is

computed, and then parameter values are fitted from threshold data.

Actual MOS transistors do not abruptly turn off at $V_{GS} = V_T$, as shown in Figure 4.2. Below threshold, drain current is an exponential function of gate voltage [42, 43]. The Berkeley model includes an optional term to account for this "prethreshold conduction"; it was omitted from the discussion in chapter 2 because the option was turned off in the compilation experiments. (The Mosaid model does not account for this current). The gradual transition from "off" to "on" makes the task of selecting a unique V_T difficult, so parameter values extracted from such measurements will have large uncertainties.

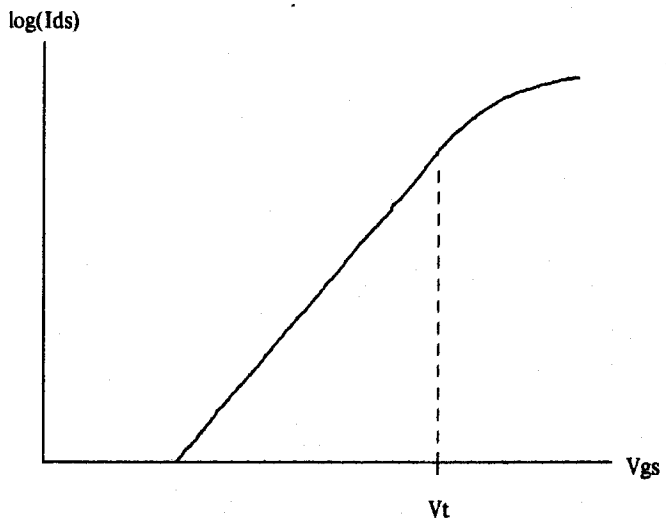

Figure 4.2 Prethreshold conduction

4.2: Numerical Techniques

The manual parameter extraction technique presented in the last section can be thought of as a sequence of minimizations: in each case, values for parameters are chosen so as to minimize modeling error. This is accomplished by using linear

regressions or by simple averaging. Unfortunately, the errors introduced in each of the individual extractions can accumulate, leading to a higher total modeling error (over the entire range of device operation).

The single-pass nature of the manual extraction technique does not permit residual model errors to be spread among several parameters, which could lead to an overall error reduction. Since each parameter is extracted in a small region of the total device operating range, with this scheme it is impossible to distribute errors over many such regions. In this section, numerical minimization algorithms (presented in chapter 3) will be used to extract all parameters simultaneously.

The objective function to be minimized will be a measure of the modeling error, for example the normalized error at each data point

$$e_j = \frac{I_{DSm}(j) - I_{DSp}(P)(j)}{I_{DSm}(j)} \qquad (4.2.1)$$

At the jth data point, the modeling error e_j is the normalized difference between the measured current $I_{DSm}(j)$ and the model's predicted current $I_{DSp}(P)(j)$. (The notation $I_{DSp}(P)(j)$ stands for the predicted value of I_{DS} at the jth data point, using the vector P of model parameters). Measurement j is taken on a transistor of size (W_j/L_j), at $(V_{GS}(j),\ V_{DS}(j),\ V_{SB}(j))$.

If the model current is a continuous, differentiable function of the parameters, then so is the modeling error e_j at each point. Quasi-Newton minimization algorithms, which require require the objective function to be differentiable, may therefore be used with equation (4.2.1).

A simple measure of the total modeling error with parameter-vector P is the sum of the squares of the n individual errors

$$f(P) = e_1^2 + e_2^2 + \cdots + e_n^2 \tag{4.2.2}$$

which is a differentiable function of P.

The distribution of measurements about the total operating range of the transistor will affect the accuracy of the final model. For example, if the vast majority of measurements are made in the linear region, model accuracy in the saturation region will be impaired. Regions of operation are, therefore, implicitly weighted in the error formulation, according to their frequency of appearance in the measurement data. If the measurements are clustered in a small region, the error will be lowest in that vicinity. Large errors at a few outlying points will be tolerated, because their contribution to the final total is small.

Explicit weighting is also possible [46], giving a total error function

$$f(P) = w_1 e_1^2 + w_2 e_2^2 + \cdots + w_n e_n^2 \tag{4.2.3}$$

Weights might be explicitly applied if it is desired to emphasize modeling accuracy in certain crucial regions of operation. Although implicit weighting could be used (simply by including many data points in those regions), explicit weights are more suitable, because they require fewer total data points (and fewer evaluations of f).

If the measurements are made on physical instruments, some accuracy will be lost at very low currents. It is desirable to prevent the extraction algorithm from generating faulty parameter values due to noise in these measurements. A modified error function can be used to set a "current floor"; below this floor value I_{min}, measurements are given less weight in the overall error function [46].

$$e_j = \frac{I_{DSm}(j) - I_{DSp}(P)(j)}{\max\left[I_{DSm}(j),\ I_{min} \right]} \tag{4.2.4}$$

A floor value is also useful if the model to be extracted does not account for

prethreshold conduction.

Several of the algorithms presented in chapter 3 are sensitive to the scale of the variables; if the elements of the parameter-vector differ by orders of magnitude, severe loss of accuracy will occur [31]. For this reason the parameter vector is approximately normalized; its elements are divided by typical values, insuring that all parameters are near unity. The parameters are of course de-normalized for the evaluation of the actual modeling routine.

The organization of a model parameter extraction program is diagrammed in Figure 4.3. First, n transistor measurements are input; they include the measurement conditions (W, L, V_{GS}, V_{DS}, V_{SB}) and the resulting drain current I_{DS}. A numerical minimization algorithm is then used to perturb the parameter vector P until the error function $f(P)$ of equation (4.2.2) is minimized. For each measurement, the parameters P and the measurement conditions are input to the empirical MOS modeling routine, giving a predicted drain current I_{DSp}. Equation (4.2.4) is used to find the error for that measurement, which is summed with the others to form $f(P)$.

This program organization is rather independent of the specific model equations; new models may be extracted merely by substituting a new empirical modeling routine.

4.3: Compilation

The program of Figure 4.3 can be fed input measurements which are in fact the results of a simulation; a model compilation is performed exactly this way. Parameters for an empirical model are then generated automatically, using input parameters for a theoretical model to drive the simulation.

To examine the model accuracy obtainable with this approach, five parameter extraction programs were constructed and tested. The only differences among the programs were the numerical minimization algorithms used. Extractions were performed

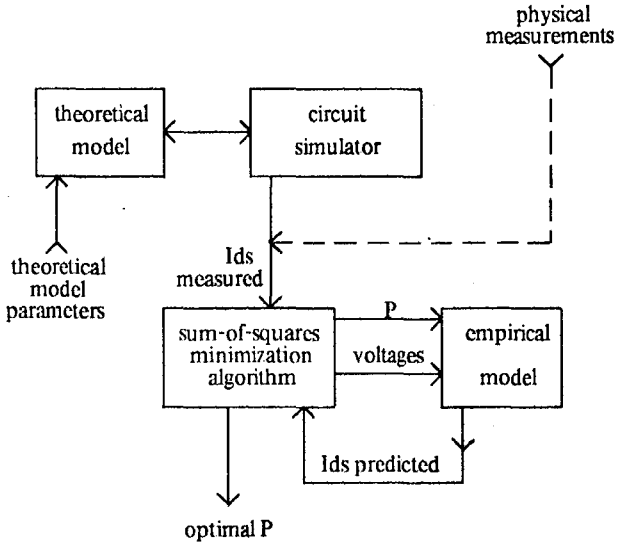

Figure 4.3 Program organization

with Hooke & Jeeves, Simplex, Conjugate Gradient, Fletcher, and Marquardt procedures.

Parameter "compilations", as described in chapter 1, were performed: a set of parameters for the Berkeley model was supplied, and the extraction programs computed a corresponding set of Mosaid model parameters. The SPICE 2G.5 program was used to simulate measurements of test transistors (using the Berkeley model and the given parameters), and these measurements were fed to the extraction programs. Extraction algorithms minimized the error function (4.2.3), attempting to generate Mosaid model parameters which would predict exactly the same current-voltage behavior as the Berkeley parameters did.

The simulated "measurements" must be carefully chosen to represent the whole range of anticipated device operation. If no data is supplied for a particular region of operation, the resulting empirical model is forced to extrapolate to predict behavior in that region, resulting in higher modeling error. Short, long, wide, and narrow channel

devices should be included in the measurements, operating in cutoff, triode, and saturation. Both low and high-current measurements should be taken, and back-bias effects ($V_{SB} > 0$) should be included. Table 4.4 shows the simulated measurements used in the compilation experiments. Six values of V_{DS}, five values of V_{GS}, four values of W/L, and two values of V_{SB} were simulated, for a total of 240 data points.

Variable	Values					
(W/L)	(100/100)	(100/4)	(4/100)	(4/4)		
V_{SB}	0	-3				
V_{GS}	0.5	1	1.5	3	5	
V_{DS}	0.05	1	2	3	5	6

Table 4.4 Data points for parameter compilation

The parameters used for the Berkeley theoretical model are shown in Table 4.5. They represent a typical contemporary n-channel MOS process, approximately of the same dimensions as "HMOS-1" [36].

A set of normalization constants were used to keep the elements of the Mosaid parameter vector approximately at the same order of magnitude (near unity); these are shown in Table 4.6. The starting values of the normalized parameters were all set to 1.0.

Explicit weighting was used to emphasize measurements at small currents; this boosts model accuracy near "V_T", and improves circuit simulations of inverter trip-

Parameter	Value	Units
$2\varphi_F$	0.633	Volts
T_{ox}	7E–8	Meters
NSUB	3E21	Meters^{-3}
N_{ss}	1E14	Meters^{-2}
XJ	5E-7	Meters
L_D	5E–7	Meters
μ_0	6E–2	Meters^2/Volt–Second
UCRIT	4E6	Volts/Meter
UEXP	0.37	---
VMAX	6E4	Meters/Second
NEFF	6.0	---

Table 4.5 Berkeley parameter values

points. The weighting constants are

$$w_j = 4.0 \quad \text{if } I_{meas}(j) \leq (10^{-6})\ \frac{W_j}{L_j}$$
$$w_j = 1.0 \quad \text{otherwise} \qquad (4.3.1)$$

The choice of a "current floor" I_{min} for equation (4.2.4) represents a tradeoff; low values of I_{min} give high accuracy at low currents with diminished accuracy at high currents. Large values of I_{min} have the opposite effect. Experiments were performed

Parameter	Multiplier
V_{T0}	1.0
KP	1E-5
γ	0.1
φ	0.1
θ	0.1
L_D	1E-6
C_{M0}	1E-7
K_D	1E-7
R_S	1E-3

Table 4.6 Mosaid parameter normalization coefficients

to find a value of I_{min} that gives a reasonable balance between errors at low and high currents.

Table 4.7 shows the total modeling error $f(P)$ and RMS percentage error $[100\ (\ f(P)/240\ )^{1/2}\ ]$ at four different values of I_{min}. The error values cited in this table are the errors produced by Fletcher's minimization algorithm. As I_{min} increases, total error decreases, since the error contributions of low-current measurements are diminished.

Figures 4.8 - 4.11 show the model fit, from parameters extracted by Fletcher's algorithm, as a function of I_{min}. The I-V curves of a (100/4) transistor are plotted, simulated with the input theoretical model (solid curve) and with the resulting empirical

I_{min} (μA)	total error	RMS error
2	4.3383	13.44%
20	0.7712	5.67%
60	0.4095	4.13%
200	0.1280	2.31%

Table 4.7 Effect of I_{min} on modeling error

model (dotted curve). One set of curves is given for low currents (V_{GS} = 0, 0.5, 1.0, ..., 2.5) and one set is plotted for high currents (V_{GS} = 0, 1, 2, ..., 5). In both cases V_{DS} ranges from 0 to 6 volts.

Comparing Figures 4.8 and 4.9, it appears that the fit is not dramatically altered by raising I_{min} from 2 to 20 microamps. However, as I_{min} is increased to 60 and 200 microamps (Figures 4.10 and 4.11), the curves begin to move and accuracy is improved at high currents. Eventually low-current accuracy is sacrificed, and increasing I_{min} worsens the problem. Based on these observations, a value of I_{min} = 60 μA was selected to be incorporated into the final parameter compilation procedure.

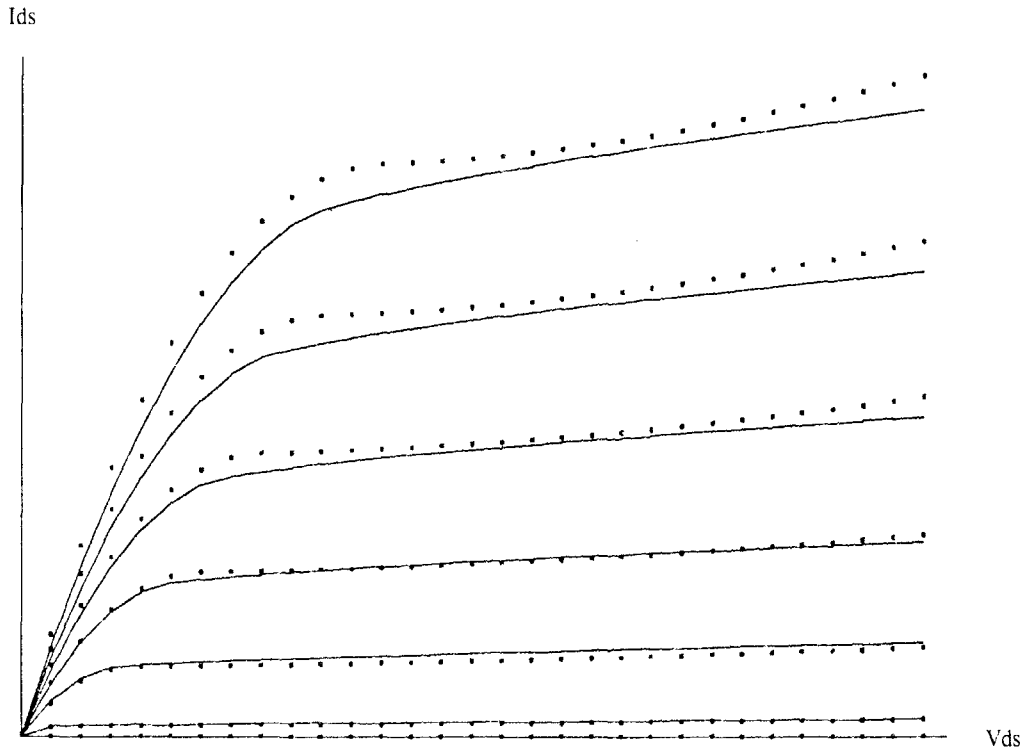

Figure 4.8a Model Fit at Low Current, I_{min} = 2 microamps.

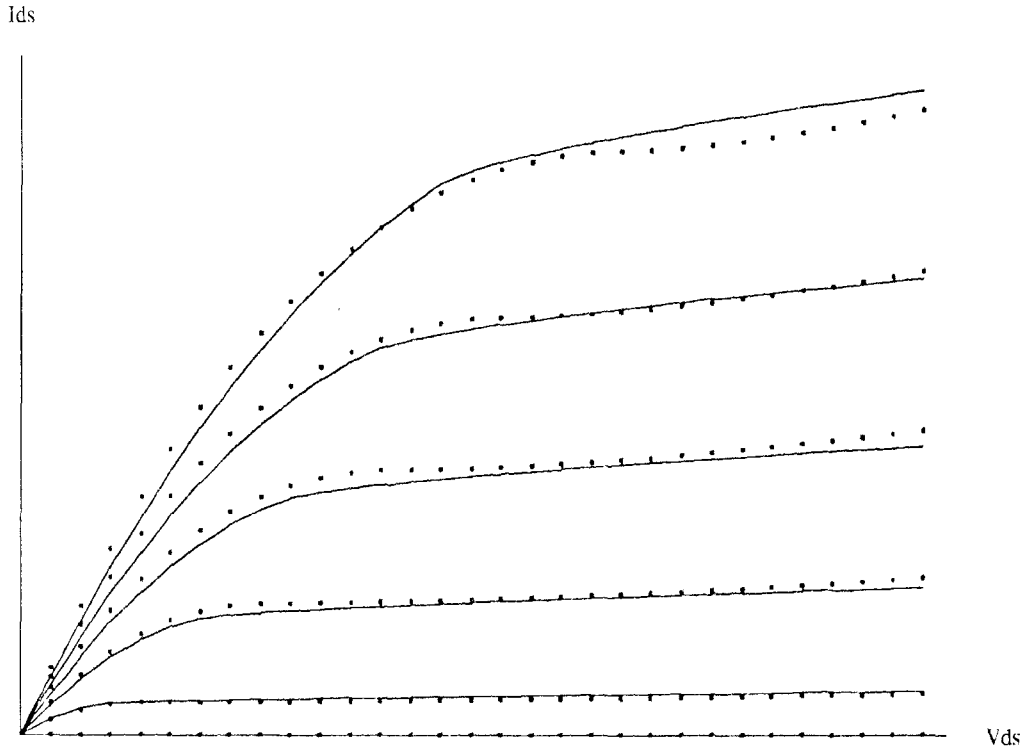

Figure 4.8b Model Fit at High Current, I_{min} = 2 microamps.

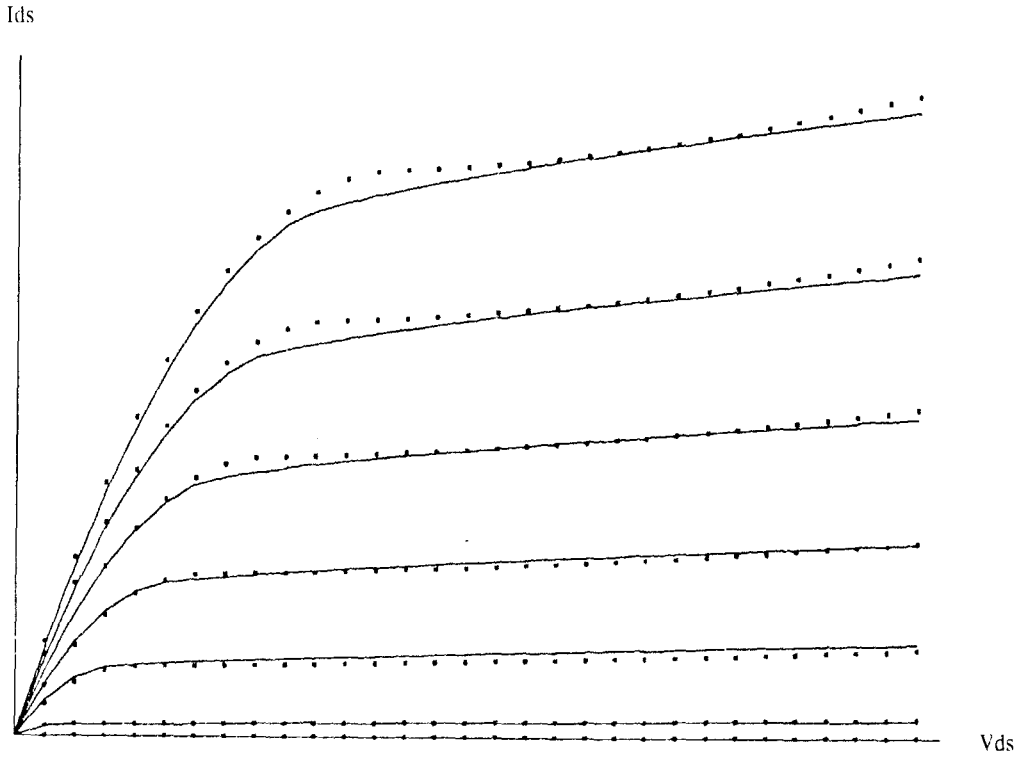

Figure 4.9a Model Fit at Low Current, I_{min} = 20 microamps.

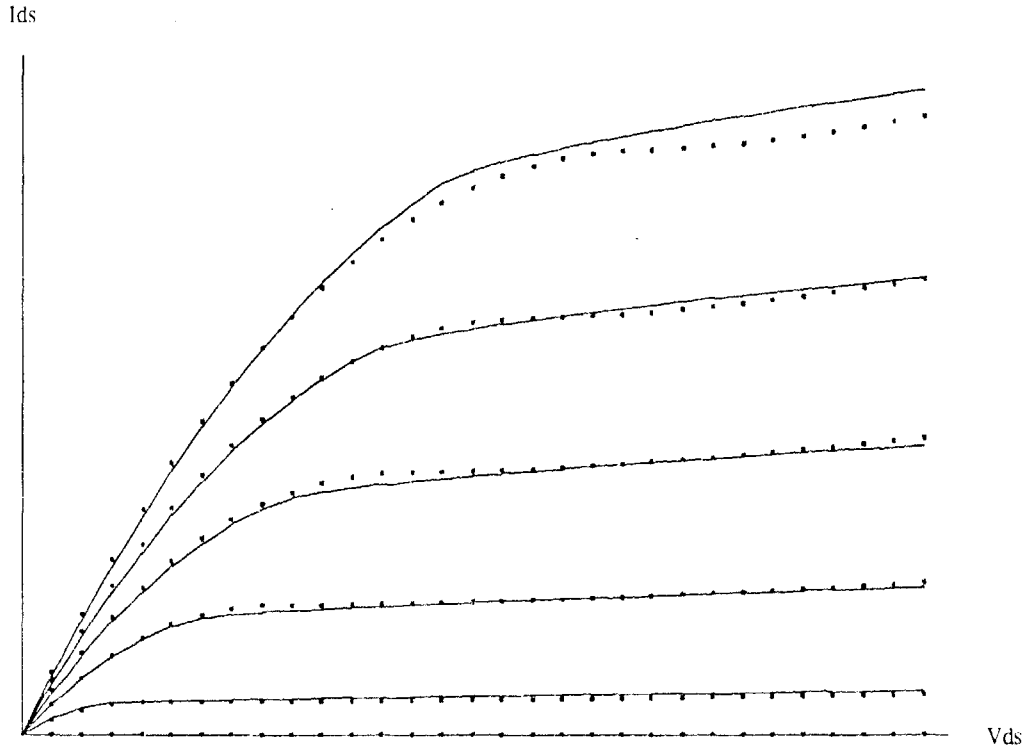

Figure 4.9b Model Fit at High Current, I_{min} = 20 microamps.

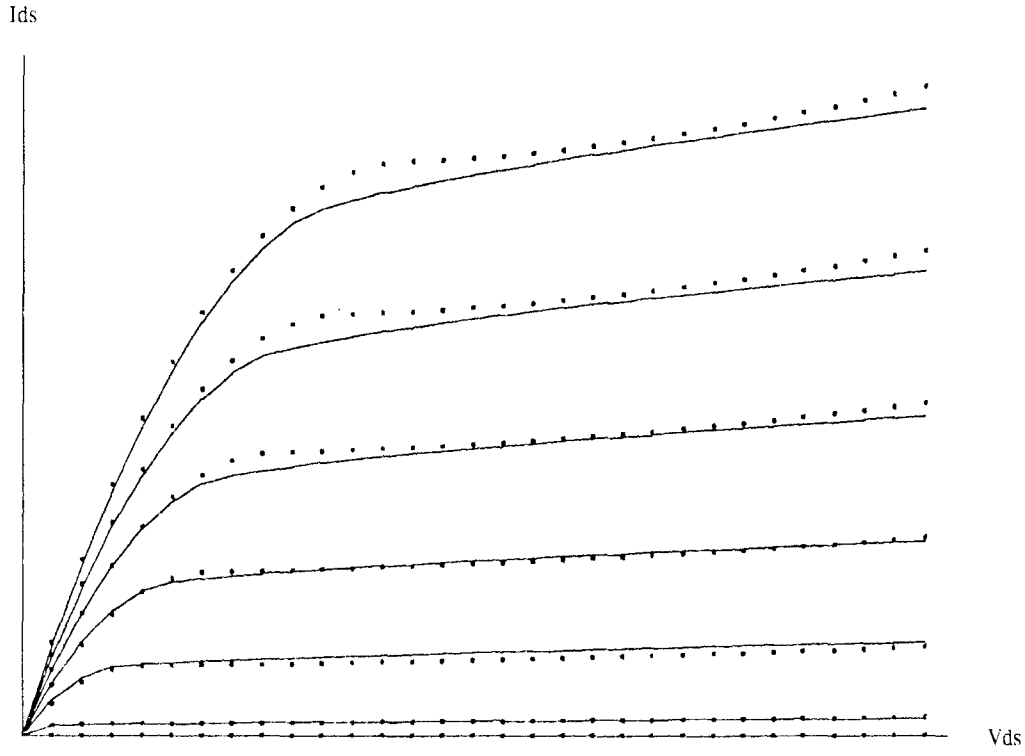

Figure 4.10a Model Fit at Low Current, I_{min} = 60 microamps.

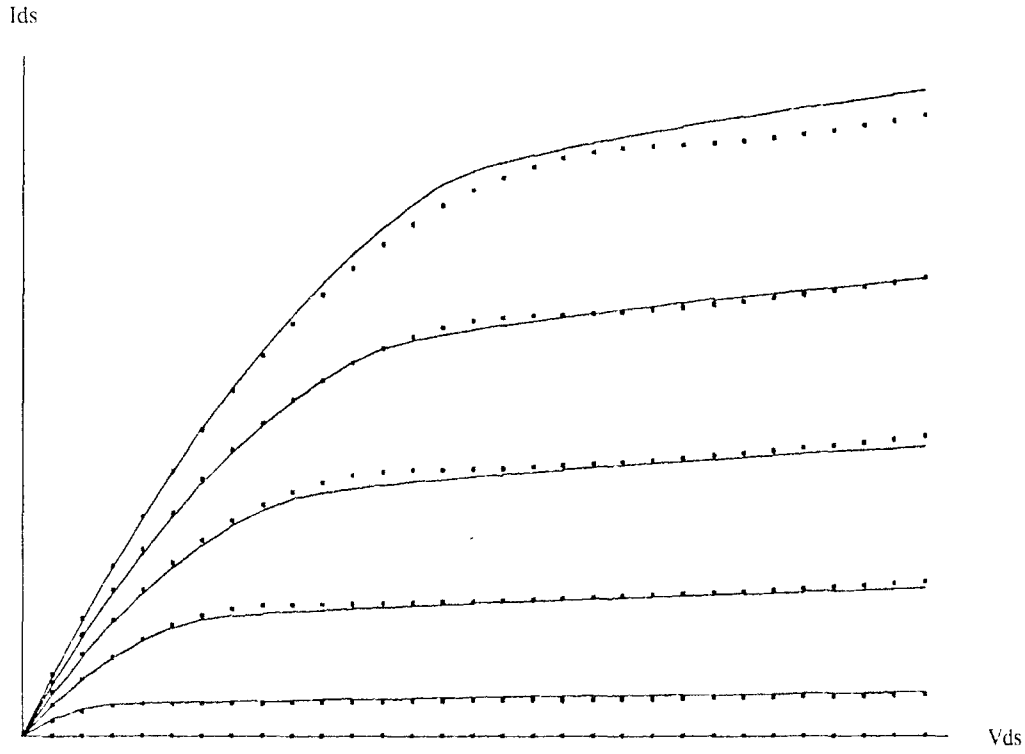

Figure 4.10b Model Fit at High Current, I_{min} = 60 microamps.

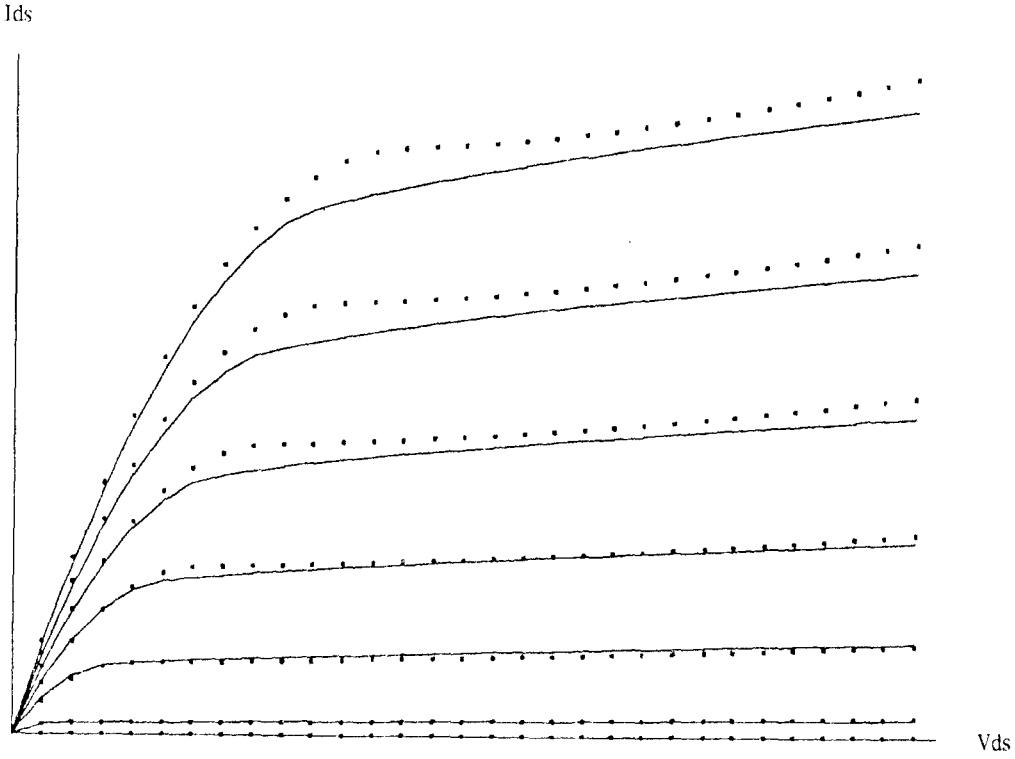

Figure 4.11a Model Fit at Low Current, I_{min} = 200 microamps.

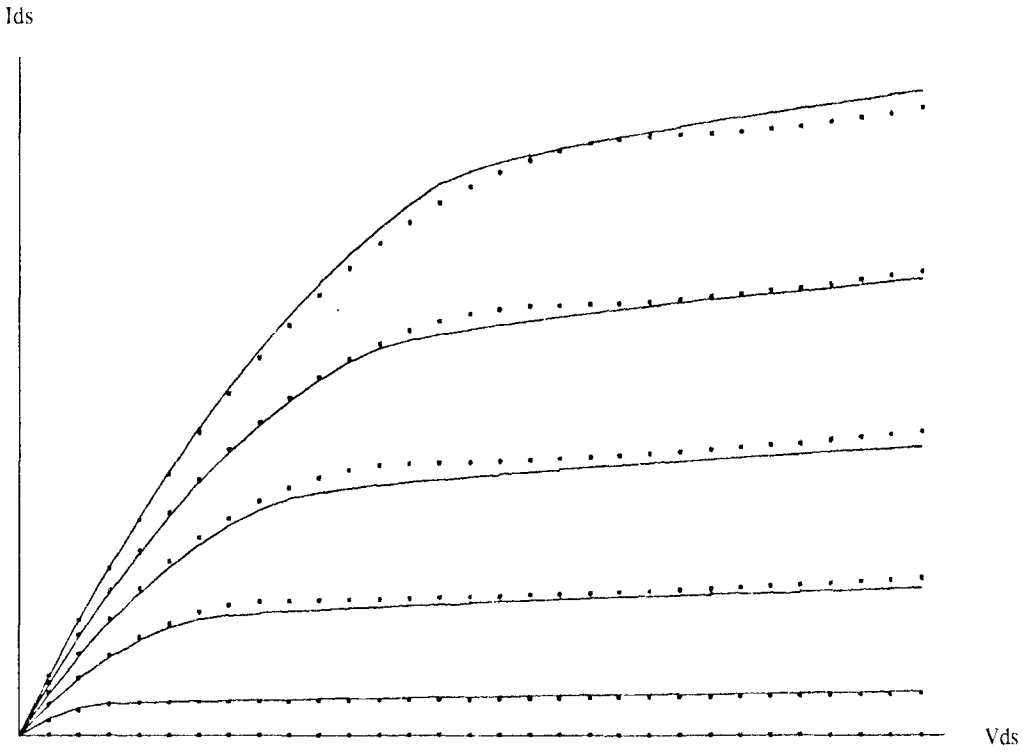

Figure 4.11b Model Fit at High Current, I_{min} = 200 microamps.

These parameter extractions were performed using a sum-of-squares measure of modeling error. Sum-of-squares tends to optimize the average case, because the error of each data point is summed into the final measurement. A different style of error measure can be defined which optimizes the worst case; it is called "minimax":

$$g(P) = \max |e_j| \tag{4.3.2}$$

Note that the function $g(P)$ in the minimax formulation is not differentiable; quasi-Newton algorithms therefore cannot minimize it directly. Although techniques exist for transforming a minimax problem into a sequence of differentiable functions [5], it is simpler to exploit a minimization algorithm which does not require differentiability.

Far away from the minimax solution, it is likely that $g(P) = 1$, because some data point will give $I_{meas} > 0$ while the model predicts $I_{pred} = 0$ (see equation 4.2.4). Small perturbations of the parameter vector P will not change $g(P)$, and the minimization algorithm will terminate because the local gradient is zero. The minimax formulation therefore requires a starting point which is close to the final solution; in particular $g(P)$ must be less than 1. A useful heuristic which often satisfies this requirement is to start the minimax algorithm with the final parameter vector produced by a sum-of-squares procedure; this effectively adds the execution times of the two programs.

A version of the Hooke & Jeeves algorithm was coded which used (4.3.2) as the error function. The value of ρ was increased to 0.8, so the steplength did not shrink so quickly between iterations. This prevents the algorithm from overlooking a promising search direction. Modeling results using this minimax technique are shown in Figure 4.12. Comparing these with Figure 4.10, the overall curve-fits obtained by minimax optimization appear to be inferior to sum-of-squares results. Since there is no penalty (in terms of g) for increasing the error at a point which is not the worst-case, the minimax algorithm tolerates rather large errors at all points.

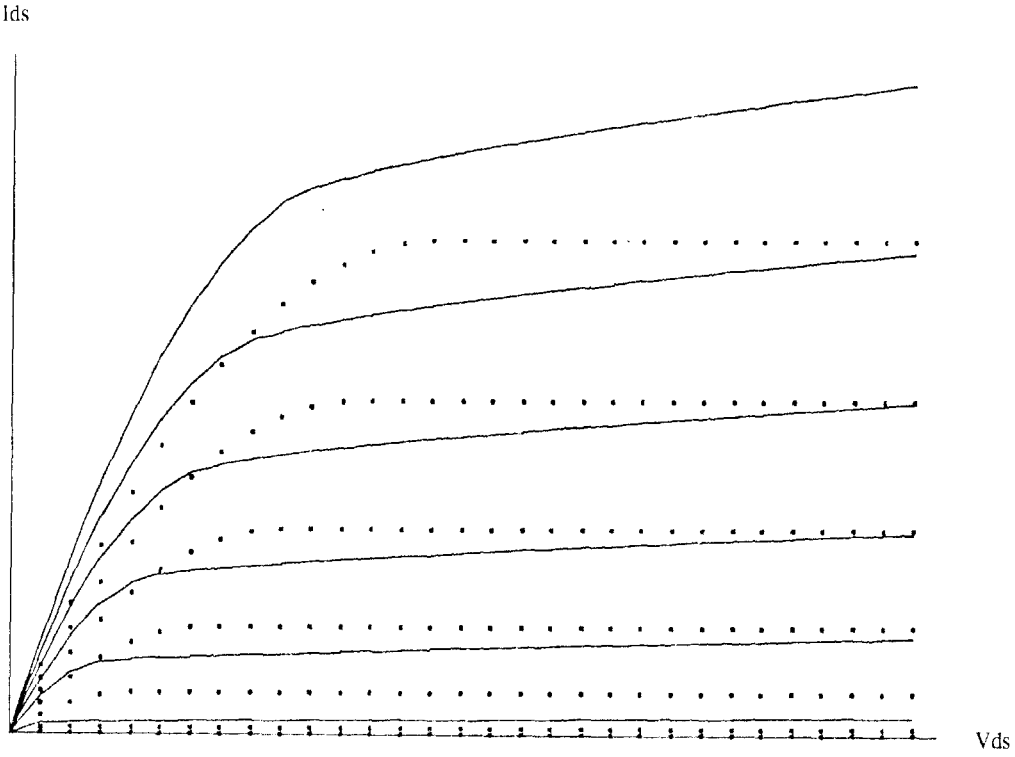

Figure 4.12a Model Fit at Low Current, I_{min} = 60 microamps.

(Minimax error function)

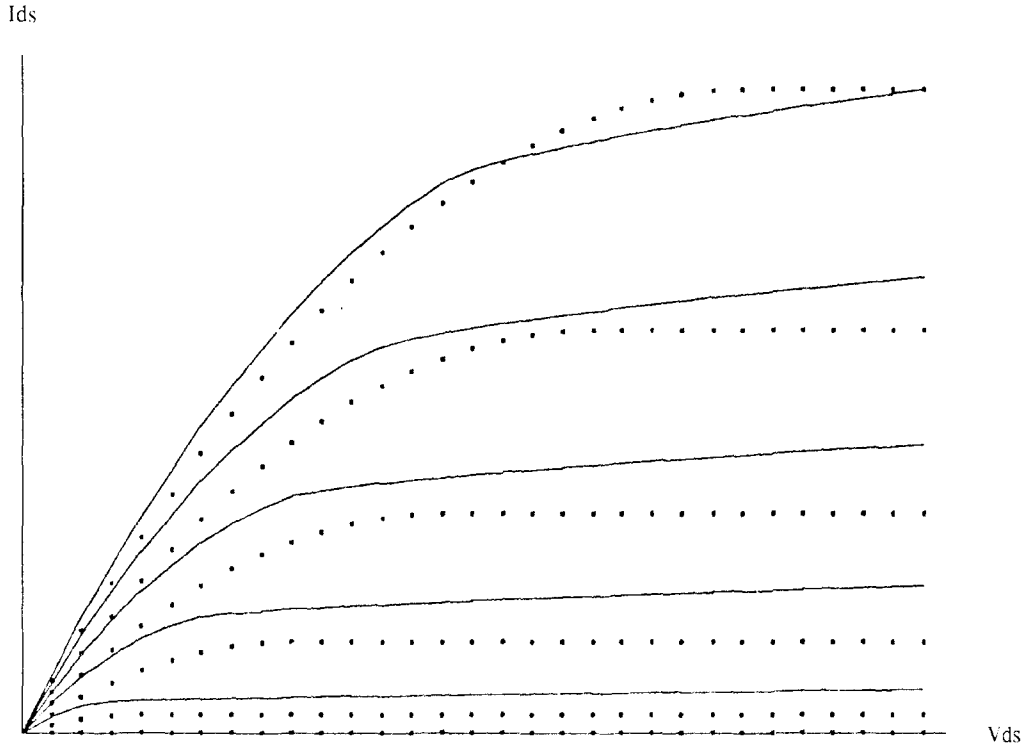

Figure 4.12b Model Fit at High Current, I_{min} = 60 microamps.

(Minimax error function)

4.4: Algorithm Performance

Five extraction programs using sum-of-squares minimization, and one program using minimax, were monitored for three values of floor current: I_{min} = 20, 60, and 200 μA. Table 4.13 shows the performance of each algorithm when compiling the example model from the previous section. All experiments were performed on a VAX[2] 11/780 computer running release 4.1 of the V7 UNIX[3] operating system. Programs were coded in the C language.

The third column of Table 4.13 indicates how many times the error function $f(P)$ (equation 4.2.3) was evaluated during the minimization, while the fourth column gives the total run time in user CPU-seconds. The last column shows the value of $f(P)$ returned when the algorithm terminated. Function evaluations and CPU times given for the Hooke & Jeeves minimax algorithm include the first sum-of-squares phase. Note also that $g(P)$ values from the minimax algorithm are not directly comparable to $f(P)$ values produced by the other algorithms.

Parameter extraction time is related to the complexity of the model; the longer it takes to evaluate $f(P)$, the longer it will take to extract the model parameters. If $f(P)$ is sufficiently expensive to compute, the computational overhead of the minimization algorithm becomes negligible, and execution speed is determined solely by the evaluations of $f(P)$. Therefore the number of computations of $f(P)$ should be considered along with the CPU time when deciding the relative merits of extraction algorithms.

The Marquardt algorithm is designed to handle only sum-of-squares problems, yet

[2]VAX is a registered trademark of Digital Equipment Corporation.
[3]UNIX is a registered trademark of Bell Telephone Laboratories.

Algorithm	I_{min}, μA	$f(P)$ evals	CPU time	final $f(P)$
Fletcher	20	3230	913.0	0.7713
Marquardt	20	3801	1549.3	0.7733
Hooke&Jeeves	20	2080	632.8	1.3864
Simplex	20	3230	2638.0	0.7713
Conj. Grad.	20	12289	3459.7	0.9567
Minimax	20	3455	1018.9	0.15986
Fletcher	60	3288	907.1	0.4095
Marquardt	60	2528	1044.8	0.4095
Hooke&Jeeves	60	1754	499.1	0.5439
Simplex	60	5774	1710.2	0.4095
Conj. Grad.	60	12205	3571.1	0.4256
Minimax	60	2976	800.6	0.13944
Fletcher	200	2880	813.3	0.1280
Marquardt	200	856	370.0	0.1280
Hooke&Jeeves	200	1455	436.3	0.1614
Simplex	200	5774	3304.6	0.1282
Conj. Grad.	200	12205	3505.6	0.1294
Minimax	200	2624	760.0	0.06888

Table 4.13 Parameter compilation speed of minimization algorithms

it is not superior to Fletcher's method except in the case $I_{min} = 200\ \mu A$. Recent data indicates that if the objective function is highly nonlinear, and the final error is nonzero, then special sum-of-squares algorithms are no better than general minimizers

[34].

Hooke & Jeeves is the generally the fastest algorithm, but it yields function values which are not minimal. If, however, the model is extremely expensive to evaluate, Hooke & Jeeves can be used to quickly locate a parameter-vector near the minimum. Then a slower, more accurate procedure (such as Marquardt's) could seek out the final optimum.

Fletcher's variable-metric algorithm consistently produces the lowest value of $f(P)$, while maintaining a relatively small number of function evaluations and a rapid execution speed. It is therefore chosen as the single minimization procedure to be incorporated into the final parameter compiler.

Table 4.14 gives the (unnormalized) values of the Mosaid parameters produced by Fletcher's algorithm for each value of l_{min}.

4.5: Simulation Speed

To asses the benefits derived from using an empirical model, simulations were performed on two identical circuits. One was simulated using the Berkeley model, and the other was simulated using the compiled Mosaid model of the previous section. The simulations were identical in every other respect, so the difference in measured execution speed was caused by the difference in the models.

This measurement was repeated for fourteen circuits, each having a different number of MOS transistors, to monitor the effect of circuit size on relative simulation speed. The circuits used are shown in Figure 4.15; they consist of an iterated parallel connection of 5-transistor modules. A parallel topology was chosen to guarantee that the simulator would evaluate every transistor at every timestep. Other topologies (such as a series connection) might contain nodes which stay at a constant value for many timesteps, allowing a simulator to bypass the evaluation of transistors connected to those nodes [3]. This would tend to invalidate size-versus-speed measurements, since

Parameter	l_{min} = 200 (SSQ)	l_{min} = 60 (SSQ)	l_{min} = 60 (Minimax)	l_{min} = 20 (SSQ)
V_{T0}	2.065E-1	1.956E-1	1.834E-1	1.838E-1
KP	1.249E-5	1.148E-5	1.205E-5	1.059E-5
γ	3.717E-1	3.900E-1	4.043E-1	4.045E-1
φ	5.485E-1	5.511E-1	7.318E-1	5.734E-1
θ	2.082E-1	1.769E-1	1.856E-1	1.516E-1
L_D	8.196E-7	8.539E-7	6.823E-7	9.076E-7
C_{M0}	8.602E-8	9.247E-8	6.654E-8	9.040E-8
K_D	3.487E-8	3.073E-8	4.386E-8	2.740E-8
R_S	1.053E-2	1.265E-2	6.734E-3	1.473E-2

Table 4.14 Compiled values of Mosaid model parameters

the number of simulator evaluations is no longer proportional to circuit size.

If all of the iterated modules in the parallel topology were identical, the waveforms at the internal nodes would be identical. A (very sophisticated) static pre-analysis could notice this fact, and bypass the majority of the computation by simulating only one module. The circuit of Figure 4.15 avoids this problem by making transistor M1 a different size in each module. Channel length was selected by generating a random number in the range ($8\mu \leq L \leq 12\mu$), and channel width was chosen from ($20\mu \leq W \leq 30\mu$).

The results of these experiments are shown in Tables 4.16 and 4.17; they are also

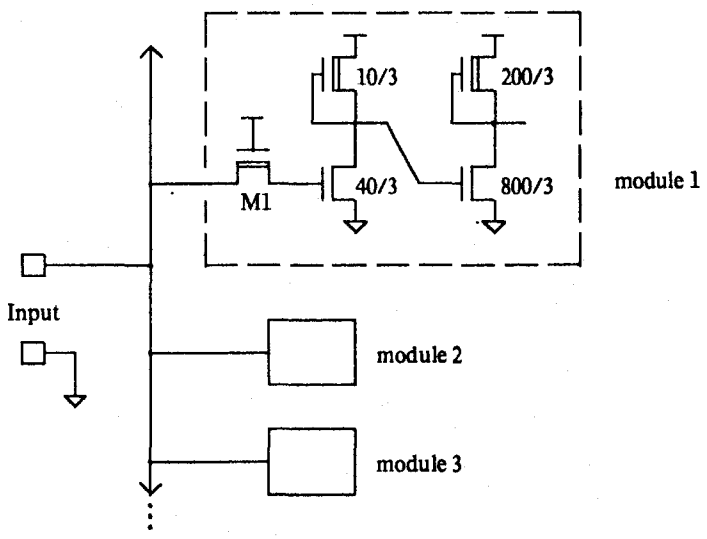

Figure 4.15 Variable-size circuit for speed experiments

plotted (in normalized form) as Figure 1.1. Measured SPICE execution time for a circuit of n transistors was fitted to the equation

$$\text{Time} = C + K\, n^{E}$$

Sum-of-squares fitting was used to find optimum values of C, K, and E; the worst-case error at any single point was 6%. The computed optimum values are

Model	C	K	E
Mosaid	2.141	3.298	1.036
Berkeley	2.455	5.125	1.042

The "average speedup" achieved by using the Mosaid model instead of the Berkeley model is $K_{\text{Mosaid}} / K_{\text{Berkeley}} = 0.6435$. That is, SPICE simulations using the Mosaid model will take 65% as long as the same simulations using the Berkeley model.

n	average SPICE time (seconds)	Number of trials	σ (seconds)
2	8.873	100	0.13
5	20.06	100	0.35
10	37.26	20	0.76
20	76.48	10	0.82
25	92.50	5	1.19
50	186.58	5	2.00
75	294.54	10	1.63
100	418.00	2	1.41
150	594.14	5	1.32
200	827.60	1	---
300	1200.1	1	---
450	1833.2	1	---
725	2943.4	1	---
1000	4341.5	1	---

Table 4.16 Simulation speed using Mosaid model

n	average SPICE time (seconds)	Number of trials	σ (seconds)
2	12.77	100	0.26
5	31.55	100	0.68
10	59.16	20	1.06
20	115.51	10	1.19
25	152.02	5	0.88
50	287.70	5	2.59
75	472.55	10	4.67
100	635.65	2	1.06
150	955.86	5	5.20
200	1307.4	1	---
300	1910.5	1	---
450	2929.2	1	---
725	4932.2	1	---
1000	7063.6	1	---

Table 4.17 Simulation speed using Berkeley model

Chapter 5: Conclusions and Directions for Further Research

The principal goal of this effort has been to demonstrate the practicality of a two-level modeling scheme. An automatic parameter extraction program was exhibited in chapter 4, proving the feasibility of the proposed translation mechanism. Parameters for the Berkeley theoretical model were successfully compiled into parameters for the Mosaid empirical model. Errors introduced by the second level of modeling were small ($\lesssim$5%), which suggests that simulation accuracy will not be appreciably degraded.

Several numerical minimization algorithms were tested, and Fletcher's switching-policy variable metric algorithm was found best for extracting Mosaid model parameters. The Marquardt algorithm did not offer substantial improvements in speed or accuracy, even though it is specifically designed to operate on parameter-extraction problems. A "current floor" of 60 microamperes was found to provide a good tradeoff between accuracy at low currents and accuracy at high currents.

The effect of model complexity on total simulation time was measured, and it was found that simple empirical models provide substantial improvements over theoretical models. Speedups approaching 35 percent were observed, indicating that the two-level modeling technique will allow significant increases in simulation productivity. Much additional investigation remains to be done; this chapter outlines several areas in which the two-level modeling idea can be extended and improved.

5.1: Highly Nonlinear Models

The Mosaid model used in the compilation experiments is characterized by a small set of equations, in which each model parameter is a coefficient for a simple polynomial. Predicted currents are gently varying functions of parameter values, so minimization procedures which operate on first derivatives will give fast convergence. If, however, a model with a strong nonlinear relationship between parameter values and

predicted currents is chosen, these algorithms will converge much more slowly [25]. This suggests a combined approach, using a function-comparison method to quickly find a point near the optimum (where the error in the Newton approximation is small). Then a quasi-Newton procedure could be used for the final convergence stage.

5.2: Nearly Redundant Parameters

Model equations may be defined such that some parameters are redundant or nearly redundant. For example, the equation $I = K_1 K_2 V$ is a poor model of a resistor because K_2 is redundant: there are an infinite number of (K_1, K_2) pairs which predict the same current-voltage behavior.

The Berkeley model suffers from this problem, because device gain is set by the ratio of two input parameters:

$$I_{DS} \propto \frac{\mu_0}{T_{ox}}$$

This apparent redundancy can send some minimization algorithms into slightly-damped oscillation, repeatedly varying μ_0 and T_{ox} until a stable solution is stumbled upon. The two parameters are not completely redundant, but the differences between them are only manifested in second order effects (such as the body-effect coefficient γ_S). Parameter extraction is therefore extremely slow, because the distinction between μ_0 and T_{ox} is only apparent near the optimum point.

Preliminary extraction experiments with the Berkeley model tend to indicate that simple function-comparison procedures (such as the Simplex algorithm) converge sooner than quasi-Newton routines, particularly if the starting point is far away from the optimum. Other extraction efforts on the Berkeley model [46] have ignored the

redundant-parameter problem, preferring to fix one of the two parameters as a constant, and using a quasi-Newton algorithm to extract the other.

5.3: Incremental Extraction

To adequately represent device behavior in all important regions of operation, the parameter extraction algorithm requires many input measurements. For each measurement, a model prediction must be calculated, so the time required to perform an extraction is proportional to the number of measurements supplied. Unfortunately, there seems to be a tradeoff between extraction time and goodness of fit.

This difficulty may lend itself to a divide-and-conquer approach: an extraction is performed on a small subset of the data, producing an optimal point in the parameter space for that subset. Then a full extraction is run on the entire set of measurements, using this solution as a starting point. The improved starting point will allow the full extraction to use fewer iterations to compute the optimum parameter values. If several iterations of the full extraction can be eliminated, the net savings in extraction time will be substantial.

5.4: Multiple Models

Circuit simulators have traditionally implemented a single model, with a single set of parameters, to represent all transistors that could possibly be built with a given fabrication technology. This restriction could be relaxed, allowing model parameters to be optimized for a particular subset of all possible transistors. An obvious example would be to extract four sets of parameters, separately modeling the four different size-classes of transistors:

Width	Length	Class
$\geq 9\mu$	$\geq 9\mu$	Wide, Long
$\geq 9\mu$	$\leq 9\mu$	Wide, Short
$\leq 9\mu$	$\geq 9\mu$	Narrow, Long
$\leq 9\mu$	$\leq 9\mu$	Narrow, Short

Model parameters could be separately optimized within each class, which permits better curve-fits and lower total model error. This approach would especially improve modeling of narrow, short devices, because interactions between narrow and short channel effects would be explicitly accounted for. A simple preprocessing stage in the circuit simulator would assign each transistor in the simulated circuit to one of these classes, so the proper set of parameters is always used.

References

[1] L. A. Akers and J. J. Sanchez, "Threshold Voltage Models of Short, Narrow, and Small Geometry MOSFET's: A Review", *Solid-State Electronics*, Vol. 25, pp. 621-641, 1982.

[2] L. A. Akers, "An Analytical Expression for the Threshold Voltage of a Small Geometry MOSFET", *Solid-State Electronics*, Vol. 24, pp. 621-627, 1981.

[3] C. M. Baker and C. Terman, "Tools for Verifying Integrated Circuit Designs", *Lambda*, Vol. 1, No. 3, pp. 22-30, 1980.

[4] M. B. Bandali and T. C. Lo, "On Modeling of the Self-Aligned Field Implanted MOS Devices with Narrow Widths", in *1975 Tech. Dig. Int. Electron Devices Meeting*, pp. 573-576.

[5] J. W. Bandler and C. Charalambous, "Nonlinear Programming Using Minimax Techniques", *J. Optimization Theory and Appl.*, Vol. 13, pp. 607-619, 1974.

[6] C. S. Beightler, D. T. Phillips, and D. J. Wilde, *Foundations of Optimization*, Prentice-Hall, Englewood Cliffs, NJ. 1979.

[7] E. M. Butler, "On-Line Transistor Modeling in a Manufacturing Environment", *IEEE Trans. Circuit Theory*, vol. CT-20, pp. 683-687, 1973.

[8] P. K. Chatterjee, and J. E. Leiss, "An Analytic Charge-Sharing Predictor Model for Submicron MOSFETs", in *1980 Tech. Dig. Int. Electron Devices Meeting*, pp. 28-33.

[9] J. Compeers, H. J. De Man, and W. M. C. Sansen, "A Process and Layout Oriented Short-Channel MOST Model for Circuit-Analysis Programs", *IEEE Trans. Electron Devices*, vol. ED-24, pp. 739-745, 1977.

[10] L. M. Dang, "A Simple Current Model for Short-Channel IGFET and Its Application to Circuit Simulation", *IEEE J. Solid-State Circuits*, vol. SC-14, pp. 358-367, 1979.

[11] R. H. Dennard, F. H. Gaensslen, H. N. Yu, V. L. Rideout, E. Bassous, and A. R. LeBlanc, "Design of Ion-Implanted MOSFET's with Very Small Physical Dimensions", *IEEE J. Solid-State Circuits*, vol. SC-9, pp. 256-267, 1974.

[12] L. C. W. Dixon, ed., *Optimization in Action*, Academic Press, London, 1976.

[13] Y. A. El-Mansy and A. R. Boothroyd, "A New Approach to the Theory and Modeling of Insulated-Gate Field-Effect Transistors", *IEEE Trans. Electron Devices*, vol. ED-24, pp. 241-253, 1977.

[14] J. K. Fidler and C. Nightingale, *Computer Aided Circuit Design*, John Wiley and Sons, New York, 1978.

[15] R. Fletcher, "A New Approach to Variable Metric Algorithms", *Computer Journal*, Vol. 13, pp. 317-322, 1970.

[16] R. Fletcher, *Practical Methods of Optimization, Volume 1: Unconstrained Optimization*, John Wiley and Sons, Chichester, U.K., 1980.

[17] R. Fletcher and M. J. D. Powell, "A Rapidly Convergent Descent Method for Minimization", *Computer Journal*, Vol. 6, pp. 163-168, 1963.

[18] R. Fletcher, ed., *Optimization*, Academic Press, London, 1969.

[19] R. C. Foss, R. Harland, and J. Roberts, "An MOS Transistor Model for a Micro-Minicomputer based Circuit Analysis System", in *Fifth European Solid-State Circuits Conf., Dig. Tech. Papers*, pp. 56-57, 1979.

[20] D. Frohman-Bentchkowsky and A. S. Grove, "Conductance of MOS Transistors in Saturation", *IEEE Trans. Electron Devices*, vol. ED-16, pp. 108-113, 1969.

[21] M. Fukuma and Y. Okuto, "Analysis of Short-Channel MOSFET's with Field-Dependent Carrier-Drift Mobility", *IEEE Trans. Electron Devices*, vol. ED-27, pp. 2109-2114, 1980.

[22] P. E. Gill and W. Murray, "Safeguarded Steplength Algorithms for Optimization using Descent Methods", British National Physical Laboratory Report NAC-37, Teddington, UK, 1974.

[23] D. Goldfarb, "A Family of Variable Metric Methods Derived by Variational Means", *Mathematics of Computation*, Vol. 24, pp. 23-26, 1970.

[24] Hewlett-Packard Company, *HP-32E Owner's Handbook*, Corvallis, Oregon, 1978.

[25] D. M. Himmelblau, "A Uniform Evaluation of Unconstrained Optimization Techniques", in F. A. Lootsma, ed., *Numerical Methods for Non-linear Optimization*, pp. 69-98, Academic Press, London, 1972.

[26] R. Hooke and T. A. Jeeves, "Direct Search Solution of Numerical and Statistical Problems", *J. ACM*, Vol. 8, pp. 212-229, 1961.

[27] H. Y. Huang, "Unified Approach to Quadratically Convergent Algorithms for Function Minimization", *J. Optimization Theory and Appl.*, Vol. 5, pp. 405-423, 1970.

[28] S. A. Lill, "A Survey of Methods for Minimizing Sums of Squares of Nonlinear Functions", in L. C. W. Dixon, ed., *Optimization in Action*, pp. 1-26, Academic Press, London, 1976.

[29] S. S. Liu, C. H. Fu, G. E. Atwood, J. Langston, E. Hazani, H. Dun, I. Beinglass, S. Sachdev, and K. Fuchs, "HMOS III Technology", in *1982 Int. Solid-State Circuits Conf., Dig. Tech. Papers*, pp. 234-235.

[30] F. A. Lootsma, ed., *Numerical Methods for Non-linear Optimization*, Academic Press, London, 1972.

[31] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters", *SIAM J. Appl. Math.*, Vol. 11, pp. 431-441, 1963.

[32] G. Merckel, J. Borel, and N. Z. Cupcea, "An Accurate Large-Signal MOS Transistor Model for Use in Computer-Aided Design", *IEEE Trans. Electron Devices*, vol. ED-19, pp. 681-690, 1972.

[33] R. S. Muller, and T. I. Kamins, *Device Electronics for Integrated Circuits*, John Wiley and Sons, New York, 1977.

[34] J. J. McKeown, "Specialised versus General-Purpose Algorithms for Minimising Functions that are Sums of Squared Terms", *Mathematical Programming*, Vol. 9, pp. 57-68, 1975.

[35] J. A. Nelder and R. Mead, "A Simplex Algorithm for Function Minimization", *Computer Journal*, Vol. 7, pp. 308-311, 1965.

[36] R. D. Pashley, W. H. Owen, K. R. Kokkonen, R. M. Jecmen, A. V. Ebel, C. N. Ahlquist, and P. Schoen, "A High Performance 4K Static RAM Fabricated with an Advanced MOS Technology", in *1977 Int. Solid-State Circuits Conf., Dig. Tech. Papers*, pp. 22-23.

[37] E. Polak, *Computational Methods in Optimization: A Unified Approach*, Academic Press, New York, 1971.

[38] M. J. D. Powell, "An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives", *Computer Journal*, Vol. 7, pp. 155-162, 1964.

[39] G. A. Richards, "The Application of Optimisation Techniques to Bi-polar Transistor Modelling", *Proc. IEE Conf. Optimisation Techniques in Circuit and Control Applications*, pp. 51-56, 1970.

[40] H. Shichman and D. A. Hodges, "Modeling and Simulation of Insulated-Gate Field-Effect Transistor Switching Circuits", *IEEE J. Solid-State Circuits*, vol. SC-3, pp. 285-289, 1968.

[41] T. Shima, T. Sugawara, S. Moriyama, and H. Yamada, "Three-Dimensional Table Look-Up MOSFET Model for Precise Circuit Simulation", *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 449-454, 1982.

[42] R. M. Swanson, and J. D. Meindel, "Ion-Implanted Complementary MOS Transistors in Low-Voltage Circuits", *IEEE J. Solid-State Circuits*, vol. SC-7, pp. 146-153, 1972.

[43] R. R. Troutman, "Subthreshold Slope for Insulate Gate Field-Effect Transistors", *IEEE Trans. Electron Devices*, vol. ED-22, pp. 1049-1051, 1975.

[44] A. Vladimirescu and S. Liu, *The Simulation of MOS Integrated Circuits using SPICE2*, Electronics Research Laboratory, Memorandum ERL M80/7, University of California, Berkeley, 1980.

[45] P. P. Wang, "Device Characteristics of Short-Channel and Narrow-Width MOSFET's", *IEEE Trans. Electron Devices*, vol. ED-16, pp. 779-786, 1978.

[46] D. Ward, and A. Doganis, "Optimized Parameter Extraction", MOS Modeling Workshop, IEEE Solid-State Circuits and Technology Committee Meeting, San Francisco, Feb. 9, 1982.

[47] M. H. White, F. Van de Wiele, and J. P. Lambot, "High-Accuracy MOS Models for Computer-Aided Design", *IEEE Trans. Electron Devices*, vol. ED-27, pp. 899-906, 1980.

[48] L. D. Yau, "A Simple Theory to Predict the Threshold Voltage of Short-Channel IGFET's", *Solid-State Electronics*, Vol. 17, pp. 1059-1063, 1974.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: MIT/LCS/TR-277	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): Efficient Modeling for Short Channel MOS Circuit Simulation.	5. TYPE OF REPORT & PERIOD COVERED: Technical Report Aug. '82
	6. PERFORMING ORG. REPORT NUMBER: MIT/LCS/TR-277
7. AUTHOR(s): Mark Griffin Johnson	8. CONTRACT OR GRANT NUMBER(s): DARPA N00014-75-C-0661
9. PERFORMING ORGANIZATION NAME AND ADDRESS: MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, Ma. 02139	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: DARPA, 1400 Wilson Blvd., Arlington, Va. 22217	12. REPORT DATE: August 1982
	13. NUMBER OF PAGES: 91
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office): Office of Naval Research, Department of the Navy, Information Systems Program, Arlington, Va. 22217	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document is approved for public sale and release, distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Unlimited

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

See back

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

See Back

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Efficient Modeling for Short Channel MOS Circuit Simulation

by

Mark Griffin Johnson

Submitted to the Department of Electrical Engineering and Computer Science
on August 13, 1982 in partial fulfillment of the requirements for the
Degree of Master of Science

Abstract

Existing circuit models for short-channel MOS transistors represent a compromise between computation speed and ease of use. Empirical models are very fast to evaluate, but their parameters must be fitted from experimental measurements. Theoretical models require longer computation time, but they may be used to predict the performance of new, unmeasured MOS technologies since their parameters are not curve-fitted from experimental data.

This thesis combines the best features of both types of model, yielding a fast circuit simulator whose input parameters need not be extracted from experimental mensurements. A nonlinear optimization algorithm is used to "compile" the parameters of a theoretical model into parameters for an empirical model, providing the superior user-interface of theoretical models without sacrificing simulator execution speed. Results produced by a prototype model compiler are presented, showing the modeling error to be approximately 5 percent.

Thesis Supervisor: Stephen A. Ward

Title: Associate Professor of Computer Science and Engineering

Key words and phrases: MOS Transistor Modeling, Numerical Optimization, Nonlinear Parameter Estimation.