Adaptive Analog VLSI Signal Processing and Neural Networks

A Thesis
Presented to
The Academic Faculty

by

Jeff Dugger

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
November 2003

Adaptive Analog VLSI Signal Processing and Neural Networks

Approved by:

Professor Paul Hasler, Advisor

Professor Steve DeWeerth

Professor David Anderson

Professor Dieter Jaeger
(Emory University)

Professor Mark Clements

Date Approved 24 November 2003

To my father, Don Dugger, and to my mother, Shirley Dugger.

PREFACE

While the digital world frantically pursues ever-faster clock speeds to satisfy demanding signal processing applications, a quiet revolution in analog computing has been brewing, which promises to do more with less — more sophisticated signal processing delivered at less power in a smaller space. Novel application of a digital memory technology, the floating-gate MOSFET (used in EEPROMs), as an analog memory and computation device provides the basic building block of this technology. Utilization of inherent device physics provides the adaptivity and programmability needed to realize compact reconfigurable analog VLSI systems. Floating-gate charge storage provides non-volatile memory for a matrix of coefficients, while the nonlinear current-voltage relation of the MOSFET provides signal-coefficient multiplication. Summation of products is achieved simply using Kirckhoffs Current Law. Matrix coefficients adapt according to a correlation learning rule which utilizes physical device phenomena (electron tunneling and hot-electron injection) to program floating-gate charge. All of this functionality costs only four transistors per coefficient, each operating at nanowatts of power consumption. The resultant adaptive analog matrix-vector operations form the core of a novel analog VLSI signal-processing model, which is called computing in memory. Peripheral circuitry determines learning behavior, controls programmability, and expands core matrix functionality.

ACKNOWLEDGEMENTS

I wish to thank my colleagues in the Integrated Computational Electronics lab for their encouragement and support, particularly Venkatesh Srinivasan for assistance with the design and construction of the adaptive test board, as well as producing some of the simulation results in Chapter 6.

TABLE OF CONTENTS

DEDICATION . . . . . **iii**

PREFACE . . . . . **iv**

ACKNOWLEDGEMENTS . . . . . **v**

LIST OF TABLES . . . . . **ix**

LIST OF FIGURES . . . . . **x**

SUMMARY . . . . . **xix**

I INTRODUCTION TO ADAPTIVE ELECTRONIC SYSTEMS **1**

1.1 Basic Neural Network Theory 2

1.1.1 Feedforward Computation 3

1.1.2 Adaptation 8

1.2 VLSI Implementations of Neural Networks 9

1.2.1 Neuron Design 10

1.2.2 Synapse Design 11

1.2.3 Analog vs. Digital Circuits 13

1.2.4 Memories for VLSI Neural Networks 14

1.2.5 Floating-Gate Technology for VLSI Neural Memories 15

II FLOATING-GATE SYNAPSE FUNDAMENTALS **17**

2.1 Floating-Gate Transistor Basics 18

2.2 Charge Adaptation through Electron Tunneling and Hot-Electron Injection 20

2.2.1 Electron Tunneling 20

2.2.2 Hot-Electron Injection 22

2.3 Investigating Signal-Adaptive Behavior in Floating-Gate Circuits 23

2.3.1 Separation of Timescales 24

2.3.2 Fast-Timescale Behavior 25

2.3.3 Slow Timescale Behavior 26

2.3.4 The Autozeroing Floating-Gate Amplifier 28

2.3.5 Source-Degenerated pFET Devices: Modeling and Behavior 30

2.4 Concluding Comments 35

III THE FLOATING-GATE PFET CORRELATION LEARNING RULE 36

3.1 Weight Update for the Continuously Adapting pFET Synapse 36

3.2 Effects of Drain Voltage on the Equilibrium Weight 37

3.3 Effects of Gate Voltage on the Equilibrium Weight 39

3.4 Equilibrium Weight is Determined by Correlations Between Gate and Drain Voltages 41

3.5 Hebbian Learning Rule from Approximated Weight Dynamics 43

3.6 Concluding Comments 44

IV FROM FLOATING-GATE SYNAPSES TO FLOATING-GATE NODES 46

4.1 A Simple Two-Input Floating-Gate Node 47

4.1.1 Simple Floating-Gate Learning Node Experiments 50

4.1.2 Learning a Square Wave from Sinusoidal Inputs 51

4.2 Removal of Non-Ideal Effects in the Correlation Learning Rule 53

4.2.1 Harmonic Distortion, Gate Variance, and Gate Pre-Distortion . . . 54

4.2.2 Drain Variance Effects, Drain Pre-Distortion, and Signal Correlations 57

4.2.3 Cancelling Constant Offset and Gate Variance in the Weight 59

4.3 Concluding Comments 61

V GENERAL LEAST-MEAN-SQUARES LEARNING IN FLOATING-GATE NODES 62

5.1 Issue of Weight Decay for Physical LMS computation structures 62

5.2 The LMS Floating-Gate Synapse Circuit 66

5.2.1 Feedforward Synapse Computation 67

5.2.2 Weight Adaptation and the Floating-Gate Correlation Learning Rule 70

5.2.3 Amplitude Correlation Experiments 73

5.3 The Least-Mean-Square (LMS) Learning Rule 76

5.4 The Two-Input Adaptive Node 77

5.5 Concluding Comments 81

VI FROM SIMPLE NODES TO ADAPTIVE NETWORKS 82

6.1 The n-Input Node 82

6.2 A Multiple-Input/Multiple-Node Adaptive Floating-Gate Integrated Circuit 84

6.2.1 Hardware Test and Development System 85

6.3 All-Transistor Synapse Circuit Model and Simulation 87
6.4 Fourier Series Example 92
6.5 Example Application: Adaptive Channel Equalization 93
6.6 Concluding Comments 95
VII CONCLUSIONS 96
APPENDIX A — OJA'S RULE DERIVATION 98

LIST OF TABLES

Table 1 Tunneling parameter, V_x, versus IC fabrication process. 22

Table 2 Definition of constants in equations (26) and (30) 32

LIST OF FIGURES

Figure 1 Classic picture of a two-layer neural network from the perspective of implementating these networks in hardware. The neural networks are layers of simple processors, called neurons, interconnected through weighting elements, called synapses. The neurons aggregate the incoming inputs (including a threshold or offset) and are applied through a $\tanh(\cdot)$ nonlinearity. The synapse elements, which in general are far more numerous than neuron elements, must multiply the incoming signal by an internally stored value, called the weight, and must adapt this weight based upon a particular learning rule. Learning rules implemented in silicon are typically functions of correlations of signals passing through each synapse processor. 2

Figure 2 Typical architectures for neural network implementations. Although the routing looks complicated in Fig. 1, it can be easily implemented in a mesh architecture. Diagram of the Classic Mesh Architecture, typically used for fully-connected systems. 3

Figure 3 Typical architectures for neural network implementations. Although the routing looks complicated in Fig. 1, it can be easily implemented in a mesh architecture. Diagram of a Mesh processor architecture optimized for nearest-neighbor computations. 4

Figure 4 Learning in a single layer. We can build either supervised algorithms (LMS is explicitly shown) or unsupervised one-layer networks in this architecture. 5

Figure 5 Building a multilayer architecture from one-layer mesh structures. Multiple layers can be directly combined to form multilayer neural networks; each layer will rotate 90 degrees from the previous layer. 6

Figure 6 Possible architectures for adaptive multilayer neural networks. Implementation for Backpropagation networks. There are many forms and modifications, but from an implementation viewpoint, these approaches can be modified towards this architecture. This approach significantly increases synapse size, because one typically requires the complexity of two synapses for weight feedback. Further, this approach limits some circuit approaches to building dense synapses. The output from the hidden layer, or layer 1, is $\mathbf{y}_h$ and the error signal given to the hidden layer is $\mathbf{e}_h$. The synapses in the second layer must also output a current proportional to the product of the error and the stored weight; the sum of these currents along a column is the error for the next layer. As a result, the synapses on the second layer must be more complex. 7

Figure 7 Possible architectures for adaptive multilayer neural networks. Implementation using Helmholtz machine concepts. This approach requires twice as many synapses for all but the first layer, which yields the same complexity as the Backpropagation approaches. This approach will converge to the same steady states, only requires a modular tiling of single layer networks, and its reciprocal feedback has a similar feel to communication between layers of cortical neurons. 8

Figure 8 Dynamic storage of analog values on capacitors. (a) Circuit schematic illustrating voltage storage on a capacitor and leakage current through a pn junction. (b) Plot of the stored voltage versus time. The voltage decreases linearly over time since the leakage current is constant. 14

Figure 9 Layout, cross section, and circuit diagram of the floating-gate pFET in a standard double-poly $n$well MOSIS process. The cross section corresponds to the horizonatal line slicing through the layout (or top) view. The pFET transistor is the standard pFET transistor in the nwell process. The gate input capacitively couples to the floating-gate by either a poly-poly capacitor, a diffused linear capacitor, or a MOS capacitor, as seen in the circuit diagram (not explicitly shown in the other two figures). We add floating-gate charge by electron tunneling, and we remove floating-gate charge by hot-electron injection. The tunneling junction used by the single-transistor synapse is a region of gate oxide between the polysilicon floating-gate and $n$well (a MOS capacitor). Between V_{tun} and the floating-gate is our symbol for a tunneling junction, a capacitor with an added arrow designating the charge flow. 18

Figure 10 The tunneling junction is the capacitor between the floating gate and the $n$well (a MOScap); we use high-quality gate oxide to reduce the effects of electron trapping. Over a wide range of oxide voltage, most of the tunneling occurs between the floating gate and n^+ diffusion region because this region is accumulated and the higher electric fields at the corner of the floating gate. 20

Figure 11 Electron tunneling current versus 1/oxide voltage in a 2.0μm process with 42nm gate oxide. The two straight line fits are to the classic Fowler-Nordheim expression in (6). The two different straight-line regions might be due to tunneling through intermediate traps, or due to initially tunneling through the junction edge for low oxide voltages and tunneling through the middle of the junction for high oxide vo ltages. 21

Figure 12 Band diagram of a subthreshold pFET transistor under conditions favorable for hot-electron injection. E_{ox} is the Si–SiO_2 barrier, which is 3.1eV for no field across the oxide. 23

Figure 13 pFET hot-electron injection. Measured data of pFET injection efficiency versus the drain-to-channel voltage for four source currents. Injection efficiency is the ratio of injection current to source current. The injection efficiencies are nearly identical for the different source currents; therefore, they appear to be indistinguishable on the plot. At Φ_{dc} equal to $8.2V$, the injection efficiency increases by a factor of e for an increase in Φ_{dc} of $250mV$. 24

Figure 14 The autozeroing floating-gate amplifier (AFGA). 27

Figure 15 Response of the AFGA to a 1Hz sinewave superimposed on a 19s voltage pulse. The AFGA has a closed-loop gain of 11.2, and a low-frequency cutoff at 100 mHz. We see that the signal is amplified, but the much slower step is adapted away. 29

Figure 16 Minimum and maximum output voltages versus the peak-to-peak output-voltage amplitude. The frequency of the input sine wave was 100Hz; the AFGA had a gain of 146. For small input amplitudes, the minimum and maximum output voltages symmetrically deviate from the steady-state voltage; for large input amplitudes, however, the DC output voltage follows the maximum output voltage. The DC voltage was fit to the function $0.5\ln(\mathrm{I}_0(V_{dc}$ / 1.0V)), which is equal to (25) with V_{inj} =500mV. 31

Figure 17 The source-degenerated (s-d) pFET synapse. The s-d pFET modifies the basic pFET dynamics by local negative-feedback; the device converges to a stable equilibrium for either a constant current or voltage. Circuit diagram of the s-d pFET synapse. The s-d pFET synapse is comprised of a floating-gate pFET synapse and a second ultra -short pFET, which provides feedback to the source terminal. We utilize the DIBL effect in short-channel MOSFETs to build a compact weakly exponential element. 33

Figure 18 The behavior of the current autozeroing circuit using a source-degenerated pFET synapse. Unlike the pFET synapse, this circuit converges to its steady-state current. We use this data to measure weight changes versus time. The tunneling voltage was held at a constant value throughout this experiment; there was non-negligible tunneling and injection current throughout this experiment. 34

Figure 19 Plot of the time-derivative of w versus w for the pFET, and source-degenerated pFET synapses. The arrows show the direction that the differential equation will take. This data shows that the pFET synapse will diverge from the $w = 1$ steady state, but that source-degenerated synapses will stabilize to the $w = 1$ steady state. The s-d pFET modifies the basic pFET dynamics by local negative-feedback. We use zero $\Delta\hat{V}_g$ and zero $\Delta\hat{V}_g$ . . 35

Figure 20 Experimental measurements of floating-gate dynamics from a 0.5μ gate-length process. The gate input is a step decrease in voltage followed later by a step increase in voltage. We used a power supply voltage, $V_{dd} = 5.60V$, and tunneling voltage, $V_{tun} = 15.65V$, to set the operating point for these experiments, as well as the other measurements in this paper. (a) Convergence of the output current of a single synapse back to equilibrium after the step perturbations. The response is nonlinear with asymmetries between tunneling and injection. These nonlinear asymmetries are what allow us to compute correlations. b Time derivative of the weight ($\dot{w}$) vs. weight value (w). We graphically see that weight value converges toward its equilibrium level. 37

Figure 21 Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the drain terminal. The gate terminal is remaining at a fixed voltage through this experiment. $\frac{dw}{dt}$ versus w for three drain input amplitudes. 38

Figure 22 Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the drain terminal. The gate terminal is remaining at a fixed voltage through this experiment. Equilibrium w (w_{eq}) versus drain input amplitude. 39

Figure 23 Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the gate terminal. The drain voltage is remaining at a fixed voltage through this experiment. $\frac{dw}{dt}$ versus w for three gate input amplitudes. 40

Figure 24 Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the gate terminal. The drain voltage is remaining at a fixed voltage through this experiment. Equilibrium weight (w_{eq}) versus gate input amplitude. 41

Figure 25 dw/dt vs. w for varying ΔV_d and fixed ΔV_g. This data gives strong evidence for our learning rule, $\tau' \frac{dw}{dt} = -w + \eta E[xy]$. 42

Figure 26 Change in the dynamical weight equation due to correlated sinusoidal signal amplitudes between the gate and drain terminals. Change in the dynamical weight equation due to different size drain amplitudes for three different gate amplitudes. 43

Figure 27 Change in the dynamical weight equation due to correlated sinusoidal signal amplitudes between the gate and drain terminals. Change in the dynamical weight equation due to various degrees of correlation (phase difference) between sinusoidal input signals at the gate and drain terminals. 44

Figure 28 The differential synapse array enables four quadrant multiplication of weights with inputs. Each floating-gate transistor behaves as a transistor amplifier with an adaptive gain. Adaption occurs through slow-timescale dynamics due to the tunneling and injection process at the floating-gates. We continuously adapt the synapses with the positive signals, and we program the synapses with negative signals to balance the steady-state current of the positive synapse. The network output is the total drain current, and the *learning* signal is the applied drain voltage. By applying the appropriate relationship between output drain current and the resulting drain voltage, we could get a variety of learning rules. Since pFET hot-electron injection is unstable when channel current is the free parameter, we stabilize each synapse element by incorporating DIBL transistors to provide source-degenerative feedback. 47

Figure 29 Experimental measurements of positive synapse current phase correlations. We would program the negative synapses to currents of 5.5μA (w_1^-) and 3.5μA (w_2^-), and therefore both weights are either positive or negative. These results show correlations between a specific gate and global drain terminal direct the convergence of that synapse's weight. 48

Figure 30 Experimental measurements of frequency correlations for sinusoidal inputs to the two-input node given by $V_{g1} = sin(2\pi 3ft)$ and $V_{g2} = sin(2\pi ft)$. The learning signal, $V_d = sin(2\pi f_d t)$, takes on three different frequencies $f_d = 0.7f, f, 3f$. Current outputs from our differential synapses. We programmed our negative weights to offset the positive synapse current measured with input signals with the same signal variance. We see that the synapse that has an identical synapse input frequency as the drain signal has a non-zero weight. 49

Figure 31 Experimental measurements of a square wave learning signal applied to V_d Time course of steady-state currents showing convergence of weights. . . 50

Figure 32 Experimental measurements of a square wave learning signal applied to V_d Spectrum of output current shows amount of each node input frequency matched to frequency components in learning signal. The frequency axis is normalized; the frequency does not affect these results until it approaches the adaptation rate. We obtain 1 and 1/3 for the fundamental and third harmonics as expected. The fifth harmonic appears due to the drain voltage coupling into the floating gate through overlap capacitance. 51

Figure 33 The pre-distort circuit is a simple current mirror which implements $\Delta V_g = V_{mg} \ln(1 + x)$ to eliminate harmonic distortion in the output current. . . 52

Figure 34 The pre-distort circuit for the drain voltage implements $\Delta V_d = V_{md} \ln(1 + y)$ to eliminate drain variance terms in the weight value. 53

Figure 35 This plot shows the DC value (subtracting the equilibrium current) and 2^{nd} harmonic of the drain current vs. gate voltage pre-distort value. The gate pre-distortion is given by $\Delta V_g = V_{mg} \ln(1 + A \sin \omega t)$ for $\Delta V_d = 0$. We find that the appropriate pre-distort factor ($V_{mg} \approx -0.6V$)to be that which makes the 2^{nd} harmonic to be zero and coincides with the non-zero root of $I_{dc} - I_{eq}$. I_{eq} is the value the drain current assumes when there is no signal input. 54

Figure 36 These plots compare the dc values and second-harmonic values of the drain current vs. gate voltage amplitude both with and without pre-distortion. We see that the dc value follows a similar quadratic form in both cases, implying that there is still significant gate variance with pre-distortion for a given pre-distort value. The second-harmonic plot shows that harmonic distortion has been significantly reduced. 55

Figure 37 A plot of DC value of the drain current (weight) vs. drain voltage amplitude for a pre-distorted sinusoid applied to the gate and a non-distorted sinusoid applied to the drain. The gate variance contributes to constant offsets in the data. The quadratic drain variance term masks the linear correlation term. 56

Figure 38 This plot shows the DC value of the drain current vs. drain voltage amplitude when $\Delta V_d = V_{md} \ln(1 + A \sin \omega t)$ with $\Delta V_g = 0$. Here we have plots for several values of V_{md}. We choose that value of $V_{md} = -0.22V$ (corresponding to the flattest curve) as the appropriate distortion factor to eliminate drain variance effects from the weight value. 57

Figure 39 The top figure is a plot of the dc current (weight) minus the equilibrium current value and those current offsets due to gate variance vs. the amplitude of the drain voltage. We see that the quadratic effect due to drain variance has been eliminated. The curves presented have a *tanh* form due to the relationship between the signal amplitude and drain voltage amplitude because of pre-distortion. The bottom figure shows the same data plotted vs. signal amplitude revealing linear correlations. Steeper slopes correspond to larger gate voltages. Thus, we see that $w \propto -E[xy]$, where x and y are sinusoidal signals to be pre-distorted, is verified. 58

Figure 40 In the ideal situation of no mismatch, an adapting differential pair of floating-gate devices would cancel both the gate-variance and constant terms out of the correlation learning rule. 59

Figure 41 Plot of dc current values (weights) minus equilibrium values for two different source-degenerated floating-gate pFETs vs. gate voltage amplitude. Tunneling junction mismatch leads to significant differences in the gate-variance terms in the correlation learning rule. The legend demonstrates the large variation of equilibrium current values also due to mismatch. . 60

Figure 42 The Source-Follower Floating-Gate (SFFG) synapse. 63

Figure 43 Block diagram of a single adaptive floating-gate node. Because the synapse elements far outnumber all of the other components, each synapse element must be as small as possible. Filters extract the output signal (y) from its slow-timescale and DC bias currents. Program circuitry on the periphery allows us to set the initial conditions and / or to restart the weights in a known state when experimenting with various algorithms. 64

Figure 44 Basic block diagram for a weight update using transistors for the weight update element. The result is either non-negligible weight decay or a difficult analytical system to control. Therefore in practice, we would design our weight decay to be small enough for our algorithm but large enough to achieve the desired level of stability. 65

Figure 45 Building a node from a correlating synapse element. Full circuit with required peripheral interface circuitry. Interface circuitry linearizes synapse computation and adaptation. The figure also shows programming circuitry needed to set all of the floating-gate elements. V_{tun1} is the tunneling line for the adaptive and programmed floating-gate elements, where V_{tun2} is the tunneling line for the program only floating-gate elements. 67

Figure 46 10.9513.6=1111The bandwidth of the pre-distort input current mirror limits the speed of the feedforward computation. Bandwith depends on the bias current, , and the total capacitance seen at the node, . One pre-distort mirror feeds the input current to each column of the adaptive matrix, driving the capacitance on m-rows of the matrix. 69

Figure 47 Amplitude correlation results for sinusoids of same frequency and phase with input and error signal amplitudes of 0.3. (a) Synapse output current vs. time. Sinusoidal signals are turned on at 16 seconds and turned off at 48 seconds. (b) Extracted dimensionless weight value vs. time, showing convergence to steady-state due to correlations affected by weight decay (c) Extracted dimensionless signal amplitude vs. time, which follows a path similar to the weight, increasing from zero to a steady-state value. Without an input signal, the output is zero due to multiplicative effects. 73

Figure 48 Basic correlating floating-gate synapse measurements for an individual isolated synapse element. We measured the initial slope from this synapse for multiple input (sinewave) amplitudes near the zero-input steady-state condition. This initial slope shows the correlation function used by this LMS synapse; steady-state solution has less meaning for an LMS synapse with small weight decay. 74

Figure 49 Results for a single-weight LMS experiment. The input signal is a sinusoid of fixed amplitude; the target signal has frequency and phase identical to the input, but varying amplitude. Steady-state weight values for the single-weight LMS experiment are plotted versus target signal amplitude. The steady-state weight value is a linear function of the target signal amplitude. We have plotted the results for several gain levels. 75

Figure 50 Results for a single-weight LMS experiment. The input signal is a sinusoid of fixed amplitude; the target signal has frequency and phase identical to the input, but varying amplitude. Transient solution for a given target signal amplitude of the single-weight LMS experiment shows the output signal for this trial tracking the target signal. 76

Figure 51 Experimental setup for examining Least-Mean-Squares behavior in a two-input node. A scaling operation followed by application of a rotation matrix to an orthogonal signal-space basis of harmonically related sinusoids yields the system input signals; the fundamental sinusoid is chosen as the target. The experiment runs for different values of θ, uniformly sampled from a circle. 77

Figure 52 Two-input Least-Mean-Squares experimental results for the source-follower floating-gate synapse circuit. Measured data for the case $\theta_T = 0$ and $\lambda_1 = \lambda_2$ show steady-state weight dependence on the parameter, θ, of the two-dimensional input mixing-matrix. As expected from equation (84), we get a cosine curve for the first weight, and a sine curve for the second weight. 78

Figure 53 Two-input Least-Mean-Squares experimental results for the source-follower floating-gate synapse circuit. Measured data for the same case is plotted as one steady-state weight versus the other. An ideal LMS rule would produce a circle similar to our circuit results. 79

Figure 54 Two-input Least-Mean-Squares experimental results for the source-follower floating-gate synapse circuit. Illustration of measured data compared with model results computed from equation (83) assuming non-zero constant weight decay for the case where $\lambda_1 = 1$, $\lambda_2 = 2$, and $\theta_T = \pi/3$. The gain of the filter and the amplitude of the input signals determine the scale of the ellipse. The open circles show steady-state values measured from the actual circuit. The asterisks and the corresponding fit-line illustrate computed model results. We observe good agreement between the measured data and the computed values. As predicted in the text, the results form an ellipse. 81

Figure 55 The source-follower floating-gate synapse (SFFG) that was introduced in the preceding chapter. The chip described in the text contains an array of 4 rows and 16 columns of this circuit. 83

Figure 56 Functional diagram of the SFFG array chip described in this chapter. The chip, fabricated through MOSIS in a $0.5\mu m$ process, contains an array with 16 column inputs and 4 row outputs. Additional circuitry around the periphery allows for programming the array as desired. The pre- and post-distort circuitry particular to this design and the SFFG cell appear in the accompanying figures. 84

Figure 57 The pre-distort circuit for each column of the array includes an extra floating-gate pFET for adjusting the charge on the charge on gate of the pre-distort transistor. The nFET mirror provides the input signal to the pre-distort circuit as well as feeds the pFET input to the source-follower bias transistor in the SFFG synapse for the correlation learning rule. . . 85

Figure 58 The post-distort circuit for each row of the array includes a switchable capacitor bank that allows for tuning the post-distort factor. The floating-gate pFET on the far left provides a means for setting the charge on the post-distort circuit gate node. The two pFET transistors and the nFET mirror on the right provide the error signal for the correlation learning rule. 86

Figure 59 Sample-and-hold circuit formed by a capacitor and a unity-gain follower provides analog multiplexing allowing one input pin to drive sixteen synapse inputs. 87

Figure 60 Photograph of the adaptive system test and measurment circuit board. . 88

Figure 61 Functional diagram of the custom circuit board which appears in Fig. 60. Two floating-gate synapse array chips form the heart of the test system. One of the arrays has its floating-gate charge programmed to a bias point, while the second array is allowed to adapt. The programmed array provides a zero-point for for quadrant multiplication. The differential inputs, $+x_n$ and $-x_n$ are applied as a stream of time-division multiplexed samples from the FPGA through the parallel DACS. The target signals for learning, $\hat{y}_m$, are applied through through a serial DAC. The bias signals, including power-supply and tunneling voltages, are also supplied from the DACs. Finally, the output currents for each and programming currents are The signals determining the mode of operation, run and $r\bar{u}n$ are also provided by the FPGA. 89

Figure 62 Integrator-based current-to-voltage converter 90

Figure 63 Schematic of the All-Transistor Synapse (ATS) model of the Source-Follower Floating-Gate (SFFG) synapse 91

Figure 64 Comparison of system output and the square wave target signal for the ATS model Fourier LMS simulation. 92

Figure 65 Fourier spectra comparing the learned system weights with the target signal square wave coefficients. 93

Figure 66 Adaptive channel equalization filter 94

Figure 67 Circuit diagram of a capacitively coupled current conveyer (C^4). This is a bandpass filter with electronically tunable corner frequencies (controlled by $v_{\tau l}$ and $v_{\tau h}$) that can be moved independently of one another. The roll offs are first-order. 95

SUMMARY

Research presented in this thesis provides a substantial leap from the study of interesting device physics to fully adaptive analog networks and lays a solid foundation for future development of large-scale, compact, low-power adaptive parallel analog computation systems. The investigation described here started with observation of this potential learning capability and led to the first derivation and characterization of the floating-gate pFET correlation learning rule. Starting with two synapses sharing the same error signal, we progressed from phase correlation experiments through correlation experiments involving harmonically related sinusoids, culminating in learning the Fourier series coefficients of a square wave [13]. Extending these earlier two-input node experiments to the general case of correlated inputs required dealing with weight decay naturally exhibited by the learning rule. We introduced a source-follower floating-gate synapse as an improvement over our earlier source-degenerated floating-gate synapse in terms of relative weight decay [15]. A larger network of source-follower floating-gate synapses was fabricated and an FPGA-controlled testboard was designed and built. This more sophisticated system provides an excellent framework for exploring applications to multi-input, multi-node adaptive filtering applications. Adaptive channel equalization provided a practical test-case illustrating the use of these adaptive systems in solving real-world problems. The same system could easily be applied to noise and echo cancellation in communication systems and system identification tasks in optimal control problems. We envision the commercialization of these adaptive analog VLSI systems as practical products within a couple of years.

CHAPTER I

INTRODUCTION TO ADAPTIVE ELECTRONIC SYSTEMS

Seeing, walking, and navigating an unknown environment are just a few of the sensory, motor, and cognitive abilities at which biological systems excel, but which are difficult for the best man-made computing systems. While a computer can prove theorems that tax the best human minds, a baby is superior to a digital computer in many respects. For example, a baby is better and faster at visually recognizing objects or faces than the most advanced AI algorithm running on the best supercomputer. Although the digital computer is very effective at giving precise answers to well-defined questions, the nervous system is better at interacting with the real world where sensory data is ambiguous and adaptation to the environment is required. Biological nervous systems have the advantage of being small, compact, and dissipating very little power, which is more than can be said for a supercomputer. They are also robust and fault-tolerant, degrade gracefully, and can learn to solve difficult problems on their own without specialized programming These advantages should inspire engineers to look at how biology has solved difficult sensory perception and motor control problems. Practical engineering examples of learning systems include adaptive filters which are used in cases where the desired signal processing is unknown, unknown and time-varying, or too complicated or costly to model Communication systems provide examples in adaptive channel equalization, and adaptive noise and echo cancellation. Control systems utilize adaptive filters to regulate processes where the system concerned is unknown or too costly to model. Adaptive electronics provide important tools for realizing these systems.

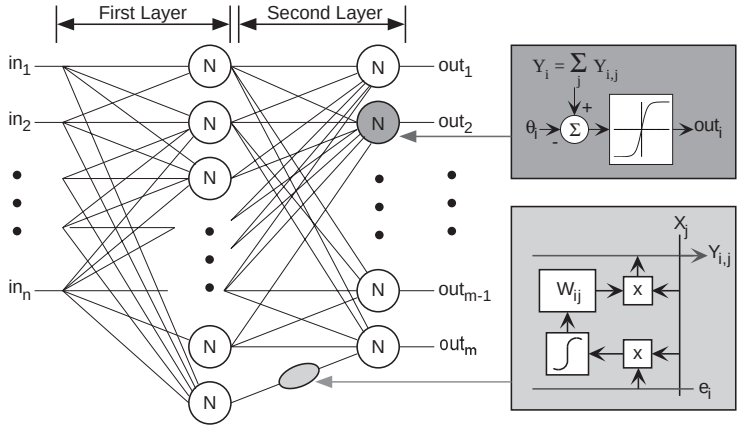

Figure 1: Classic picture of a two-layer neural network from the perspective of implementating these networks in hardware. The neural networks are layers of simple processors, called neurons, interconnected through weighting elements, called synapses. The neurons aggregate the incoming inputs (including a threshold or offset) and are applied through a $\tanh(\cdot)$ nonlinearity. The synapse elements, which in general are far more numerous than neuron elements, must multiply the incoming signal by an internally stored value, called the weight, and must adapt this weight based upon a particular learning rule. Learning rules implemented in silicon are typically functions of correlations of signals passing through each synapse processor.

1.1 Basic Neural Network Theory

Parameterized mathematical models form the basis of many systems studied by engineers and scientists. Scientists and engineers usually derive these models using basic principles and intuitive insight, determining parameters by fitting experimental data to the model. In many cases where difficult to construct from first principles, adaptivity allows a generic parameterized model to tune itself to the data.

Neural networks and adaptive filters represent the best known examples of generic parameterized models where parameters adjust through learning. Both neural networks and adaptive filters comprise a collection of nodes interconnected through a number of synapses. Figure 1 shows the basic feedforward structure typically used in neural network implementations. Most approaches focus on feedforward structures, since feedback systems and networks with time dynamics (e.g. time delays) are straightforward extensions for silicon implementation, although the algorithm design is considerably more difficult. In this model, we encode a neuron's activity as an analog quantity based on the mean spiking rate in a

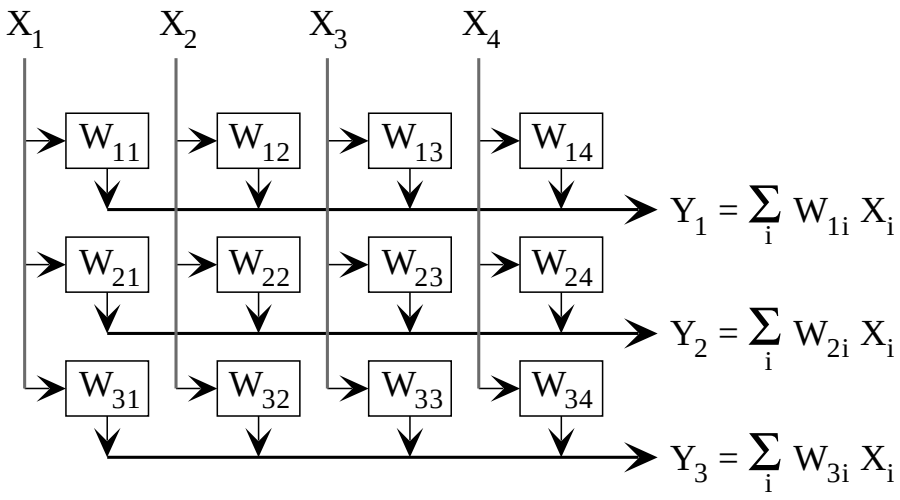

Figure 2: Typical architectures for neural network implementations. Although the routing looks complicated in Fig. 1, it can be easily implemented in a mesh architecture. Diagram of the Classic Mesh Architecture, typically used for fully-connected systems.

given time window. One can build linear or nonlinear filters at the input to the sigmoid function; typically, a low-pass filter is built or modeled since that will naturally occur for a given implementation. This model is excellent for describing biology if only mean-firing rate behavior with minimal dendritic interactions is considered.

Before considering circuit implementations of neurons and synapses, we first frame the overall architecture issues involved in implementing neural networks. In most implementations, a single layer of synapses are built as mesh architectures connected to a column of neuron processors to achieve the functionality illustrated in Fig. 1, because silicon ICs are two-dimensional and require routing schemes that optimally work with two-dimensional constraints.

1.1.1 Feedforward Computation

A basic model synapse must be able to store a weight, multiply its input with the stored weight, and adapt that weight based upon a function of the input and a fed-back error signal. We model feedforward computation mathematically as

$$y_i = W_{ij} x_j \rightarrow \mathbf{y} = \mathbf{W}\mathbf{x} \tag{1}$$

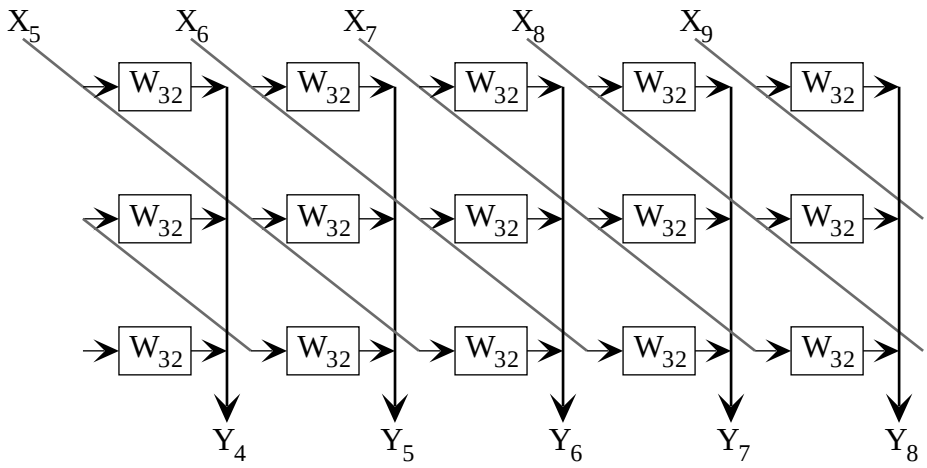

Figure 3: Typical architectures for neural network implementations. Although the routing looks complicated in Fig. 1, it can be easily implemented in a mesh architecture. Diagram of a Mesh processor architecture optimized for nearest-neighbor computations.

where x_j is the j^{th} input ($\mathbf{x}$ is a vector of inputs), y_i is the i^{th} output ($\mathbf{y}$ is a vector of outputs), and w_{ij} is the stored weight at position (i,j) (W is a matrix of weights). The result of this output is passed through a nonlinear function,

$$z_i = \text{out}_i = tanh(a(y_i - \theta_i)) \tag{2}$$

where we designate z_i or out_i as the result of the computation, a is a gain factor, and θ_j is a variable threshold value.

Figure 2 shows the typical implementations for feedforward computation for a single layer architecture. Currents are preferred for outputs, because the summation typically required for most connectionist models is easily performed on a single wire, and voltages are preferred for inputs because they are easy to broadcast. Figure 3 shows how to modify a mesh architecture when considering m-nearest neighbor connections. Figure 5 shows the multilayer architecture from one-layer mesh structures.

Synapses require both feedforward and adaptation computations; therefore architectural constraints imposed by the learning algorithm are an essential consideration for any neural network. Only learning algorithms that scale to large numbers of inputs and outputs are practical. A single layer architecture with a local supervised or unsupervised rule only requires communicating the *error* signal along each row, as seen in Fig. 4. The complexity

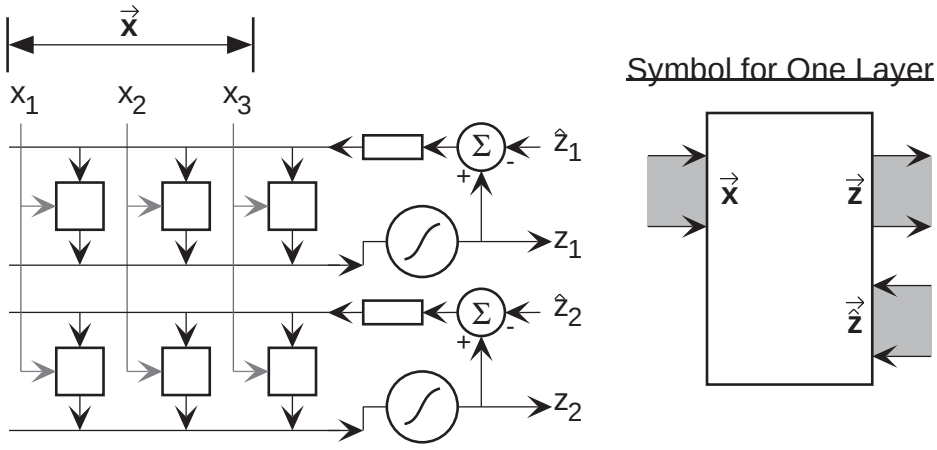

Figure 4: Learning in a single layer. We can build either supervised algorithms (LMS is explicitly shown) or unsupervised one-layer networks in this architecture.

of the synapse computation will depend upon the particular learning rule. Many complicated algorithms, like the generalized Hebbian Algorithm (GHA) [27] and Independent Component Analysis (ICA), require additional matrix-vector multiplications, but can be developed into a mesh architecture. Algorithms requiring matrix-matrix multiplications are not feasible in standard IC technologies.

For multilayer algorithms, the architecture gets more complicated, particularly for supervised algorithms such as multilayer backpropagation. To extend the basic silicon synapse to a backpropagating synapse, we need an additional function; we need an output current that is the product of the fedback error signal (drain voltage) and stored weight. We show this architecture in Fig. 6. This additional function results in two issues, one concerning the signal-to-noise ratio of the resulting error signal, and the other concerning the overall synapse size. The effect of these small error signals, even without the resolution issues, is a slow learning rate.

The neural network literature abounds with possible alternative approaches, but we will base our proposed research on the Helmholtz machine concept [11]. Our primary reason for this approach rests on our desire to use single layer networks as primitives for building larger networks, as well as the fact that this reciprocal adaptive single layer network architecture is seen in various models of sensory neurosystems, such as the pathways from

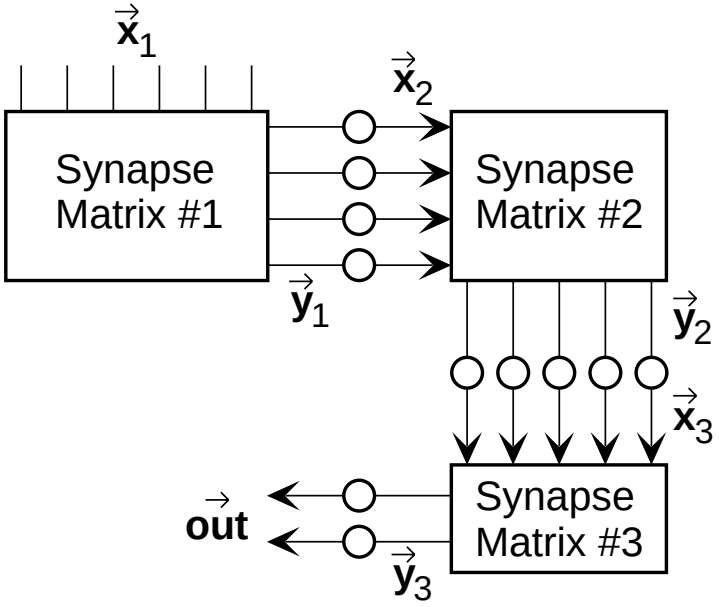

Figure 5: Building a multilayer architecture from one-layer mesh structures. Multiple layers can be directly combined to form multilayer neural networks; each layer will rotate 90 degrees from the previous layer.

Retina to LGN to V1 or some of the pathways between the Cochlea and Auditory Cortex (A1). Figure 7 considers a two layer network implementation of a backpropagation-like learning rule using this Helmholtz block. In this case, we double the number of layers, and therefore double the effective synapse size; for a backprop algorithm, we require the same number of floating-gate multipliers, but with significant additional implementation costs that greatly increase the synapse complexity. This approach seems more IC friendly for the development of adaptive multilayer algorithms than backpropagation approaches, although its digital implementation is nominally equivalent to backpropagation approaches. This approach directly expands to multiple layers and could be used in limited reconfigurable networks because we are building networks with single adaptive layers. Starting with the single-layer network as the basic building block simplifies the abstraction towards system development.

Synapses in previous silicon implementations have required large circuit complexity because they have typically been constructed using traditional circuit building blocks to realize memory, computation, and adaptation functions separately, rather than taking advantage

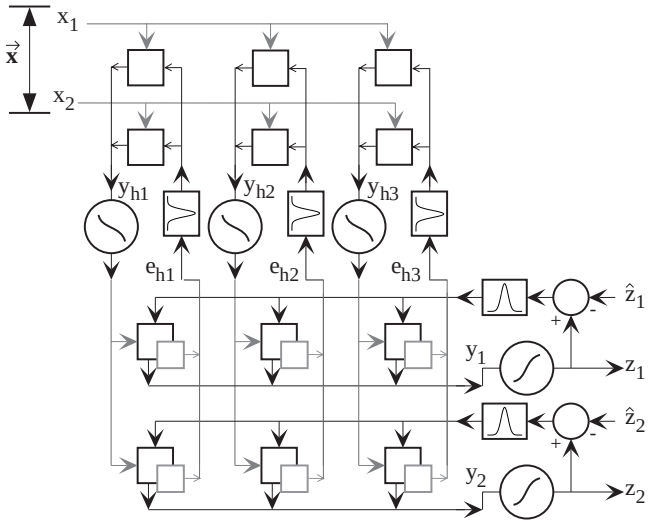

Figure 6: Possible architectures for adaptive multilayer neural networks. Implementation for Backpropagation networks. There are many forms and modifications, but from an implementation viewpoint, these approaches can be modified towards this architecture. This approach significantly increases synapse size, because one typically requires the complexity of two synapses for weight feedback. Further, this approach limits some circuit approaches to building dense synapses. The output from the hidden layer, or layer 1, is $\mathbf{y}_h$ and the error signal given to the hidden layer is $\mathbf{e}_h$. The synapses in the second layer must also output a current proportional to the product of the error and the stored weight; the sum of these currents along a column is the error for the next layer. As a result, the synapses on the second layer must be more complex.

of device physics to combine these functions in a compact circuit element. Not only does large circuit complexity consume tremendous circuit area and power, but the chance of a network operating correctly decreases exponentially with cell size.

The most difficult problem to overcome when building efficient adaptive circuits is the effect of p-n junction leakage currents, illustrated in Fig. 8. First, since many implementations dynamically store their weight parameters on a capacitor, these junction leakage currents typically limit the hold time on the order of seconds; therefore weight storage often becomes a critical concern in many of these applications. Several on-chip refreshing schemes have been proposed and built [27], and are currently finding applications in various ICs [21]. Second, since real-time learning often requires time constants from 10ms to days, junction leakage currents limit the use of capacitor storage techniques, unless prohibitively large capacitor areas are used. Weight update schemes based upon weight perturbation

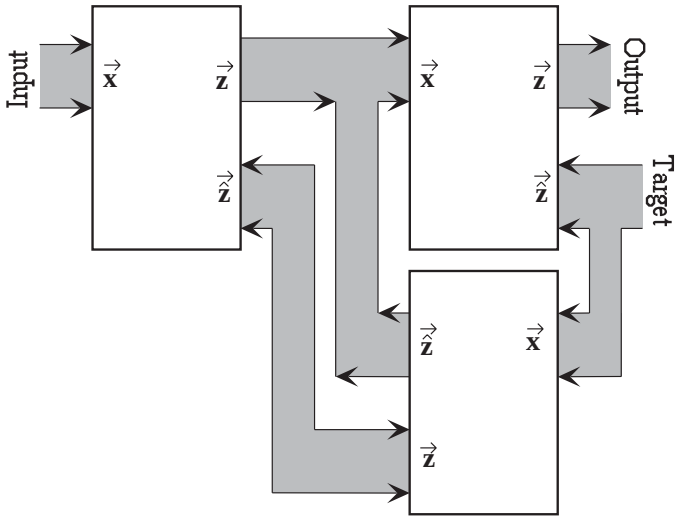

Figure 7: Possible architectures for adaptive multilayer neural networks. Implementation using Helmholtz machine concepts. This approach requires twice as many synapses for all but the first layer, which yields the same complexity as the Backpropagation approaches. This approach will converge to the same steady states, only requires a modular tiling of single layer networks, and its reciprocal feedback has a similar feel to communication between layers of cortical neurons.

methods, i.e. where the error signal is based upon random known changes in the weights, can often work in these constraints if some form of dynamic refreshing schemes are used [21]. Often junction leakage is too large for many adaptive system problems.

1.1.2 Adaptation

We model the weight adaptation mathematically as

$$\tau \frac{d\mathbf{W}}{dt} = f(\mathbf{W}, \mathbf{x}\mathbf{e}^T) \tag{3}$$

where $\mathbf{e}$ is a vector of error signals that is fed-back along various rows. We call this an outer-product learning rule, or a *local* learning rule, because of the $\mathbf{x}\mathbf{e}^T$ computation. The outer-product learning rule is dependent upon the choice of $f(\mathbf{W}, \mathbf{x}\mathbf{e}^T)$ and the choice of the error signal.

Several learning algorithms have been proposed that conform to this functional form (several representive examples can be found in [35, 36]). Learning algorithms usually divide into two categories: supervised and unsupervised. Supervised algorithms adapt the weights based upon the input signals and a supervisory signal to *train* the network to produce

an appropriate response. Unsupervised algorithms adapt the weights based only upon the input signals, and in general the weights are a function of the input statistics. Although these learning algorithms result in very different results, both weight-update rules are similar from an implementation viewpoint. In many supervised algorithms such as backpropagation [35, 36]). this weight change is a time average of the product of the input and some fed-back error signal. Most unsupervised algorithms are based upon Hebbian learning algorithms, which correspond to neurobiological evidence of learning [7]. For a Hebbian synapse, the weight change is a time average of the product of the input and output activity.

1.2 VLSI Implementations of Neural Networks

We face the challenge that we cannot simply duplicate biological models in the silicon media, because the constraints imposed by the biological and silicon media are not identical. Approaches which have been successful begin with the constraints that the silicon medium imposes on the learning system. Therefore letting the silicon medium constrain the design of a system results in more efficient methods of computation.

Parallel computation is an often cited benefit of neural networks for such reasons as speed of simultaneous operation and fault-tolerance to individual component failure [36]. The matrix-vector operations required by neural networks naturally lend themselves to parallel implementations. Ironically, most neural algorithms are implemented on serial microprocessors, where the matrix-vector data for the network is stored in a separate memory, fetched by the processor as needed, processed in pieces, with the results being written back to memory.

Implementation using VLSI circuits provides solutions which take advantage of the inherent parallelism of neural networks. A readily apparent benefit of using parallel implementations is the algorithmic speed-up gained by performing many multiply-adds simultaneously. A more subtle consideration involves time spent in memory accesses and data transfers. The most resource efficient implementation would not only use arrays of parallel processors for computation, but would also use local memory for storing the operating data

in the processors themselves. This concept is known as computation in memory. Computation in memory is more efficient in terms of time, area, and power consumption than traditional computation systems which use separate memory and computation.

In addition, learning algorithms implemented using digital systems are represented by difference equations, whereas algorithms implemented in analog integrated circuits, operate in continuous-time and are therefore described by differential equations. Numerical stability concerns, which plague difference-equation realizations of adaptive algorithms, do not plague analog learning rules. Moreover, all of the analog weights adapt in parallel. Therefore the learning algorithm convergence time depends on the system time constant alone, which is a function of a bias voltage, making the adaptation rate independent of both input signal power and network size.

1.2.1 Neuron Design

To implement a neuron, we need a function that can compute a $\tanh(\cdot)$ function. Fortunately, this function occurs in many IC circuits using either BJT or MOSFET (subthreshold or above-threshold) devices, such as the differential transistor pair [42]. Since we only need a column of neuron circuits, they do not have the same area constraints that are imposed on synapse elements. Dynamics (e.g. low-pass filtering) are usually achieved by adding additional capacitance. Often one needs a current to voltage conversion between the summed synapse outputs and $\tanh(\cdot)$ output, as well as at the output of a differential transistor pair. This conversion often can be nonlinear or needs to be nonlinear to interface with later processing stages.

One could build a more biologically realistic *neuron* element. Several researchers have argued that using pulse computation —which can include mean firing rate computation, pulse-width modulation, and related techniques— would make multipliers, and therefore synapses, denser than analog-valued multipliers [45]. Pulse computation may prove important in biologically inspired implementations that employ event-based processing, since biological systems communicate and compute with action potentials that encode events. Unfortunately, no advantage has yet been presented by using these techniques as opposed

to encoding the number of action potentials or pulse width as an analog quantity, which is the typical analog neural network paradigm.

1.2.2 Synapse Design

Synapses provide the of the computation and adaptation in a learning system, multiplying input signals by adaptive gain parameters, called weights, storing the weights in local memory, and adapting the weights according to some learning algorithm. therefore synapses are the key component of neural networks. Because the synapse is the critical element of any neural network implementation, we previously stated five properties of a silicon synapse which are essential for building large-scale adaptive analog VLSI synaptic arrays [34]:

1. The synapse stores a weight permanently in the absence of learning.
2. The synapse computes an output current as the product of its input signal and its synaptic weight.
3. The synapse modifies its weight using outer-product learning rules
4. The synapse consumes minimal silicon area, thereby maximizing the number of synapses in a given area.
5. The synapse dissipates a minimal amount of power; therefore the synaptic array is not power constrained.

Achieving all five requirements requires detailed discussions on circuits used to implement a synapse. Accordingly, an electronic synapse implementation must realize all of these functions in a compact circuit to afford large networks. Several neural networks have been built as direct mappings of mathematical and computer models of neural networks into analog silicon hardware. A primary issue in synapse circuit designs is developing dense multiplier circuits, because multiplication of an input x weight is fundamental to every synapse. Since an input voltage should modulate an output current, most implementations employ a variable resistance or transconductance element. The approaches include synapses based on

- Fixed resistances, which were the earliest implementations [22],
- Switched-capacitor, charge-coupled devices (CCD), and related implementations [48, 27],
- Gilbert multiplier cells [18],
- Linearized conductance elements [10, 16, 21, 28].

Intel's ETANN chip was the first commercially available neural-network integrated circuit, and the ETANN used floating gates for weight storage [39]. The implementation of the Heuralt–Juetten algorithm by Andreou's group was one of the most successful large-scale adaptive neural systems, but it required a great deal of circuit complexity [9]. Other researchers implemented unsupervised learning and back-propagation algorithms with mixed success [4, 5, 27]. The successful analog implementations of connectionist networks included algorithmic modifications that facilitate its implementation in silicon [19, 27, 47]; history has shown that the success of an implementation is strongly correlated to the degree to which the algorithm is adapted to the silicon medium.

Because most networks directly implemented mathematical models, the synapses in these networks required large circuit complexity. The synapses were large because the implementations used separate memory, computation, and adaptation blocks. Not only does large circuit complexity consume tremendous circuit area and power, but the chance of a network operating correctly decreases exponentially with cell size. Further, the most difficult problem to overcome when building efficient adaptive circuits was the effect of p-n junction leakage currents, illustrated in Fig. 8. Real-time adaptation and learning requires time constants in the range from 10ms to days. Junction leakage currents restrict most integrated-circuit processes to time constants shorter than 1s unless prohibitively large capacitor areas are used. The adaptation rate must be much smaller than the input-signal rate; therefore, the addition of adaptation to a system constrains the minimum computation rate. A system with several levels of adaptation requires the slowest time constant be several orders of magnitude slower than the slowest input-signal rate. Often a 1s leakage rate is far too fast for many adaptive system problems.

In previous neural network implementations, the adaptive elements (synapses) were complex and required a tremendous amount of area. Typical implementations used separate computation, memory, and adaptation blocks, because these networks were direct implementations of mathematical models. Transistor leakage currents limit adaptation time-constants.

1.2.3 Analog vs. Digital Circuits

Much work has been done to implement both analog and digital array processors to exploit the full advantages of parallel neural computation [6]. Although digital implementations of neural networks have been preferred for programmability and flexibility, analog VLSI implementations have several features that make them attractive for neural networks. Analog computational circuits are more compact (measured by chip area or board complexity) and energy efficient than digital computational circuits.

Consider the computation performed by a single node. In some cases, the node output is simply the weighted sum of inputs. In other cases, the node output is a nonlinear function of the weighted sum of inputs. We see that there are two fundamental operations performed by a node, multiplication and addition, and possibly a third, nonlinear function evaluation.

Digital multiplication and summation require many transistors to implement a handful of logic gates to multiply or add each bit. Analog multiplication and summation are much more compact. Analog multiplication can be obtained by exploiting the nonlinear properties of MOSFETs [24]. Analog summation follows easily by application of Kirchoff's current law (KCL). When the analog signals are currents, they can be tied to a common wire, with the resulting current out of that wire being the sum.

While implementation of nonlinear functions is difficult in digital systems, analog implementations provide easy realization of nonlinear functions [3]. Nonlinear functions commonly used in neural networks such as sigmoids and radial-basis type functions can be implemented with only a handful of transistors [42, 12]. Furthermore, digital systems which interface to an analog world require analog-to-digital converters. This necessity further increases the use of resources which could otherwise be devoted to computation.

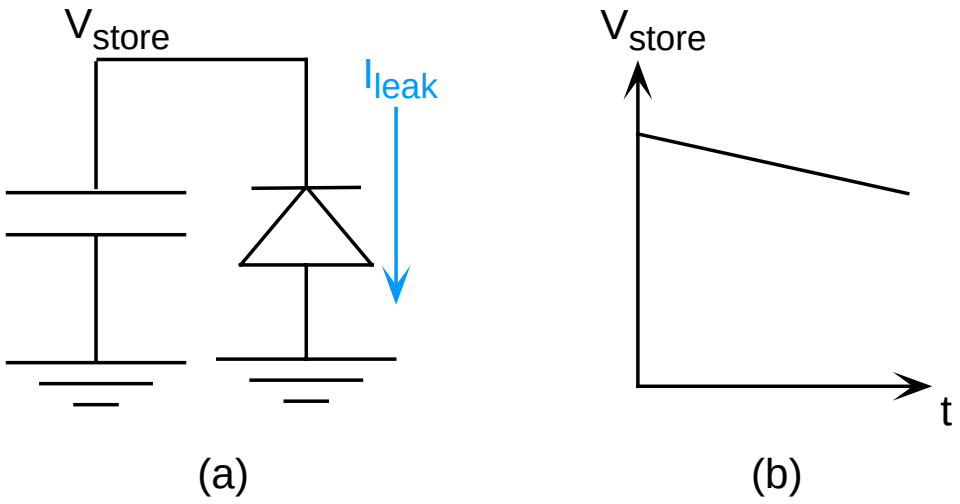

Figure 8: Dynamic storage of analog values on capacitors. (a) Circuit schematic illustrating voltage storage on a capacitor and leakage current through a pn junction. (b) Plot of the stored voltage versus time. The voltage decreases linearly over time since the leakage current is constant.

Aside from chip area, power consumption is another metric used for comparing VLSI circuits. In modern CMOS digital circuits power consumption occurs when current is switched through transistors. Faster digital systems result in more switching per unit time (higher clock speed), and therefore faster digital computation requires more power. The massive heat-sinks on current Pentium chips reflect this fact. In contrast, some experiments suggest that adaptive analog systems can be operated using 10,000 times less power than comparable digital systems [41]. When it comes to area and power consumption, analog circuits have many advantages over digital circuits.

1.2.4 Memories for VLSI Neural Networks

Aside from performing matrix-vector computations, neural network circuits must also store and adapt the elements of the weight matrix. There are two types of storage available, whether the VLSI system is analog or digital: volatile and non-volatile.

Volatile storage is similar to dynamic RAM, where charge is stored on a capacitor connected to the gate of a MOSFET. Because of p-n junction leakage currents through pass transistors used to control charge on these capacitor memories, charge will gradually decay

so that the circuit "forgets" its stored value. These types of memories therefore require frequent value refreshment to avoid losing their data. The effect of p-n junction leakage currents has been the most difficult problem to overcome to build efficient adaptive circuits. Real-time adaptation and learning requires time constants in the range from 10ms to days; because of junction leakage currents, most integrated-circuit processes are restricted to time constants shorter than 1s unless they use prohibitively large capacitor areas.

Non-volatile storage uses floating-gate technology to store parameters indefinitely without refreshment. Floating-gate technology is utilized in Electrically Erasable Programmable Read Only Memory (EEPROM) chips. Figure 9 illustrates a floating-gate MOSFET. A piece of polysilicon completely surrounded by $Si0_2$ forms the floating-gate. Charge on the floating gate is almost permanently stored, since $Si0_2$ is a high-quality insulator. Consequently, once the adaptation is finished, the resulting network state is preserved nearly indefinitely. Input signals are capacitively coupled to the MOSFET channel through the floating gate, and the charge on the floating gate modulates the effect of these signals on the channel current. Non-volatile memories not only have the advantage of retaining their values after the power is turned off, but they also do not require sense amps and other circuitry to refresh values when the power is on. Thus, floating-gate non-volatile memories not only have the advantage of long-term data storage, but are more resource efficient than volatile memory circuits as well.

1.2.5 Floating-Gate Technology for VLSI Neural Memories

Floating-gate technology has been proposed for analog neural network memories for several years [17, 20, 3]. Intel's Electrically Trainable Artificial Neural Network (ETANN) chip was the first commercially available neural-network integrated circuit to use floating gates for weight storage [39], and remains the best VLSI neural network implementation to date. Although this was the first practical analog neural network to use floating-gate storage, there was much concern over its shortcomings. First, most researchers assumed that the specialized EEPROM processes used, which were unavailable to most researchers at the

time, were necessary for programming the synapses [25]. However, we are able to use standard CMOS processes to implement our floating-gate transistor synapses, thus avoiding the need for special processes. Second, the ETANN was limited by off-chip learning. This limitation presents a serious drawback to practical usage of the ETANN chip. Therefore, some researchers have pursued analog on-chip learning in non-volatile floating-gate memories [3], but have remarked on the difficulty of obtaining learning rules using floating-gate circuits. Typically, these neural network hardware implementations have attempted to directly fit mathematical models of neural networks into silicon using standard analog techniques [26].

Our single-transistor floating-gate synapse research goes beyond standard circuit design and exploits device physics, enabling us to remedy the previously asserted lack of a practical update mechanism for floating-gate neural networks. Since the floating-gate voltage modulates a MOSFET's channel current, the floating gate not only serves as a memory, but is also an integral part of computation. Our single-transistor learning synapse not only performs a multiplicative computation with weight storage in a single device, but it also exploits device physics to implement weight adaptation through a correlation learning rule [31, 29, 13]. We take advantage of device physics, specifically, we take advantage of hot-electron injection and electron tunneling, physical processes of transistors that standard CMOS process designers consider a bug rather than a feature. Using tunneling and injection processes allows us to add adaptation to the computation and storage capabilities of the single-transistor floating-gate synapse, providing the key breakthrough necessary to implement practical, large-scale on-chip learning neural systems.

CHAPTER II

FLOATING-GATE SYNAPSE FUNDAMENTALS

While floating-gate transistors have long been utilized for long-term charge storage, and have even been explored as weight storage elements for VLSI neural networks, we now consider floating-gate circuits not only as memory elements, but as continuous-time circuit elements computing at several timescales. We present the single transistor learning synapse and show how floating-gate circuits can adapt based on signals applied to gate and drain. First, we introduce the floating-gate pFET device and demonstrate the nonlinear current-voltage relation that provides synapse computation. Second, because the gate of a floating-gate MOSFET does not possess any conductive leakage paths, adaptive floating-gate circuits must utilize continuous programming currents arising from special physical processes to adapt the floating-gate charge based upon the input signals. In these systems, a quantum physics phenomena, Fowler-Nordheim electron tunneling, provides a bias current for the adaptation process. Signal-dependent adaptation in floating-gate devices derives from the feedback mechanism caused by electrons moving from the transistor channel to the floating gate by hot-electron injection [32, 31]. Discussions of these physical charge-adaptation mechanisms appear in Section 2.2. Finally, having introduced the fundamental physics behind floating-gate device adaptation, we proceed to Section 2.3.1 which develops the mathematical tools with which to analyze the dynamics of pFET floating-gate devices resulting from continuous-time floating-gate currents. These tools include the concept of separation of device behavior into two timescales, one slow and one fast, lead to definitions for weight and computation signal behavior, respectively. The Autozeroing Floating-Gate Amplfiier (AFGA), discussed in Section 2.3.4, provides an example of the first practical circuit to use a single adaptive floating-gate element. Observation of signal-dependent adaptation behavior in this circuit inspired initial interest in the possiblity of obtaining correlation learning rules from these devices. The first floating-gate synapse to

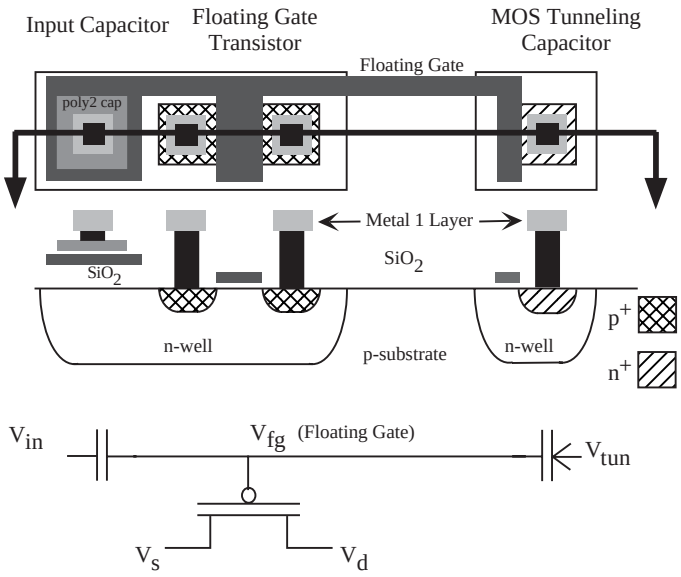

Figure 9: Layout, cross section, and circuit diagram of the floating-gate pFET in a standard double-poly $n$well MOSIS process. The cross section corresponds to the horizonatal line slicing through the layout (or top) view. The pFET transistor is the standard pFET transistor in the nwell process. The gate input capacitively couples to the floating-gate by either a poly-poly capacitor, a diffused linear capacitor, or a MOS capacitor, as seen in the circuit diagram (not explicitly shown in the other two figures). We add floating-gate charge by electron tunneling, and we remove floating-gate charge by hot-electron injection. The tunneling junction used by the single-transistor synapse is a region of gate oxide between the polysilicon floating-gate and $n$well (a MOS capacitor). Between V_{tun} and the floating-gate is our symbol for a tunneling junction, a capacitor with an added arrow designating the charge flow.

illustrate correlation learning, the source-degenerated pFET synapse, resulted from exploration of these possibilities; Section 2.3.5 demonstrates this circuit and its properties. The work presented in this chapter provides the foundation for building dense on-chip learning networks implementing a wide variety of learning algorithms. The floating-gate pFET outer-product learning rule, developed in the following chapter, directly results from the device physics of the floating-gate synapse discussed in this chapter.

2.1 Floating-Gate Transistor Basics

Figure 9 shows the layout, cross-section, and circuit symbol for our floating-gate pFET device. A floating gate is a polysilicon gate surrounded by SiO_2. Charge on the floating gate

is stored permanently, providing a long-term memory, because it is completely surrounded by a high-quality insulator. Signals on the gate terminal of the pFET synapse capacitively couple into the channel by way of the floating gate.

Since we operate our floating-gate synapses in the subthreshold regime, the relationship between channel current, floating gate voltage (V_{fg}), and source voltage (V_s) around a bias current, I_{so}, is given by

$$I_s = I_{s0} e^{-(\kappa \Delta V_{fg} - \Delta V_s)/U_T} e^{-V_{ds}/V_A} \tag{4}$$

where κ is the fractional change in the pFET surface potential due to a change in ΔV_{fg} and U_T is the thermal voltage, $\frac{kT}{q}$, and V_A represents the Early effect. We do not consider the Early effect (channel-length modulation) and assume the transistor is in saturation throughout the remainder of this discussion. For a fixed gate voltage, adding charge to the floating-gate effectively increases the gate voltage seen by the channel and therefore causes the channel current to decrease. Similarly, removing charge from the floating gate causes the channel current to increase. Many of the behaviors extend qualitatively to above-threshold operation; the quantitative behaviors do not.

To begin our analysis of adaptation in floating-gate devices, we consider floating-gate devices that use continuous electron-tunneling and hot-electron injection currents. To model the effect of continuous floating-gate currents, we apply Kirchoff's Current Law to the floating-gate node:

$$C_T \frac{dV_{fg}}{dt} = C_1 \frac{dV_g}{dt} + C_2 \frac{dV_d}{dt} + I_{tun} - I_{inj}, \tag{5}$$

where we have included the effects of electron tunneling (I_{tun}) and hot-electron injection (I_{inj}), C_T is the total capacitance connected to the floating-gate, C_1 is the capacitance between the input and the floating-gate, and C_2 is the capacitance between the drain and the floating-gate. We have fixed the tunneling voltage terminal to a constant bias voltage for this formulation; a change in this would simply add another capacitive term into (5) as well as modify the tunneling current.

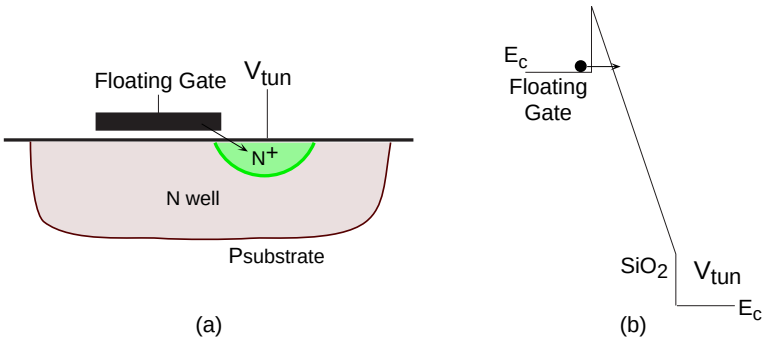

Figure 10: The tunneling junction is the capacitor between the floating gate and the $n$well (a MOScap); we use high-quality gate oxide to reduce the effects of electron trapping. Over a wide range of oxide voltage, most of the tunneling occurs between the floating gate and n^+ diffusion region because this region is accumulated and the higher electric fields at the corner of the floating gate.

2.2 Charge Adaptation through Electron Tunneling and Hot-Electron Injection

Given the update equation (5) derived from Kirckhoff's Current Law at the floating-gate of the pFET in Fig. 9, we see that we need to understand the tunneling and injection processes to fully understand adaptation. In this section we consider terminal-voltage dependent current models of electron tunneling and hot-electron injection.

2.2.1 Electron Tunneling

We add charge to the floating gate by removing electrons. We use electron tunneling (Fowler-Nordheim tunneling [38]) to remove electrons from the floating gate. The tunneling junction is schematically represented by a capacitor coupling the tunneling voltage terminal to the floating gate as shown in Fig. 9. The arrow on the capacitor denotes the charge flow. Increasing the voltage across this tunneling capacitor, either by increasing the tunneling voltage (V_{tun}) or decreasing the floating-gate voltage, increases the effective electric field across the oxide, thereby increasing the probability of the electron tunneling through the barrier. Figure 11 shows measured electron tunneling current through our tunneling capacitor versus (1 / applied oxide voltage). Typical values for the oxide field range from $0.75V/nm$ to $1.0V/nm$. We start from the classic model of electron tunneling through

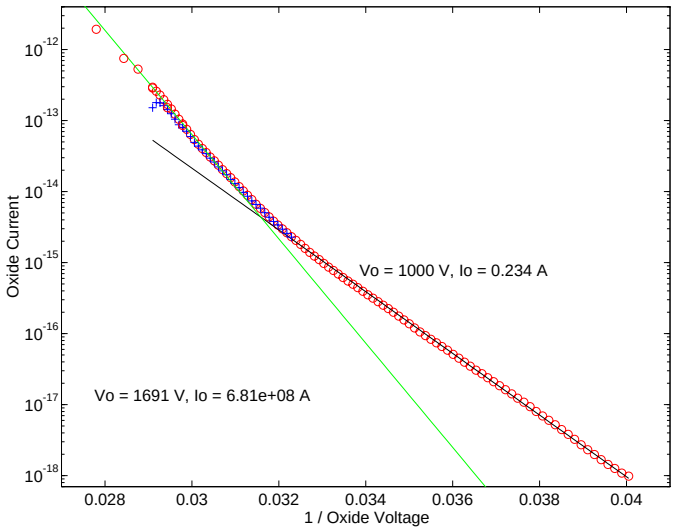

Figure 11: Electron tunneling current versus 1/oxide voltage in a 2.0μm process with 42nm gate oxide. The two straight line fits are to the classic Fowler-Nordheim expression in (6). The two different straight-line regions might be due to tunneling through intermediate traps, or due to initially tunneling through the junction edge for low oxide voltages and tunneling through the middle of the junction for high oxide vo ltages.

a silicon–silicon-dioxide system [26, 38], in which the electron tunneling current is given by

$$I_{tun} = I_0 \exp\left(\frac{\mathcal{E}_o}{\mathcal{E}_{ox}}\right) = I_0 \exp\left(\frac{t_{ox}\mathcal{E}_o}{V_{tun} - V_{fg}}\right), \tag{6}$$

where $\mathcal{E}_{ox}$ is the oxide electric field, t_{ox} is the oxide thickness, and $\mathcal{E}_o$ is a device parameter that is roughly equal to $25.6V/nm$ [40]. The cause for two separate regions might be due to tunneling through intermediate traps [51], or due to initially tunneling through the junction edge for low oxide voltages and tunneling through the middle of the junction for high oxide voltages.

We can approximate the tunneling current for a fixed bias on the tunneling line by

$$I_{tun} = I_{tun0} e^{(\Delta V_{tun} - \Delta V_{fg})/V_x}, \tag{7}$$

where V_x is a parameter related to the quiescent tunneling and floating-gate voltages, ΔV_{tun} is the change in the tunneling voltage, and ΔV_{fg} is the change in the floating-gate voltage from the quiescent floating-gate voltage. Typical values of V_x for various processes at given operating conditions appear in Table 2.2.1.

Table 1: Each IC fabrication process yields a different gate-oxide thickness which determines tunneling current for a given tunneling-junction voltage. Typical values used for V_{tun} given these oxide thicknesses yields the corresponding V_X used in the tunneling current approximation given in 6.

Process	Oxide Thickness	$V_{tun0} - V_{fg0}$	V_x
1.2μm Orbit	30 nm	30 V	1.172 V
0.5μm AMI	11 nm	11 V	0.430 V
0.35μm AMI	8 nm	8 V	0.313 V

2.2.2 Hot-Electron Injection

We use pFET hot-electron injection to add electrons (remove charge) to the floating-gate [34, 26, 28]. We use pFET hot-electron injection, because pFET hot-electron injection can not be eliminated from a CMOS process without adversely affecting basic transistor operation, and therefore will be available in all commercial CMOS processes. One might wonder how pFETs, where the current carriers are holes, inject hot electrons onto the floating gate. Figure 12 shows the band diagram of a pFET operating under bias conditions that are favorable for hot-electron injection. The hot-hole impact ionization creates electrons at the drain edge of the drain-to-channel depletion region, due to the high electric fields there. The hole impact-ionization current is proportional to the pFET source current, and is the exponential of a smooth function (f_1) of the drain-to-channel potential (Φ_{dc}). These electrons travel back into the channel region, gaining energy as they go. When their kinetic energy exceeds that of the silicon–silicon-dioxide barrier, they can be injected into the oxide and transported to the floating gate. We express this relationship as follows:

$$I_{impact} = I_p e^{f_1(\Phi_{dc})}, \tag{8}$$

where Φ_{dc} is the potential drop from channel to drain. The injection current is proportional to the hole impact-ionization current, and is the exponential of another smooth function (f_3) of the voltage drop from channel to drain. We express this relationship as follows:

$$I_{inj} = I_{impact} e^{f_3(\Phi_{dc})}. \tag{9}$$

Figure 13 shows measured injection efficiency for four source currents; injection efficiency is the ratio of the injection current (I_{inj}) to source current (I_s). The measurements for four

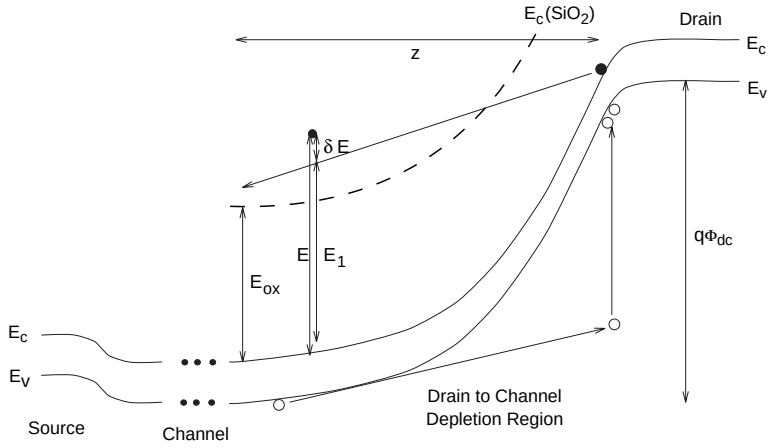

Figure 12: Band diagram of a subthreshold pFET transistor under conditions favorable for hot-electron injection. E_{ox} is the Si–SiO_2 barrier, which is 3.1eV for no field across the oxide.

different source current values are nearly equal, which is consistent with injection efficiency being independent of source current. Injection efficiency is approximately an exponential of a linear function in Φ_{dc} over ranges spanning 1V. The slope of the curve on this exponential scale decreases with increasing Φ_{dc}. Using this linear approximation, we can model the hot-electron injection current for a changing gate and drain-to-source (ΔV_{ds}) voltage as [26]

$$I_{inj} = I_{inj0} \left(\frac{I_s}{I_{s0}} \right)^{\alpha} e^{-\Delta V_d s / V_{inj}}, \tag{10}$$

where V_{inj} is a device and bias dependent parameter, and α is $1 - \frac{U_T}{V_{inj}}$. For a quiescent $\Phi_{dc} = 8.2V$, a typical value for V_{inj} is $250mV$, and for α is 0.90. We have validated this model over several orders of magnitude in current and wide ranges in voltage elsewhere [26, 31].

2.3 Investigating Signal-Adaptive Behavior in Floating-Gate Circuits

The key idea behind signal-adaptive floating-gate circuits is to find cases where input signals can modify the equilibrium value of the floating-gate charge through nonlinearities in the tunneling and injection processes. The first indication that floating-gate devices with continuous tunneling and hot-electron injection currents are capable of adaptation to input signal properties was given by measurements showing floating-gate circuits that adapt

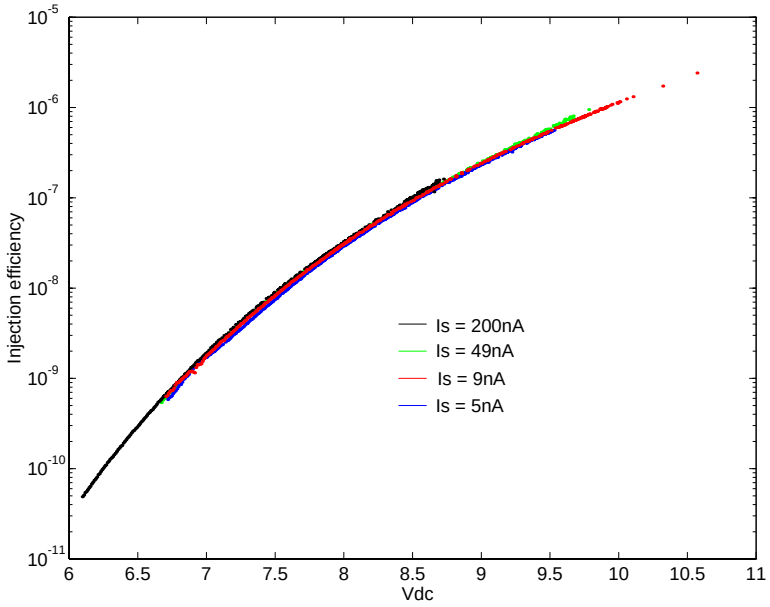

Figure 13: pFET hot-electron injection. Measured data of pFET injection efficiency versus the drain-to-channel voltage for four source currents. Injection efficiency is the ratio of injection current to source current. The injection efficiencies are nearly identical for the different source currents; therefore, they appear to be indistinguishable on the plot. At Φ_{dc} equal to $8.2V$, the injection efficiency increases by a factor of e for an increase in Φ_{dc} of $250mV$.

their equilibrium charge based upon statistics of gate and drain terminal voltage signals. The focus of this section is to develop the mathematics for handling these cases, as well as showing initial circuit approaches and applications. First, we discuss separation of circuit behavior into fast (signal) and slow (adaptive)timescales. Next we present the first practical adaptive floating-gate circuit, the autozeroing floating-gate amplifier (AFGA) and examine its nonlinear adaptive properties. These results will allow us to investigate correlative behaviors in the next chapter. The approach developed here is no different from computing other correlation type functions, say with a multiplier and low-pass filter.

2.3.1 Separation of Timescales

Floating-gate adaptive circuits explore practical uses of continuous electron-tunneling and hot-electron injection currents in floating-gate devices. Our analysis begins by taking a closer look at the floating-gate nodal equation in (5). Because the tunneling and injection

currents in the device are small compared with the transistor bias current, changes in floating-gate charge occur on a much slower timescale than synapse computations. We can therefore model the behavior of this device at two timescales, defining fast-timescale variables to represent rapid changes due to the input signals and slow-timescale variables to represent changes due to adaptation of the floating-gate charge. This analysis decomposes each terminal voltage, V, into three components one due to equilibrium (bias) conditions (V_0), one due to slow timescale signals (same timescale as floating-gate charge adaptation, represented as $\bar{V}$), and one due to fast timescale signals (ΔV). For simplicity, we will collectively define $\tilde{V} = \bar{V} + \Delta V$ to represent both fast and slow timescale signals. We will assume that the applied gate and drain-terminal signal voltages are zero mean — $\bar{V}_g = \bar{V}_d = 0$; that is, they only have fast timescale components, and therefore $\tilde{V}_g = \Delta V_g$ and $\tilde{V}_d = \Delta V_d$. Applying the slow- and fast-timescale decomposition to the floating-gate voltage yields

$$\begin{aligned} V_{fg} &= V_{fg0} + \tilde{V}_{fg} \\ &= V_{fg0} + \bar{V}_{fg} + \Delta V_{fg}. \end{aligned} \tag{11}$$

where V_{fg0} is the equilibrium dc bias voltage. Separation of timescales will allow us to divide the floating-gate node equation (5) into two: one modeling the fast dynamics and one modeling the slow dynamics. For sufficiently fast input signals, this decomposition is justified because the total floating-gate charge is only affected by the floating-gate currents. The consequences of this analysis follow.

2.3.2 Fast-Timescale Behavior

As previously mentioned, we consider fast-timescale variables as representing the computational signals of the synapse; now we define what that means mathematically. Applying the decomposition of V_{fg} to the subthreshold channel current of the floating-gate pFET results in the fast-timescale computation equation given by

$$I_s = I_{s0}(1 + w)e^{-\kappa \Delta V_{fg}/U_T}, \tag{12}$$

where κ is the coupling between the gate voltage and channel potential, and U_T is the thermal voltage; the bias current, I_{s0}, is set by V_{fg0}. The weight of the synapse is defined

as

$$w = e^{-\kappa_p \bar{V}_{fg}/U_T} - 1. \tag{13}$$

We assume this factor, due to the slow-timescale change in floating-gate voltage caused by tunneling and injection currents, to be constant for computation purposes.

Because we assume the tunneling and injection currents to be negligible at computation speeds, we simplify the dynamics in (5) as

$$C_T \frac{d\Delta V_{fg}}{dt} = C_1 \frac{d\Delta V_g}{dt} + C_2 \frac{d\Delta V_d}{dt} \rightarrow \Delta V_{fg} = \frac{C_1}{C_T}\Delta V_g + \frac{C_2}{C_T}\Delta V_d. \tag{14}$$

If we assume $\frac{C_2}{C_T} \approx 0$ and solve for the converged value of ΔV_{fg} then (12) becomes

$$I_s = I_{s0}(1 + w)e^{-\Delta V_g/V_{gA}}, \tag{15}$$

where V_{gA}, defined in Table 2.3.5, represents the voltage scale factor for the external gate voltage. Thus the channel current depends upon a multiplication of an exponential function of the external gate terminal voltage and an adaptive weight scaled by a bias current.

2.3.3 Slow Timescale Behavior

The slow-timescale floating-gate node voltage equation gives us the starting point for exploring the adaptive behavior of the floating-gate synapse. We analyze the adaptive properties of this device by neglecting the fast-timescale signal behavior and assuming the gate and drain voltage signals are zero mean, thus the dynamics in (5) yield

$$C_T \frac{d\bar{V}_{fg}}{dt} = I_{tun} - I_{inj}, \tag{16}$$

Differentiating the weight, w, given in (13) with respect to time

$$\frac{dw}{dt} = -(1 + w)\frac{\kappa}{U_T}\frac{d\bar{V}_{fg}}{dt}. \tag{17}$$

and substituting this for $\frac{d\bar{V}_{fg}}{dt}$ in (16), we obtain the dynamical equations at slow timescales as

$$\frac{U_T C_T}{\kappa}\frac{dw}{dt} = (1 + w)(I_{inj} - I_{tun}). \tag{18}$$

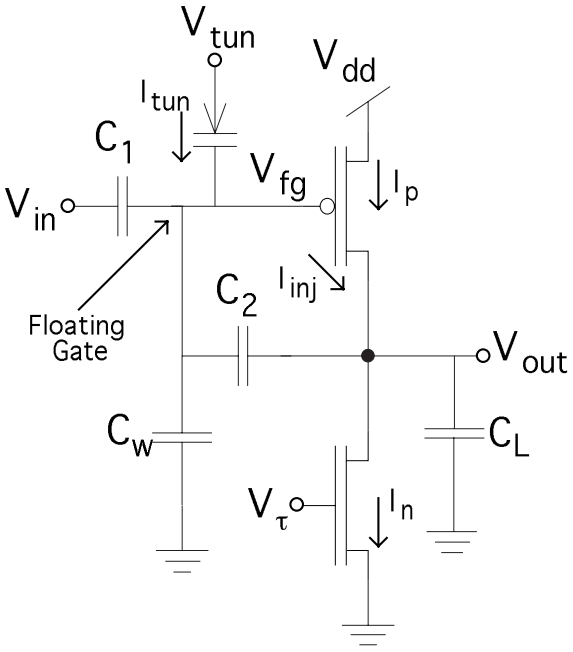

Figure 14: An autozeroing floating-gate amplifier (AFGA) that uses pFET hot-electron injection. The ratio of C_2 to C_1 sets the gain of this inverting amplifier. The nFET is a current source, and it sets the current through the pFET. Steady state occurs when the injection current is equal to the tunneling current. The capacitance from the floating gate to ground, C_w, represents both the parasitic and the explicitly drawn capacitances. Increasing C_w will increase the linear input range of the circuit. The capacitance connected to the output terminal, C_L, is the load capacitance. Between V_{tun} and V_{fg} is our symbol for a tunneling junction, which is a capacitor between the floating-gate and an $n$well.

To obtain this equation, we use separation of timescales and the assumption that the input signals are ergodic, which allows us to use time averages for the expected value. We thus define $E[\cdot]$ as

$$E[x(t)] = \frac{1}{T}\int_0^T x(t)dt \tag{19}$$

where the time interval, T, is long enough to average the fast-timescale signals, but is short enough to assume the slow-timescale variables are nearly constant.

In this chapter, we will investigate the effects of the slow-rate variables, thereby simplifying the equations above. We make this simplification by $\Delta V_d = \Delta V_g = 0$ for sustained periods of time, that is we will apply only step function inputs. In this case, we define the weight as $w = 0$ for no input signals, and therefore it is proportional to the average current at the source. In the following chapter, we will present the complete expansion of these

two timescales; in particular to analyze the effect of input signals on the slow-timescale dynamics and steady states.

2.3.4 The Autozeroing Floating-Gate Amplifier

Offsets often present a difficult problem for designers of MOS analog circuits. A time-honored tradition for addressing this problem is to use a blocking capacitor to eliminate the input DC component; however, for integrated filters, this approach requires enormous input capacitors and resistors to get time constants of less than 1Hz. Existing on-chip autozeroing techniques rely on clocking schemes that compute the input offset periodically, then subtract the correction from the input [49]. These autozeroing techniques add significant complexity to the circuit, as well as clock noise, aliasing, etc. We have developed a bandpass floating-gate amplifier that demonstrates the use of continuous-time, floating-gate adaptation in practical circuit design. This circuit, known as the auto-zeroing floating-gate amplifier (AFGA) and appearing in Fig. 14, uses tunneling and pFET hot-electron injection to set its DC operating point adaptively despite large changes in the DC input voltage. Because electron tunneling and hot-electron injection is an inherent part of floating-gate pFET behavior, we obtain this adaptation with no additional circuitry. Because the gate currents are small, the circuit exhibits a high-pass characteristic with a cutoff frequency less than 1 Hz. The high-frequency cutoff is controlled electronically, as is done in continuous-time filters. We have derived analytical models that completely characterize the amplifier and that are in good agreement with experimental data for a wide range of operating conditions and input waveforms. The autozeroing floating-gate amplifier represents the first practical floating-gate adaptive circuit and clearly illustrates the principles of floating-gate adaptive systems.

For the AFGA circuit in Fig. 14, we assume the floating-gate is fixed, since the signal path from V_{fg} to V_{out} is an ideal high-gain amplifier. At fast timescales, we previously showed that the change in the output voltage is described by

$$\Delta V_{out} \approx -\frac{C_1}{C_2}\Delta V_{in}. \tag{20}$$

Combining this result with the slow timescale behavior of $\bar{V}_{out}$ which we showed earlier, we

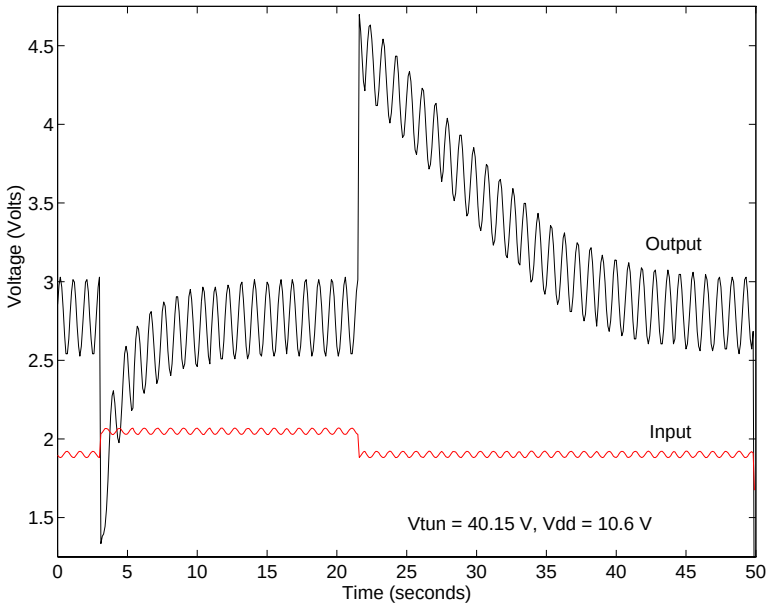

Figure 15: Response of the AFGA to a 1Hz sinewave superimposed on a 19s voltage pulse. The AFGA has a closed-loop gain of 11.2, and a low-frequency cutoff at 100 mHz. We see that the signal is amplified, but the much slower step is adapted away.

get

$$C_2 \frac{d\bar{V}_{out}}{dt} = I_{fg0} \left(E \left[e^{-\frac{C_1}{C_2 V_{inj}} \Delta V_{in}} \right] - 1 \right). \tag{21}$$

At slow timescales, we previously showed that the equation in $\bar{V}_{out}$ is [26]

$$C_2 \frac{d\bar{V}_{out}}{dt} = I_{fg0} \left(E \left[e^{-\Delta V_{out}/V_{inj}} \right] - 1 \right). \tag{22}$$

where $I_{fg0} e^{-\Delta V_{out}/V_{inj}}$ represents the floating-gate current due to hot-electron injection, and $-I_{fg0}$ represents the floating-gate current due to electron tunneling. V_{inj} is a hot-electron injection device parameter.

Figure 16 shows measured minimum and maximum output voltages versus the output signal amplitude for a sine-wave input. For small input amplitudes, the minimum and maximum output voltages deviate symmetrically from the steady-state voltage, but for large input amplitudes, most of the change in the output voltage is due to an increasing maximum output voltage. The steady-state output voltage remains within about V_{inj} of the minimum of the signal. We see that the variance of the fast time-scale input affects the dc operating point of the circuit.

We would like to see if (22) predicts this particular AFGA behavior. First, we need to express $E[e^{-V_{out}/V_{inj}}]$ in terms of the fast and slow variables:

$$E\left[e^{-V_{out}/V_{inj}}\right] = e^{-\bar{V}_{out}/V_{inj}} E\left[e^{-\Delta V_{out}/V_{inj}}\right]. \tag{23}$$

Then, using (23) to rewrite (22), the steady-state solution for $\Delta\bar{V}_{out}$ is [26]

$$\bar{V}_{out} = V_{inj} \ln\left(E\left[e^{-\Delta V_{out}/V_{inj}}\right]\right). \tag{24}$$

The AFGA always adapts its floating-gate charge such that the minimum of the output signal remains at the equilibrium output voltage. Finally, let us consider the output-voltage behavior as a function of the input-signal amplitude for a sinusoidal input. We define the amplified input signal as $\Delta V_{out} = V_1 \sin(\omega t)$; for this output signal, the steady-state voltage is

$$\begin{aligned}\bar{V}_{out} &= V_{inj} \ln\left(\int_{\omega t=0}^{2\pi} e^{-(V_1/V_{inj})\sin(\omega t)} d(\omega t)\right) \\ &= V_{inj} \ln\left(\mathrm{I}_0\left(\frac{V_1}{V_{inj}}\right)\right) \approx \frac{V_1^2}{2V_{inj}},\end{aligned} \tag{25}$$

where $\mathrm{I}_0(\cdot)$ is the modified Bessel function of zeroth order. Figure 16 exhibits experimental measurements that confirm the results of the analysis in (25) and show that fast timescale signal statistics can affect the adaptation of the output offset voltage.

2.3.5 Source-Degenerated pFET Devices: Modeling and Behavior

To use the floating-gate pFET as an adaptive computational element, we wish to configure it as a transconductance amplifier. Unfortunately, in this configuration, with continuous-time tunneling and injection current, the floating-gate pFET dynamics are unstable. The basic pFET synapse is unstable because of a positive feedback relationship that exists between the hot-electron injection current and the channel current. For a step increase in the gate-to-source voltage of the pFET, we get an increase in the channel current. This increases the hot-electron injection current in the pFET. Increasing the hot-electron injection current adds electrons to the floating-gate, which makes the gate-to-source voltage effectively larger, further increasing the channel current. This positive feedback relation does not give us a

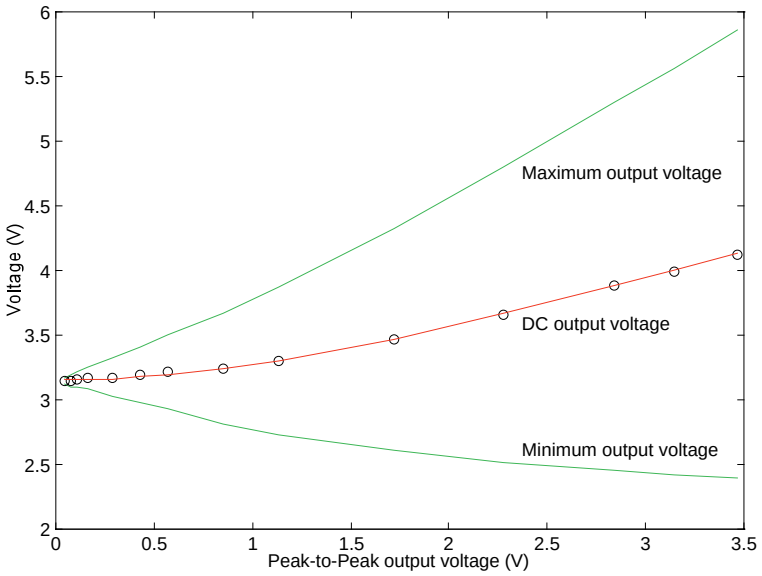

Figure 16: Minimum and maximum output voltages versus the peak-to-peak output-voltage amplitude. The frequency of the input sine wave was 100Hz; the AFGA had a gain of 146. For small input amplitudes, the minimum and maximum output voltages symmetrically deviate from the steady-state voltage; for large input amplitudes, however, the DC output voltage follows the maximum output voltage. The DC voltage was fit to the function $0.5\ln(\mathrm{I}_0(V_{dc}\ /\ 1.0\mathrm{V}\ )\ )$, which is equal to (25) with V_{inj} =500mV.

stable weight value that we can use in analog computations. We show experimental data of this phenomena in Fig. 19.

To obtain a stable continuously-adapting pFET synapse, we introduce external negative feedback by using source degeneration shown in Fig. 17. Figure 17 shows the circuit for the source-degenerated (s-d) pFET synapse [26, 31]. The circuit diagram for the s-d pFET synapse shows that we have added another pFET, M2, between V_{dd} and the source terminal of the floating-gate pFET, M1. M2 is a short-channel pFET with significant drain-induced barrier lowering (DIBL). A transistor that strongly exhibits DIBL shows an exponential change in current for a linear change in drain voltage. The resulting dependence of channel current on the floating-gate voltage for the circuit in Fig. 17 is

$$I_s = I_{s0} e^{-\kappa_x \kappa (\bar{V}_{fg} + \Delta V_{fg})/U_T} \tag{26}$$

where κ_x is a parameter that characterizes the sharpness of the exponential relationship of the source degenerative feedback element.

Table 2: Definition of constants in equations (26) and (30)

$\tau = (C_T U_T)/(\kappa_x \kappa I_{fg0})$	$V_{gA} = \frac{1}{\kappa_x}\frac{C_T}{C_g}\frac{U_T}{\kappa}$
$\gamma = 2 - \frac{1}{\kappa_x}\frac{U_T}{V_{inj}}$	$V_{g0} = V_{gA}/(1-\gamma)$
$\beta = 1 + \frac{1}{\kappa_x}\left(\frac{1}{\kappa}\frac{U_T}{V_x}\right)$	$V_{g1} = V_{gA}/(\beta - 1)$

The stabilizing effects of adding M2 can be seen in Fig. 19. The DIBL current-voltage characteristic of M2 guarantees that the decrease in the source voltage of M1 due to an increase in channel current will be sufficient to provide the negative feedback desired. Increasing the channel current in M2 causes a decrease in the source voltage of M1, this in turn decreases the drain-to-source voltage of M1. Decreasing the drain-to-source voltage of M1 decreases the channel current, so that hot-electron-injection current also decreases. Fig. 19 shows dynamics (measured) to get plots for the s-d pFET in Fig. 19.

Including the model for gate currents, we model the weight dynamics of a pFET floating-gate device as

$$-\frac{U_T C_T}{\kappa I_{fg0}}\frac{dw}{dt} = w^{1+\frac{U_T}{\kappa_p V_x}} - w^{1+\alpha} e^{-\Delta V_d / V_{inj}}. \tag{27}$$

Applying averaging to the tunneling and injection currents in the floating-gate nodal equation and using the definition for the weight in (13) yields the fundamental equation governing the weight dynamics of s-d floating-gate pFET synapse as

$$\tau\frac{dw}{dt} = w^{\gamma} E\left[e^{(\Delta V_g/V_{g0} - \Delta V_d/V_{inj})}\right] - w^{\beta} E\left[e^{-\Delta V_g/V_{g1}}\right] \tag{28}$$

where the various constants are defined in Table 2.3.5. The result of (30) shows how the slow timescale behavior depends on fast timescale signals and provides the basis for the rest of our analysis.

2.3.5.1 Source-Degenerated Synapse Weight Dynamics

To understand the basic form of the weight dynamics, we look at the simplest case of (30) when $\Delta V_g = \Delta V_d = 0$. This allows us to see the dependence of $\frac{dw}{dt}$ on w. Since we are only interested in the linear dynamics around $w_{eq} = 0$, we use the Taylor expansion and keep

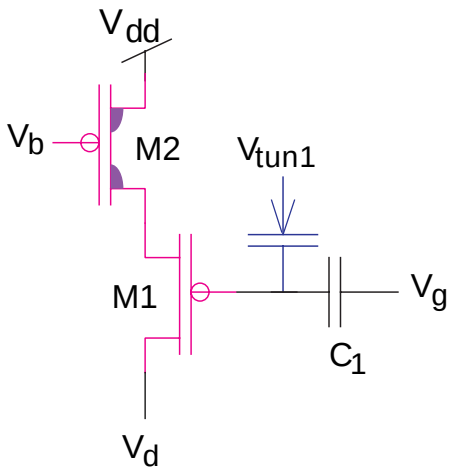

Figure 17: The source-degenerated (s-d) pFET synapse. The s-d pFET modifies the basic pFET dynamics by local negative-feedback; the device converges to a stable equilibrium for either a constant current or voltage. Circuit diagram of the s-d pFET synapse. The s-d pFET synapse is comprised of a floating-gate pFET synapse and a second ultra -short pFET, which provides feedback to the source terminal. We utilize the DIBL effect in short-channel MOSFETs to build a compact weakly exponential element.

only the linear terms of the result to obtain

$$\begin{aligned} \tau \frac{dw}{dt} &= (1 + \gamma w) - (1 + \beta w) \\ &= -(\beta - \gamma) w. \end{aligned} \tag{29}$$

From (29) it is seen that $\beta > \gamma$ yields the stable dynamical equation. This relation between β and γ is determined by the source-degenerated pFET.

For the source degenerated pFET, we experimentally observe the weight dynamics by noting that the weight of the floating-gate synapse is directly proportional to the channel current when the signal voltages (fast timescale) are zero. We measure the channel current indirectly by measuring the source voltage, since the source voltage is logarithmically related to the channel current due to the DIBL FET operating with subthreshold currents. To see the weight dynamics, we apply a slow timescale step to the gate voltage, $\bar{V}_g$, and observe the source voltage, ΔV_s. The gate voltage input and the source voltage response are shown in Fig. 18. We obtain the channel current, and thus w by taking the exponential of this value. A finite difference approximation using successive values of w gives $\frac{dw}{dt}$. The s-d pFET dynamics given by these measurements appears in Fig. 18. Figure 18 also illustrates

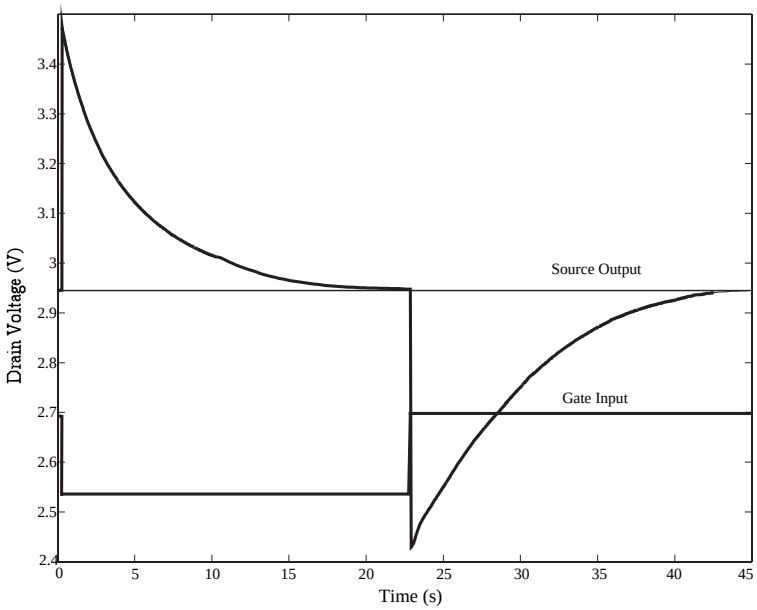

Figure 18: The behavior of the current autozeroing circuit using a source-degenerated pFET synapse. Unlike the pFET synapse, this circuit converges to its steady-state current. We use this data to measure weight changes versus time. The tunneling voltage was held at a constant value throughout this experiment; there was non-negligible tunneling and injection current throughout this experiment.

the dynamics of this equation for the basic floating-gate pFET synapse. We observe there are two equilibrium points for both systems, $w_{eq} = 0$ and $w_{eq} = -1$. We are interested in the weight dynamics around $w_{eq} = 0$, since the weight can not go below zero, and this steady state will not change with different inputs. Figure 19 verifies that the basic pFET synapse is unstable, but the source degenerated pFET is stable.

The functional form of the dynamics illustrated in Fig. 18 also hold when ΔV_g and ΔV_d are non-zero. Non-zero values of these variables will affect the non-zero equilibrium weight, the slope of the dynamics around this equilibrium weight, and the position of the seperatrix between the two equilibrium weights. In this discussion we are mainly interested in the effects of the voltages on the stable-equilibrium weight. In the next chapter, we discuss how signal statistics determine the value of this weight.

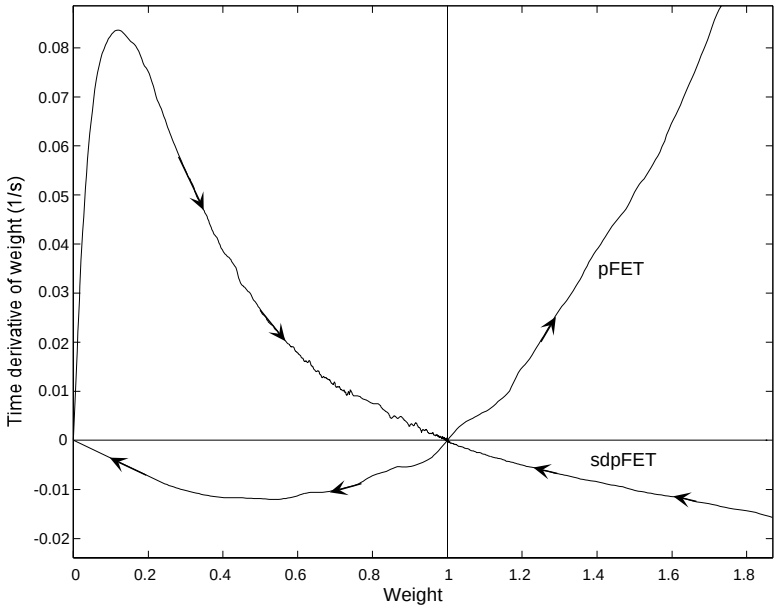

Figure 19: Plot of the time-derivative of w versus w for the pFET, and source-degenerated pFET synapses. The arrows show the direction that the differential equation will take. This data shows that the pFET synapse will diverge from the $w = 1$ steady state, but that source-degenerated synapses will stabilize to the $w = 1$ steady state. The s-d pFET modifies the basic pFET dynamics by local negative-feedback. We use zero $\Delta\hat{V}_g$ and zero $\Delta\hat{V}_g$

2.4 Concluding Comments

By taking advantage of physical processes inherent in the floating-gate pFET device, we provide a compact, low-power circuit element that can not only be utilized as a non-volatile memory cell, but that can also be utilized as an adaptive computational element. In this chapter, we discussed charge adaptation mechanisms based on electron tunneling and hot-electron injection, and developed fundamental mathematical concepts to apply these physical processes in adaptive signal processing systems. The AFGA presented us with the first simple, useful circuit which employs continuous-time adaptation, illustrating these key concepts of adaptive floating-gate circuits. Observation of signal-based adaptation exhibited by the AFGA led to development of the source-degenerated pFET synapse. The following chapter details correlation learning behavior in this novel adaptive circuit.

CHAPTER III

THE FLOATING-GATE PFET CORRELATION LEARNING RULE

When considered as an analog computational element, a floating-gate device computes its output as a product of its input and an adaptive device parameter. We call the adaptive device parameter the weight. When used as an adaptive analog computational element, we call the floating-gate device a synapse, since we will use it to build electronic neural networks. We call these elements synapses because of their relationship to synapses in adaptive filters and neural networks [35], and their loose connection with biological synapses [7]. A fundamental question we would like to answer is how the synapse weight adapts. When initially introduced, single transistor floating-gate synapses, involved complicated weight-update equations based upon the programming physics [34]. This chapter clearly illustrates how the synapse weight adapts based on correlations between voltage signals applied to the gate and the drain terminals of the floating-gate pFET synapse. We focus on pFET floating-gate devices because they are available in any standard CMOS process; although we focus on pFET floating-gate devices, these results are easily extendable to nFET floating-gate devices.

3.1 Weight Update for the Continuously Adapting pFET Synapse

In the last chapter, we noticed that floating-gate circuits can adapt their operating conditions based on terminal-voltage properties and we developed the mathematical tools to describe this phenomenon. We then introduced the source-degenerated floating-gate synapse and derived the weight update equation for it as

$$\tau \frac{dw}{dt} = w^{\gamma} E\left[e^{(\Delta V_g/V_{g0} - \Delta V_d/V_{inj})}\right] - w^{\beta} E\left[e^{-\Delta V_g/V_{g1}}\right]. \tag{30}$$

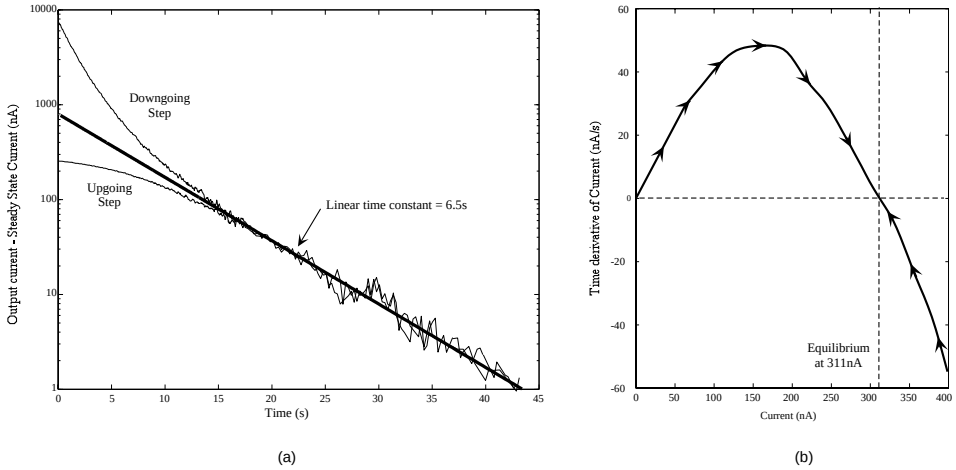

Figure 20: Experimental measurements of floating-gate dynamics from a 0.5μ gate-length process. The gate input is a step decrease in voltage followed later by a step increase in voltage. We used a power supply voltage, $V_{dd} = 5.60V$, and tunneling voltage, $V_{tun} = 15.65V$, to set the operating point for these experiments, as well as the other measurements in this paper. (a) Convergence of the output current of a single synapse back to equilibrium after the step perturbations. The response is nonlinear with asymmetries between tunneling and injection. These nonlinear asymmetries are what allow us to compute correlations. b Time derivative of the weight ($\dot{w}$) vs. weight value (w). We graphically see that weight value converges toward its equilibrium level.

We examined the case with no fast-timescale inputs and observed the stability of this system; the results appear in Figure 20. Now we investigate how statistical properties of the fast-timescale voltages affect the steady-state solution of the source-follower floating-gate synapse. First, we note the effects of drain variance on the steady-state solution. Second, we describe the effects of gate variance. Finally, we present the correlation learning rule which results from (30).

3.2 Effects of Drain Voltage on the Equilibrium Weight

Our approximation of the weight dynamics begins by considering the effects on the equilibrium weight due to drain voltage alone. We start with the weight equation when $\Delta V_g = 0$. The weight dynamics then become

$$\tau \frac{dw}{dt} = w^{\gamma} e^{-\Delta V_d / V_{inj}} - w^{\beta}. \tag{31}$$

Figure 21 shows experimental measurements of $\frac{dw}{dt}$ vs. w for fast timescale drain voltage signals of various amplitudes and no gate voltage signal. We apply a slow timescale step

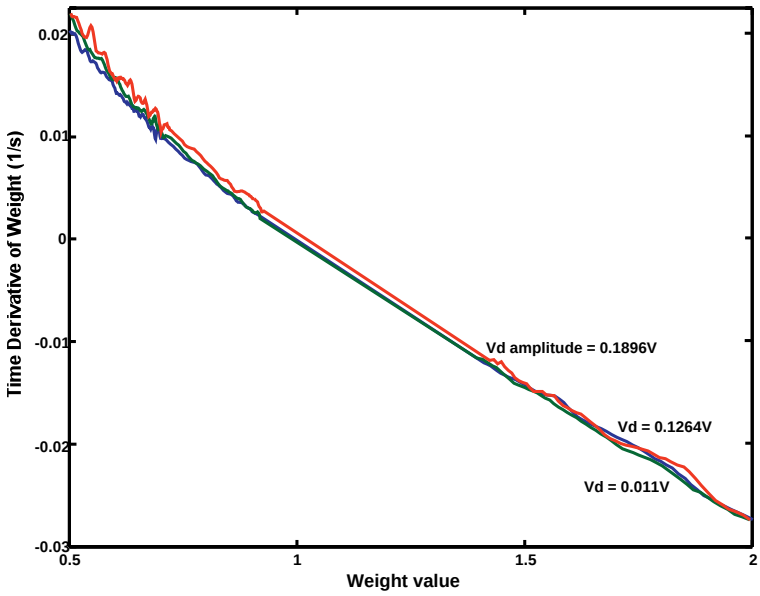

Figure 21: Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the drain terminal. The gate terminal is remaining at a fixed voltage through this experiment. $\frac{dw}{dt}$ versus w for three drain input amplitudes.

input to the gate such that we get a sweep of $\frac{dw}{dt}$ vs. w. We apply a fast timescale input, $\Delta V_d = V_1 \sin(\omega t)$, at various amplitudes to view the effects of fast timescale signals on the weight dynamics. As before, we measure the source voltage to obtain the weight dynamics.

To observe the effects of the fast timescale signals on the weight equilibrium, we take the expected value and solving for w_{eq} we get

$$w_{eq} = E[e^{-\Delta V_d/V_{inj}}]^{1/(\beta-\gamma)} \tag{32}$$

Next we use a quadratic approximation of $e^{-\Delta V_d/V_{inj}}$ by truncating the Taylor series. We use a quadratic approximation because it is the minimum order expansion needed to show correlations in our circuit. Taking the expected value of each term of the approximation yields

$$w_{eq} \approx (1 + \frac{1}{2}E[(\Delta V_d/V_{inj})^2])^{1/(\beta-\gamma)} \tag{33}$$

The linear term for ΔV_d disappears since its expected value is zero. We use the Taylor expansion approximation to eliminate the exponent $1/(\beta-\gamma)$. Solving for w_{eq} gives

$$w_{eq} \approx \frac{1}{2}\frac{1}{(\beta-\gamma)}E[(\Delta V_d/V_{inj})^2]. \tag{34}$$

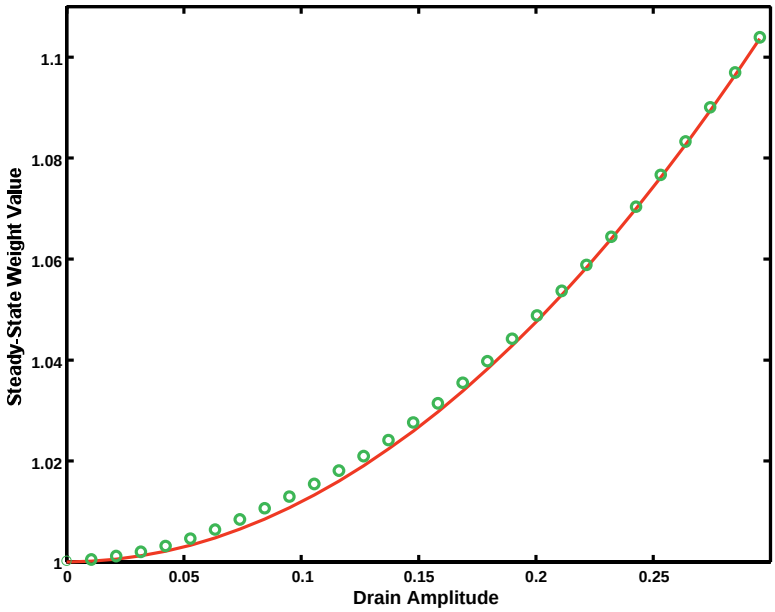

Figure 22: Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the drain terminal. The gate terminal is remaining at a fixed voltage through this experiment. Equilibrium w (w_{eq}) versus drain input amplitude.

The data for this experiment uses a sinusoidal input, $\Delta V_d = V_1 \sin(\omega t)$. Substituting this into (34), we get

$$w_{eq} \approx \frac{1}{4}\frac{1}{(\beta - \gamma)}(V_1/V_{inj})^2. \tag{35}$$

Figure 22 shows w_{eq} vs. V_1 with $V_{inj} \approx 500$ mV. The dependence of w_{eq} on V_1 is quadratic, but the V_{inj} term is large compared to the range of operation so that V_d has no real effect on the weight. Thus the effect of V_d on w_{eq} can be neglected.

3.3 Effects of Gate Voltage on the Equilibrium Weight

Substituting $\Delta V_d = 0$ in the weight equation allows us to observe the effects of gate voltage alone. This results in

$$\tau \frac{dw}{dt} = w^{\gamma} e^{\Delta V_g / V_{g0}} - w^{\beta} e^{-\Delta V_g / V_{g1}}. \tag{36}$$

Figure 23 shows experimental measurements of $\frac{dw}{dt}$ vs. w for fast timescale gate voltage signals of various amplitudes and no drain voltage signal. We get these dynamics using an experimental procedure similar to that used to measure drain voltage effects.

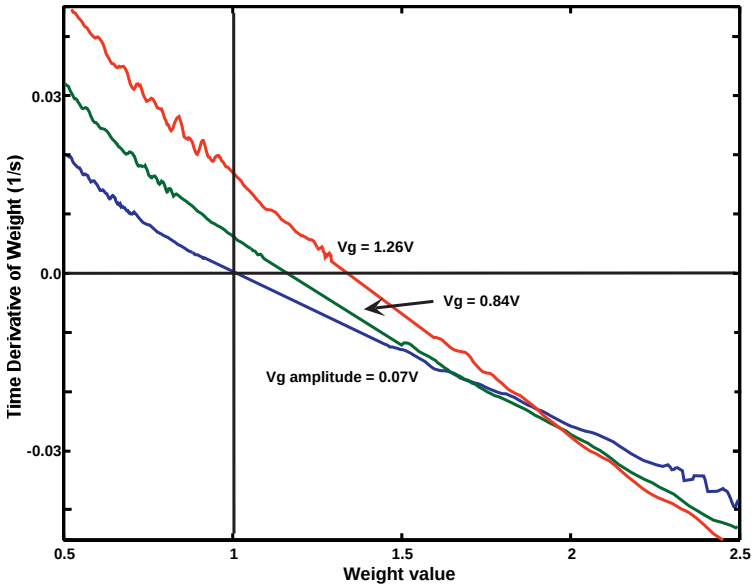

Figure 23: Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the gate terminal. The drain voltage is remaining at a fixed voltage through this experiment. $\frac{dw}{dt}$ versus w for three gate input amplitudes.

As in the drain voltage case, we take the expected value, find the equilibrium weight, and make the quadratic approximation to get

$$w_{eq} \approx \left(\frac{1 + \frac{1}{2} E[(\Delta V_g / V_{g0})^2]}{1 + \frac{1}{2} E[(\Delta V_g / V_{g1})^2]} \right)^{\frac{1}{(\beta - \gamma)}} . \tag{37}$$

Applying two successive binomial expansions and keeping only those terms relevant to the quadratic approximation we simplify the fraction and the exponent to get

$$w_{eq} \approx (1 + \frac{E[(\Delta V_g / V_{g0})^2]}{2(\beta - \gamma)})(1 - \frac{E[(\Delta V_g / V_{g1})^2]}{2(\beta - \gamma)}). \tag{38}$$

We multiply this out and truncate to the terms appropriate for our quadratic approximation in ΔV_g. Finally, we get

$$\Delta w_{eq} \approx \frac{1}{2} \frac{1}{(\beta - \gamma)} E[(\Delta V_g / V_z)^2] \tag{39}$$

where $1/V_z^2 = 1/V_{g0}^2$ - $1/V_{g1}^2$.

Applying a sinusoidal input $\Delta V_g = V_2 \sin(\omega t)$, and taking the time average, we get

$$\Delta w_{eq} \approx \frac{1}{4} \frac{1}{(\beta - \gamma)} (V_2 / V_z)^2. \tag{40}$$

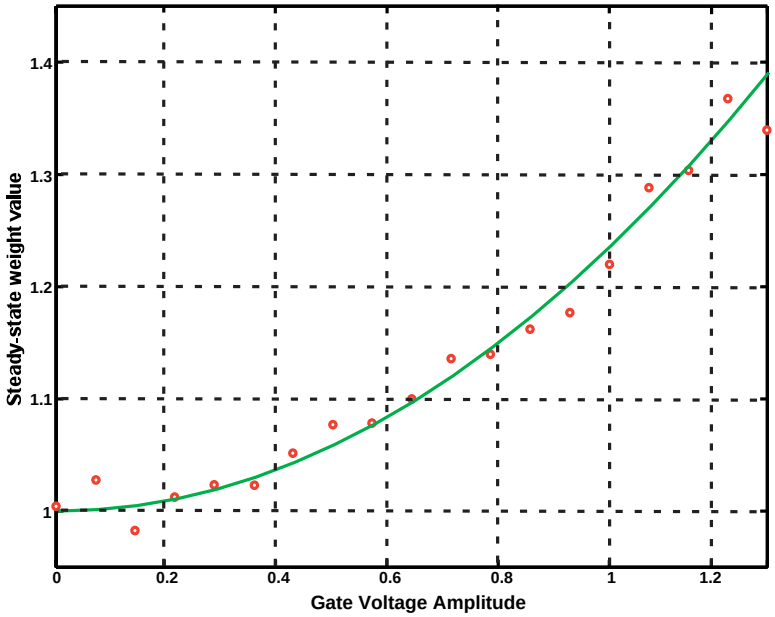

Figure 24: Change in the dynamical weight equation due to sinusoidal signal amplitudes at only the gate terminal. The drain voltage is remaining at a fixed voltage through this experiment. Equilibrium weight (w_{eq}) versus gate input amplitude.

Figure 24 shows w_{eq} vs. V_2 It is seen that the dependence of w_{eq} on V_2 is quadratic. We assume that $E[(V_g/V_z)^2]$ is constant and can be subtracted out of the weight dynamics equation. This assumption is realistic if we apply automatic gain control to the input voltage.

3.4 Equilibrium Weight is Determined by Correlations Between Gate and Drain Voltages

When we have fast timescale voltage signals on both the gate and the drain, we obtain the full equilibrium weight equation through a derivation similar to that used in the preceding sections:

$$\Delta w_{eq} \approx \frac{\frac{E[\Delta V_g^2]}{2V_z^2} - \frac{E[\Delta V_g \Delta V_d]}{V_{g0} V_{inj}} + \frac{E[\Delta V_d^2]}{2V_{inj}^2}}{\beta - \gamma}. \tag{41}$$

As we have previously stated, we can neglect the effects of the drain voltage. We can also assume that the signal energy in the gate voltage is constant. This allows us to subtract off the term due to gate voltage alone, counting it as part of the DC operating point of the

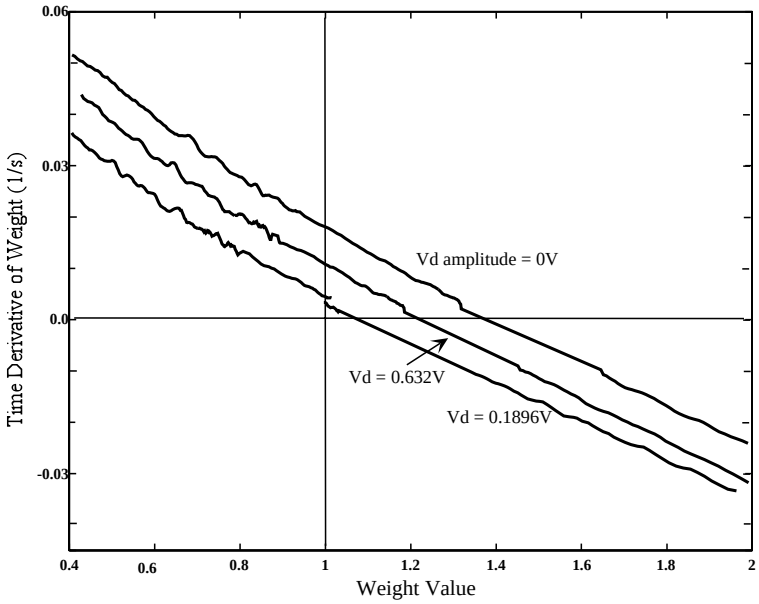

Figure 25: dw/dt vs. w for varying ΔV_d and fixed ΔV_g. This data gives strong evidence for our learning rule, $\tau' \frac{dw}{dt} = -w + \eta E[xy]$.

circuit. Taking these two points into consideration, we obtain the result

$$w_{eq} \approx -\frac{1}{\beta - \gamma} \frac{E[\Delta V_g \Delta V_d]}{V_{g0} V_{inj}}. \tag{42}$$

This result shows that we can adjust the synapse weight according to correlations between the gate and drain terminal voltage signals.

We can see this correlation effect by applying fast timescale gate and drain voltages in two different experiments. The experimental procedure follows that in the earlier cases; we apply a slow timescale step to the gate and fast timescale sinusoidal voltages to the gate and drain terminals. Assume the input for $\Delta V_d = V_1 \sin(\omega t)$ and $\Delta V_g = V_2 \sin(\omega t + \theta)$. Substituting these two inputs into (42) and computing the expected value, we get

$$w_{eq} \approx -\frac{1}{2} \frac{1}{\beta - \gamma} \frac{V_1 V_2 \cos(\theta)}{V_{g0} V_{inj}}. \tag{43}$$

For the first experiment, consider the two sinusoidal inputs with no phase difference ($\theta = 0$). Figure 26a shows w_{eq} vs. ΔV_d for various values of ΔV_g. We see that for fixed values of gate voltage, that there is a negative linear dependence of w on V_d as equation (42) suggests. In the second experiment, we sweep θ from 0 to 2π. For $\theta \neq 0$ we have Figure 26b shows w_{eq} vs. θ. We see definite correlations due to phase differences where $w_{eq} \propto -\cos\theta$.

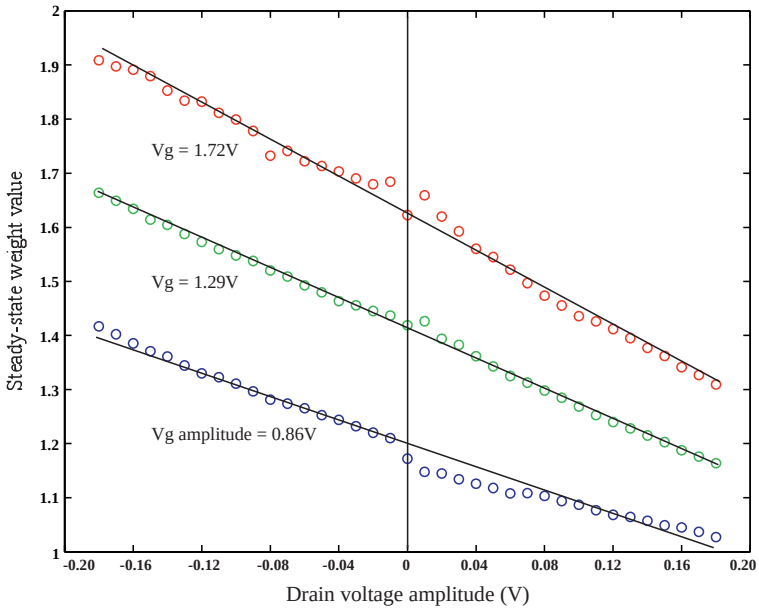

Figure 26: Change in the dynamical weight equation due to correlated sinusoidal signal amplitudes between the gate and drain terminals. Change in the dynamical weight equation due to different size drain amplitudes for three different gate amplitudes.

3.5 Hebbian Learning Rule from Approximated Weight Dynamics

We now present the approximated weight update rule for the linearized region using the results of the previous sections. The final result is

$$\begin{aligned} \tau' \frac{dw}{dt} &\approx -w - \frac{1}{\beta-\gamma}\frac{1}{V_{g0}V_{inj}}E[\Delta V_g \Delta V_d] \\ &\quad + \frac{1}{2}E[(\Delta V_g/V_z)^2] + \frac{1}{2}E[(\Delta V_d/V_{inj})^2] \end{aligned} \tag{44}$$

We derive the effects of the terminal voltages on the rate of the weight dynamics using methods similar to the equilibrium weight derivations. A new time constant, τ', is defined by the following

$$\begin{aligned} \frac{\tau}{\tau'} &\approx \beta - \gamma + \gamma\frac{E[\Delta V_g \Delta V_d]}{V_{g0}V_{inj}} \\ &\quad - \frac{1}{2}(\gamma/V_{g0}^2 - \beta/V_{g1}^2)E[(\Delta V_g)^2] - \frac{\gamma}{2}E[(\frac{\Delta V_d}{V_{inj}})^2] \end{aligned} \tag{45}$$

Neglecting the drain voltage effects as before, and assuming the effects due to the gate voltage are constant and can be absorbed into other circuit constants, we obtain the final approximation to (30) as

$$\tau' \frac{dw}{dt} \approx -w - \frac{1}{\beta-\gamma}\frac{1}{V_{g0}V_{inj}}E[\Delta V_g \Delta V_d] \tag{46}$$

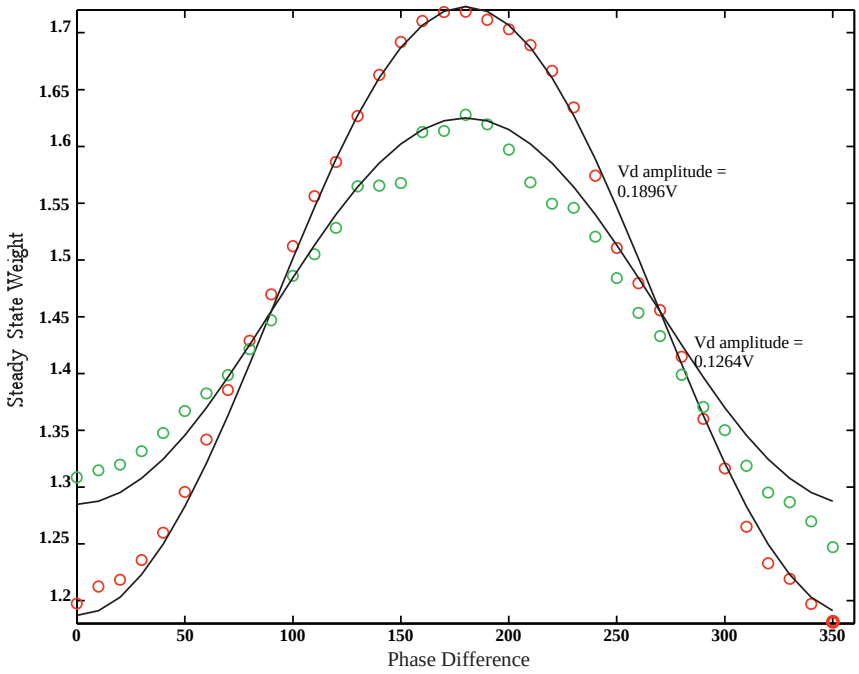

Figure 27: Change in the dynamical weight equation due to correlated sinusoidal signal amplitudes between the gate and drain terminals. Change in the dynamical weight equation due to various degrees of correlation (phase difference) between sinusoidal input signals at the gate and drain terminals.

where τ' is approximated as

$$\frac{\tau}{\tau'} \approx \beta - \gamma + \gamma \frac{E\{\Delta V_g \Delta V_d\}}{V_{g0} V_{inj}} \tag{47}$$

Figure 25 shows $\frac{dw}{dt}$ vs. w for varying ΔV_d and fixed ΔV_g. If we let $\eta = \frac{1}{(\beta-\gamma)V_{g0}Vinj}$, $x = \Delta V_{fg}$, and $y = -\Delta V_d$, then we can rewrite (46) as

$$\tau' \frac{dw}{dt} = -w + \eta E[xy]. \tag{48}$$

This is a Hebbian learning rule, based on the correlations between the signals x and y.

3.6 Concluding Comments

The first practical adaptive floating-gate circuit, the AFGA, exhibited signal-dependent adaptation leading to the development of the source-degenerated floating-gate pFET synapse. We initially noticed signal-dependent adaptation due to effects of drain variance. Next we

observed similar effects in these circuits due to gate variance. Further exploration of signal-dependent behavior in floating-gate pFET synapses culminated in discovery of the correlation learning rule. We would like to use this learning rule as the basis for several neural network and adaptive system algorithms such as principal components analysis, adaptive filtering, self-organizing neural maps, and supervised learning neural networks [36]. The remaining chapters address practical issues involved in taking floating-gate devices from single-synapse circuits to adaptive-node networks.

CHAPTER IV

FROM FLOATING-GATE SYNAPSES TO FLOATING-GATE NODES

As illustrated in the preceding chapters, floating-gate devices provide more than just digital memory functionality, they can also prove useful as computational circuit elements with analog memory and important time-domain computation and dynamic features all in one device. We envision large networks based on a few EEPROM type elements with simultaneous nonvolatile weight storage, matrix-vector multiplication, and continuous weight adaptation.

While the synapse is the basic element in any learning system, nodes are the next computational element to consider in adaptive networks. While a synapse is usually a single adaptive weight element, a node combines many synapses by summation. In some cases, the node output is simply the weighted sum of inputs. In other cases, the node output is a nonlinear function of the weighted sum. Combining several single transistor learning synapses results in a single node element for an adaptive circuit. An adaptive floating-gate node circuit provides a foundation to investigate many adaptive networks and learning algorithms.

In the first section of this chapter, we present experimental results on an adaptive floating-gate learning node built from an array of pFET single transistor synapses. This work is the first step in building dense on-chip learning networks implementing a wide space of learning algorithms. The second section addresses the effects of non-ideal behavior in the synapse and its learning rule and provides elegant circuit solutions to these problems. Additional non-ideal behaviors and ways of understanding and mitigating their effects in larger arrays of floating-gate synapses appear in the following two chapters.

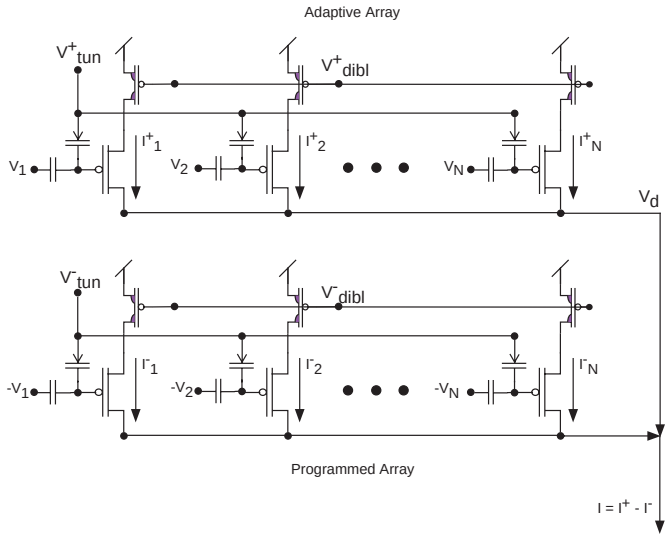

Figure 28: The differential synapse array enables four quadrant multiplication of weights with inputs. Each floating-gate transistor behaves as a transistor amplifier with an adaptive gain. Adaption occurs through slow-timescale dynamics due to the tunneling and injection process at the floating-gates. We continuously adapt the synapses with the positive signals, and we program the synapses with negative signals to balance the steady-state current of the positive synapse. The network output is the total drain current, and the *learning* signal is the applied drain voltage. By applying the appropriate relationship between output drain current and the resulting drain voltage, we could get a variety of learning rules. Since pFET hot-electron injection is unstable when channel current is the free parameter, we stabilize each synapse element by incorporating DIBL transistors to provide source-degenerative feedback.

4.1 A Simple Two-Input Floating-Gate Node

Figure 28 shows our adaptive floating-gate node, which is built from an array of differential four-quadrant synapses. The floating-gate transistor is a compact learning element which acts like an adaptive gain amplifier with built-in storage and gain adaptation. We define the adaptive gain as the weight. Gate voltages are input signals, and the drain voltage serves as a common learning signal for the node.

Because a single floating-gate device only allows positive weight values, we use a differential pair of synapses for each input signal, to obtain signed weights [33]. In this circuit, we continuously adapt one set of weights, which we will call the positive weights, while the other set of weights, which we will call the negative weights, does not adapt. Instead, the negative weights are programmed to give zero-point references for the full four-quadrant

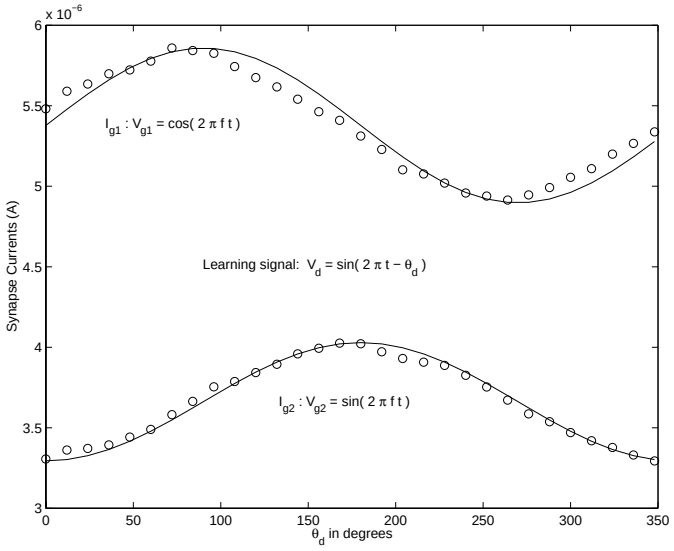

Figure 29: Experimental measurements of positive synapse current phase correlations. We would program the negative synapses to currents of 5.5μA (w_1^-) and 3.5μA (w_2^-), and therefore both weights are either positive or negative. These results show correlations between a specific gate and global drain terminal direct the convergence of that synapse's weight.

weight values. The output current of a differential synapse is described by

$$I = I^+ - I^- = I_b \sum_i (w_i^+ + w_i^-) + g'_m \sum_i (w_i^+ - w_i^-) \Delta V_i \tag{49}$$

where I_b is the bias current for every synapse in the array, $g'_m = \kappa' I_b / U_T$ is the transconductance of the floating-gate transistors, V_i is the input into the ith synapse, and w_i^+ and w_i^- are the plus and minus weights of the ith synapse, respectively. As a result, we get a weighted sum of inputs riding on a slowly moving bias current.

As seen in the previous chapter, the synapse weight adapts due to tunneling and injection currents at the floating node of the transistor, and linearizing the resulting weight-dynamics around a stable equilibrium, we obtained the weight update equation [29, 13]

$$\tau' \dot{w_i} = -w_i + a\langle \Delta V_i^2 \rangle + b\langle \Delta V_d^2 \rangle + c\langle \Delta V_i \Delta V_d \rangle, \tag{50}$$

where V_d is the drain voltage, and we define $w_i = w_i^+ - w_i^-$. The constants a, b, and c are voltage-normalizing terms dependent on tunneling and injection parameters and are discussed comprehensively in the previous two chapters as well as earlier publications [31,

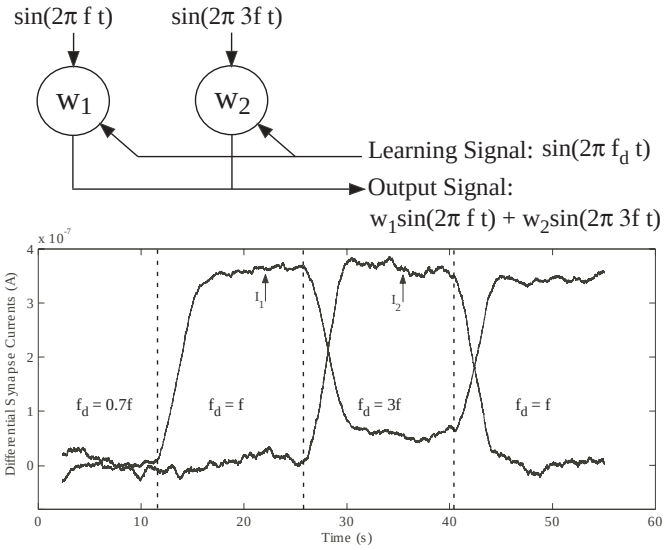

Figure 30: Experimental measurements of frequency correlations for sinusoidal inputs to the two-input node given by $V_{g1} = sin(2\pi 3ft)$ and $V_{g2} = sin(2\pi ft)$. The learning signal, $V_d = sin(2\pi f_d t)$, takes on three different frequencies $f_d = 0.7f, f, 3f$. Current outputs from our differential synapses. We programmed our negative weights to offset the positive synapse current measured with input signals with the same signal variance. We see that the synapse that has an identical synapse input frequency as the drain signal has a non-zero weight.

30]. Solving for the steady-state, or equilibrium weight, yields

$$w_i = -\langle V_i V_d \rangle. \tag{51}$$

When we have programmed the appropriate zero-point reference weights and constrain each input at some constant RMS value, then the gate variance and the drain variance terms cancel in this differential structure.

We want to compare this with the solution to the least mean square algorithm [8], $\overrightarrow{w} = Q^{-1}(\overrightarrow{V}d)$, where Q is the autocorrelation matrix of the input signal, $\overrightarrow{V}$ is a vector of inputs, and d is the desired learning signal. For orthogonal inputs (Q is diagonal), the weight equations are solved as

$$w_i = \frac{< (V_i)(d) >}{< V_i^2 >} \tag{52}$$

which compares well to (51) when the input variance remains constant. Thus, we can use this floating-gate node in adaptive signal processing and neural network applications.

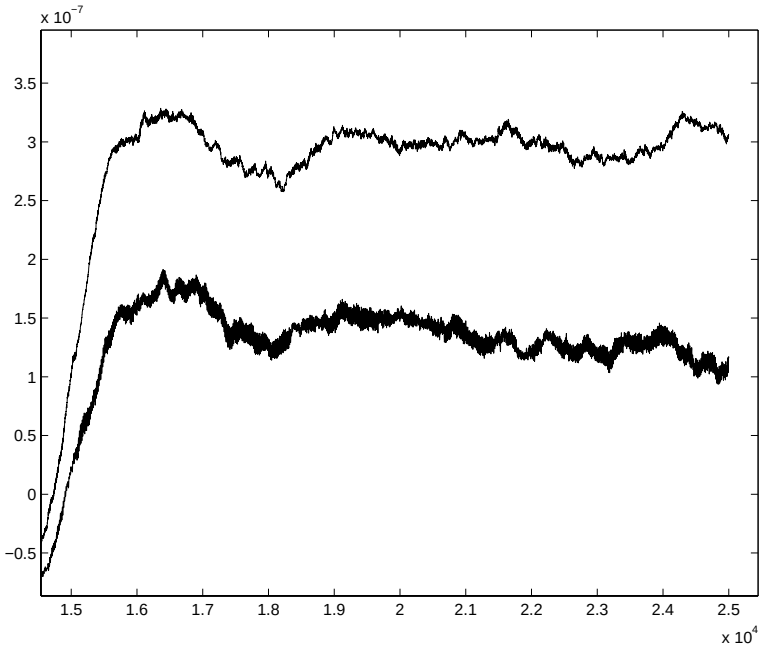

Figure 31: Experimental measurements of a square wave learning signal applied to V_d Time course of steady-state currents showing convergence of weights.

4.1.1 Simple Floating-Gate Learning Node Experiments

In the first experiment, we show adaptation to phase correlations of fixed-phase sinusoids applied to each synapse input with a sinusoid of variable phase applied as the learning signal. Figure 29 shows the results of our first experiment. The inputs to the node are two sinusoidal signals, with the same frequency (f): $V_{g1} = 0.15cos(2\pi ft)$ and $V_{g2} = 0.15sin(2\pi ft)$, where the drain *learning* signal is $V_d = 0.3sin(2\pi ft - \theta_d)$ From (51) we expect steady-state synapse weights as

$$w_1 = sin(2\pi\theta_d) \text{ and } w_2 = -cos(2\pi\theta_d), \tag{53}$$

The experimental data in Fig. 29 shows the resulting positive weight values when we swept the phase. Experimental data shows close agreement with analytic results.

As (51) predicts and Fig. 29 verifies, we get an anti-correlation instead of a correlation learning rule, where in the case of V_{g2} and V_d, the minimum value occurs when θ_d is 0° and the maximum value occurs when θ_d is 180°. Simply negating the learning signal applied to the drain terminal will yield the desired correlation rule.

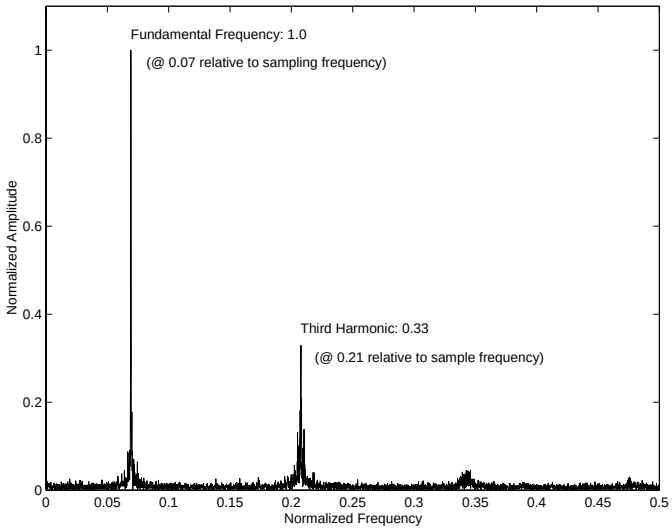

Figure 32: Experimental measurements of a square wave learning signal applied to V_d Spectrum of output current shows amount of each node input frequency matched to frequency components in learning signal. The frequency axis is normalized; the frequency does not affect these results until it approaches the adaptation rate. We obtain 1 and 1/3 for the fundamental and third harmonics as expected. The fifth harmonic appears due to the drain voltage coupling into the floating gate through overlap capacitance.

In the second experiment, we show adaptation to frequency correlations when we apply a different frequency to each of the synapse inputs, and we apply another sinusoid as a learning signal. Figure 30 shows experimental time-course measurements from our adaptive circuit where the inputs are sinusoids at a fundamental and related third harmonic frequency, and the drain voltage is also another sinusoid. We show the signals from the positive synapses and again from the differential synapses. We programmed the negative synapses to eliminate the steady-state current of the positive synapses for similar input sizes. This approach also compensates for the mismatch between synapse equilibrium points. We observe that the circuit identifies a correlation between its input and the drain *learning* signal.

4.1.2 Learning a Square Wave from Sinusoidal Inputs

We present another example where we train the network to learn the appropriate Fourier coefficients for the components of a square wave. Our experimental results agree with theoretical expectations. From the definition of Fourier series a periodic signal can be

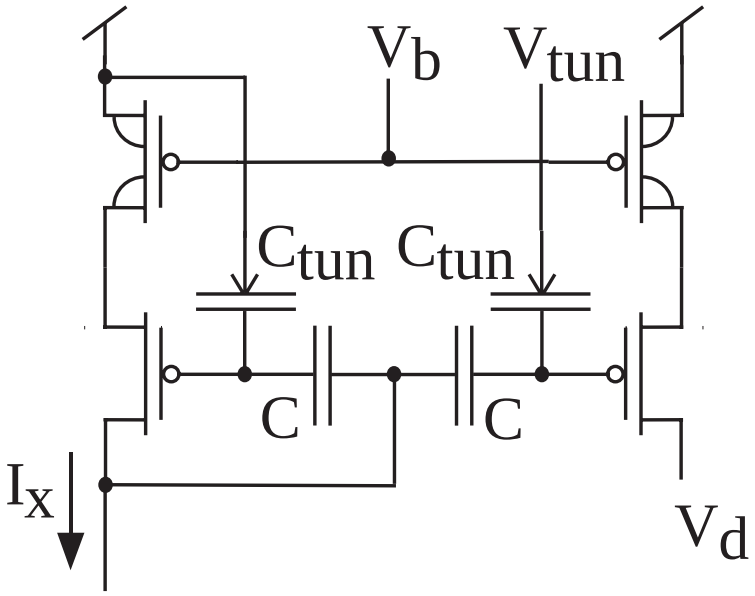

Figure 33: The pre-distort circuit is a simple current mirror which implements $\Delta V_g = V_{mg} \ln(1 + x)$ to eliminate harmonic distortion in the output current.

expressed as:

$$s(t) = \sum_n a_n cos(2\pi nft) + \sum_n b_n sin(2\pi nft), \tag{54}$$

where the coefficents result from the correlation of harmonically related sinusoids with the input signal given as:

$$a_n = \langle s(t), cos(2\pi nft) \rangle, \; b_n = \langle s(t), cos(2\pi nft) \rangle. \tag{55}$$

As a result, we expect our adaptive circuit to converge to these coefficients.

Figure 31 shows experimental time-course measurements from our adaptive circuit where the inputs are sinusoids at a fundamental and related third harmonic frequency, and the drain voltage learning signal is a square wave. We show the convergence of the signals in Fig. 31 for the positive synapses. Figure 32 shows the normalized amplitude and frequency of the Fast Fourier Transform of the resulting signal to make it easy to compare relative amplitudes at relative frequencies. From this experimental data, we get the expected square wave Fourier coefficients for the fundamental and third harmonics. This experiment demonstrates this circuit's behavior in extracting Fourier coefficients. The learning success demonstrated by this simple Fourier experiment show that the single transistor learning

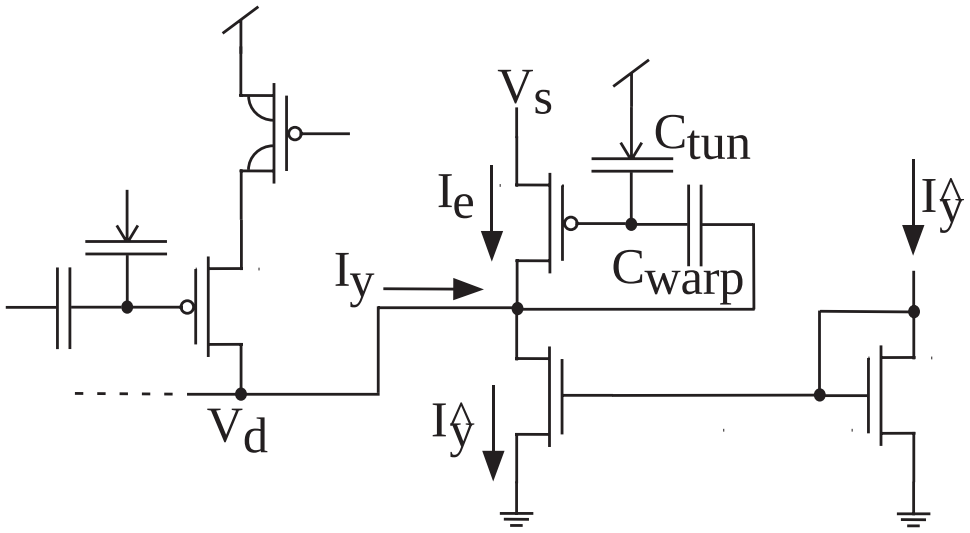

Figure 34: The pre-distort circuit for the drain voltage implements $\Delta V_d = V_{md} \ln(1 + y)$ to eliminate drain variance terms in the weight value.

synapse can be utilitzed to implement an adaptive node with many inputs. However, we had to assume constant RMS values for the input signals and account for device mismatch, gate variance, and drain variance effects in the learning rule to obtain this data. Therefore recognition of these constraints and non-ideal behaviors led to designs for compensation circuitry as will be discussed next.

4.2 Removal of Non-Ideal Effects in the Correlation Learning Rule

While we would like to obtain a learning rule purely dependent on correlations between the gate and drain voltages, the rule we actually obtain from (30) leads to weight dependencies on the variance of these two voltages as well. Also, we may not be able to make the gate voltage small enough to obtain a linear model of (26) while simultaneously keeping it large enough to see correlation behavior. The resulting harmonic distortion can mask the weight effects. In that case, we would like to eliminate harmonic distortion. Because the drain current of the source-degenerated floating-gate pFET and its equilibrium weight value are both determined by exponential functions of the terminal voltages, we explore the use of logarithmic pre-distortion to remove these undesired effects. Figure 33 shows the circuit

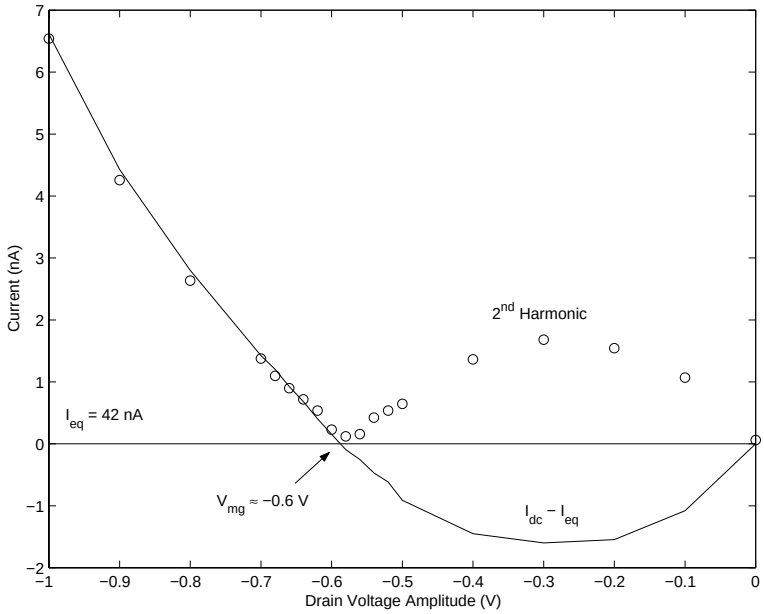

Figure 35: This plot shows the DC value (subtracting the equilibrium current) and 2^{nd} harmonic of the drain current vs. gate voltage pre-distort value. The gate pre-distortion is given by $\Delta V_g = V_{mg} \ln(1 + A \sin \omega t)$ for $\Delta V_d = 0$. We find that the appropriate pre-distort factor ($V_{mg} \approx -0.6V$)to be that which makes the 2^{nd} harmonic to be zero and coincides with the non-zero root of $I_{dc} - I_{eq}$. I_{eq} is the value the drain current assumes when there is no signal input.

structure which will provide gate pre-distortion, and Fig. 34 illustrates the circuit which will be used to provide pre-distortion on the drain voltage.

4.2.1 Harmonic Distortion, Gate Variance, and Gate Pre-Distortion

We first remove the harmonic distortion from the pFET drain current by pre-distorting the gate voltage input. The pre-distorted voltage applied to the gate is

$$\Delta V_g = V_{mg} \ln(1 + A \sin \omega t). \tag{56}$$

In this experiment, the pre-distort factor, V_{mg}, ranges from -1 V to 0 V and the sinusoidal amplitude, A, is 0.65. This experiment is performed with the adaptation mechanism turned off such that the only effect observed is the harmonic distortion. Substituting (56) in (26) and simplifying yields

$$I = I_b (1 + A \sin \omega t)^{\alpha_g}, \tag{57}$$

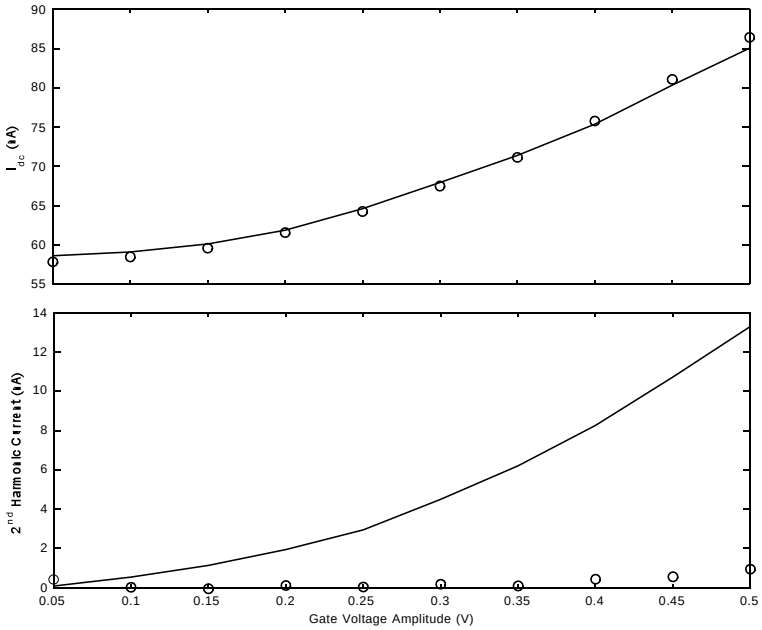

Figure 36: These plots compare the dc values and second-harmonic values of the drain current vs. gate voltage amplitude both with and without pre-distortion. We see that the dc value follows a similar quadratic form in both cases, implying that there is still significant gate variance with pre-distortion for a given pre-distort value. The second-harmonic plot shows that harmonic distortion has been significantly reduced.

where $\alpha_g = -V_{mg}/V_{gA}$. A second-order approximation of (57)leads to

$$\begin{aligned} I \approx I_b(1 + \tfrac{1}{4}\alpha_g(\alpha_g - 1)A^2 + \alpha_g A \sin\omega t \\ -\tfrac{1}{4}\alpha_g(\alpha_g - 1)A^2 \sin 2\omega t). \end{aligned} \tag{58}$$

This second-order approximation indicates that we should look at the dc value and second harmonic of the drain current in Fig. 35 to determine the appropriate pre-distort factor that will eliminate harmonic distortion. From (58), we see that we should look for the value of V_{mg} where the second harmonic becomes zero, which gives us the appropriate pre-warp factor, V_{mg}. We also see that this coincides with the non-zero root of the dc current when the equilibrium value is subtracted. Figure 35 demonstrates that the appropriate pre-distort value for this gate is $V_{mg} \approx -0.6V$, which is the corresponding V_{gA}.

Both the dc values and the second-harmonic values of the drain current with and without pre-distortion are compared in Fig. 36. Both sets of data are obtained with the continuous-time adaptation mechanism turned on. We obtain the non-distorted data by applying a

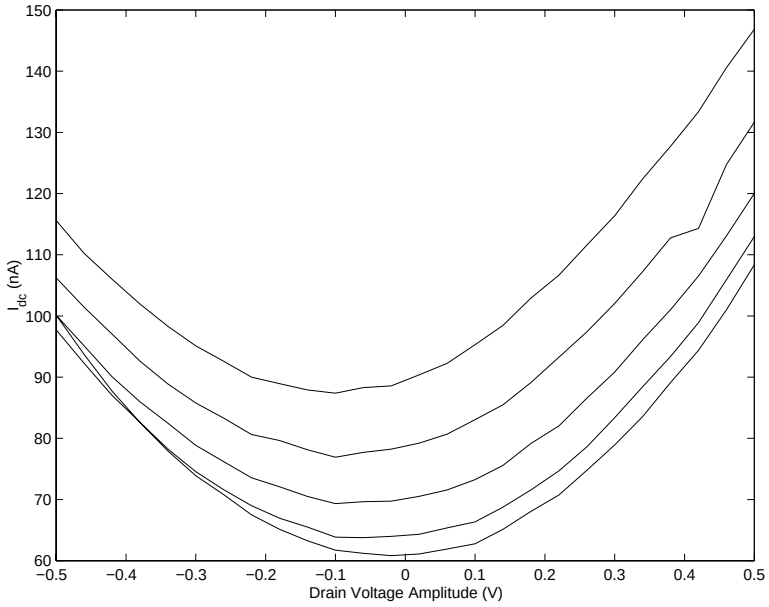

Figure 37: A plot of DC value of the drain current (weight) vs. drain voltage amplitude for a pre-distorted sinusoid applied to the gate and a non-distorted sinusoid applied to the drain. The gate variance contributes to constant offsets in the data. The quadratic drain variance term masks the linear correlation term.

sinusoidal input, $\Delta V_g = A_g \sin \omega t$, of various amplitudes ($0.05V < A_g < 0.5V$ in steps of 0.05 V) with no drain voltage change ($\Delta V_d = 0$). For the pre-distorted data, we use the factor, V_{mg}, found above. We choose the sinusoidal amplitude values, A, in 56 as

$$A = -\tanh(A_g/V_{mg}), \tag{59}$$

where A_g is the set of amplitudes for the non-distorted case and V_{mg} is the pre-distort factor. This equation gives us comparable voltage amplitudes in both the non-distorted and pre-distorted cases. In Fig. 36 we see that dc current still displays a quadratic dependence, thus gate variance still has a significant effect even with pre-distortion. This happens because the pre-distort value for harmonic distortion cancellation and that for optimal gate-variance cancellation are not the same. This can be seen by comparing the exponential voltage scale factors in (26) and (30). Figure 36 compares the second-harmonic values of the drain current from the non-distorted and pre-distorted cases and shows that the harmonic distortion has been nearly eliminated in the pre-distorted case.

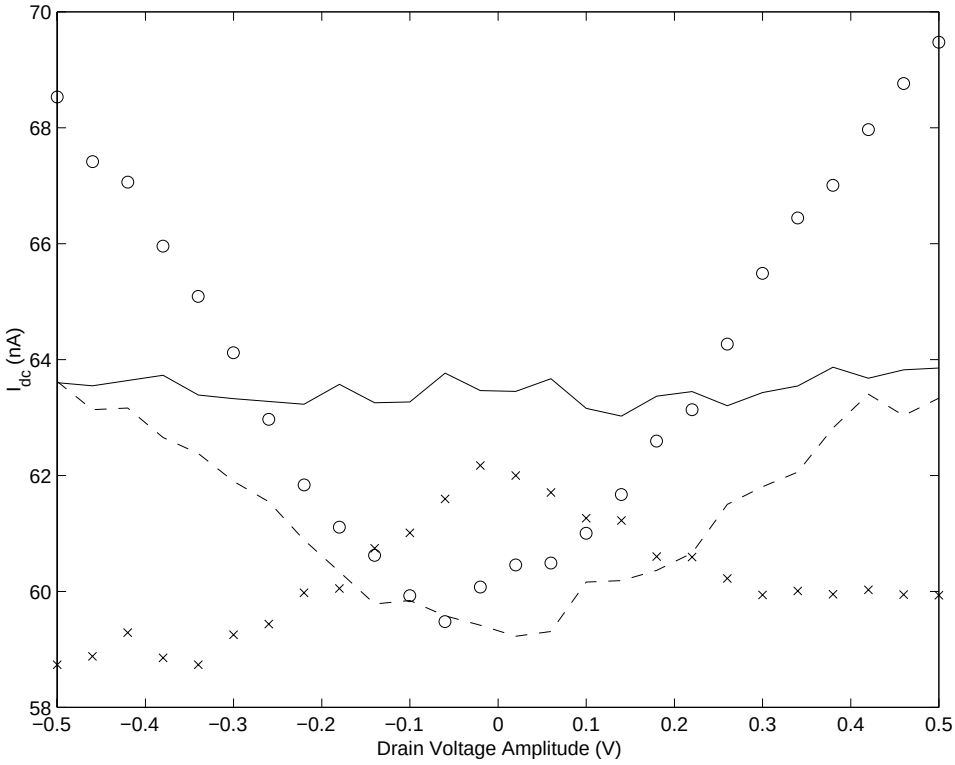

Figure 38: This plot shows the DC value of the drain current vs. drain voltage amplitude when $\Delta V_d = V_{md} \ln(1 + A \sin \omega t)$ with $\Delta V_g = 0$. Here we have plots for several values of V_{md}. We choose that value of $V_{md} = -0.22V$ (corresponding to the flattest curve) as the appropriate distortion factor to eliminate drain variance effects from the weight value.

4.2.2 Drain Variance Effects, Drain Pre-Distortion, and Signal Correlations

Now that we have eliminated the effects of harmonic distortion, we observe the weight that results for non-zero gate and drain voltages. Figure 37 illustrates how the effects of gate and drain variance can mask the desired correlation dependence of the weight. We wish to examine how pre-distortion of the drain voltage affects the resulting weight value.

An analysis of the drain pre-distort circuit illustrated in Fig. 33 leads to the following

$$V_d = V_s - V_{md} \ln[(I_{\hat{y}} - I_y)/I_b], \tag{60}$$

where $V_{md} = \frac{C_T}{C_{warp}} \frac{U_T}{\kappa}$. In this paper, we set the target-signal current in (60) to a constant bias value ($I_{\hat{y}} = I_b$), and we define the learning signal to be $y = I_y/I_b$. This leads to the drain voltage function

$$V_d = V_s - V_{md} \ln(1 - y) \tag{61}$$

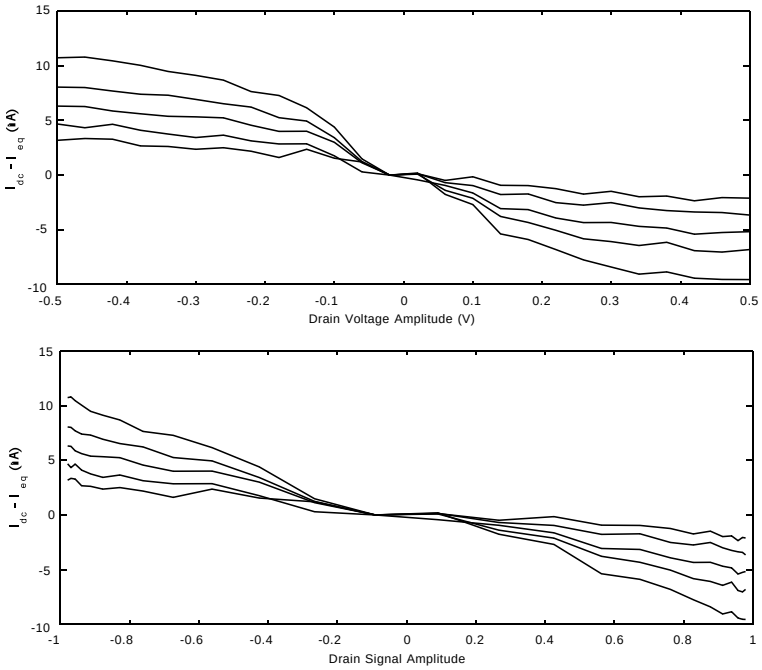

Figure 39: The top figure is a plot of the dc current (weight) minus the equilibrium current value and those current offsets due to gate variance vs. the amplitude of the drain voltage. We see that the quadratic effect due to drain variance has been eliminated. The curves presented have a *tanh* form due to the relationship between the signal amplitude and drain voltage amplitude because of pre-distortion. The bottom figure shows the same data plotted vs. signal amplitude revealing linear correlations. Steeper slopes correspond to larger gate voltages. Thus, we see that $w \propto -E[xy]$, where x and y are sinusoidal signals to be pre-distorted, is verified.

which is used in our learning rule. All of the experiments described here have the drain signal of $y = A\sin(\omega\ t + \phi)$.

We obtain the appropriate pre-distort factor for the drain by observing the dc current value vs. drain voltage amplitude for several pre-distort factors. Figure 38 exhibits the results of this experiment. The appropriate pre-distort factor gives the flattest curve. As seen from the figure, this value is $V_{md} \approx -0.22V$.

With this pre-distort value for the drain voltage, we observe the resulting weight values when we apply pre-distorted sinusoidal signals to both the gate and drain voltages. We hope to see the correlation effects more clearly. In Fig. 39a, we have plotted the dc current value vs. the drain voltage amplitude. We see a set of curves described by a tanh function, which is due to the relation given in (59) as applied to the drain. Plotting these dc current

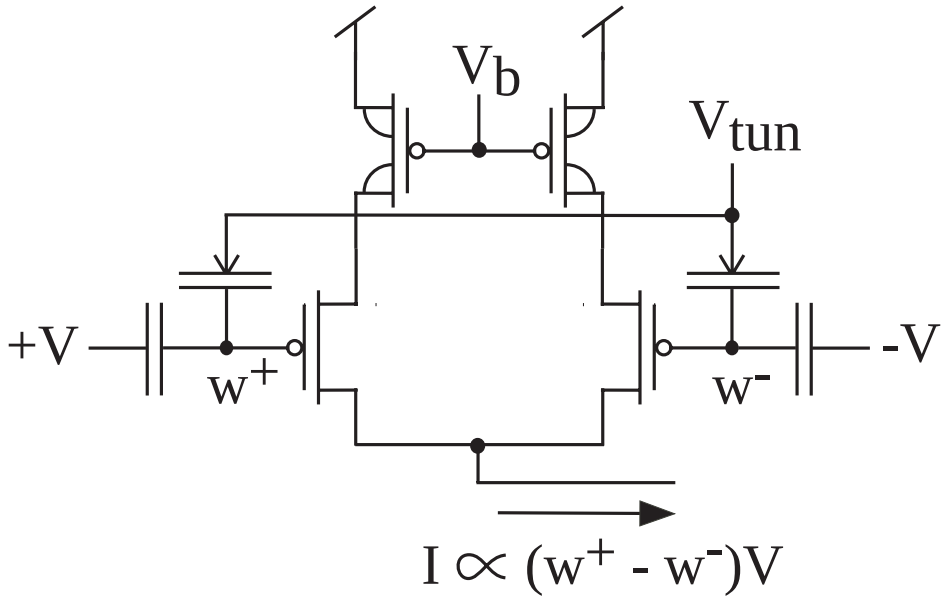

Figure 40: In the ideal situation of no mismatch, an adapting differential pair of floating-gate devices would cancel both the gate-variance and constant terms out of the correlation learning rule.

values vs. the actual drain signal amplitude, A, in Fig. 39b, we see a set of straight lines of various slopes. Steeper slopes correspond to larger gate voltage amplitudes. These lines correspond to $w \propto -E[xy]$ where x is the pre-distorted gate signal and y is the pre-distorted drain signal. This plot demonstrates that we indeed obtain clear linear correlation behavior with pre-distortion. In both plots we have subtracted offsets due to the gate variance.

4.2.3 Cancelling Constant Offset and Gate Variance in the Weight

Adding a couple of simple, elegant circuits to our basic synapse array eliminated two nonlinear effects in our adaptive floating-gate devices which tend to mask the correlation term in our weight. These circuits provide pre-distortion on the gate inputs to eliminate harmonic distortion and pre-distortion on the drain voltage to eliminate drain-variance terms from the weight equation.

To obtain a pure correlation learning rule, we must yet remove the constant offset and gate-variance terms appearing in the equilibrium weight. Theoretically, the differential structure shown in Fig. 40 should cancel all even-order terms due to the input signal. Unfortunately, device mismatch as illustrated in Fig. 41 results in significant differences in

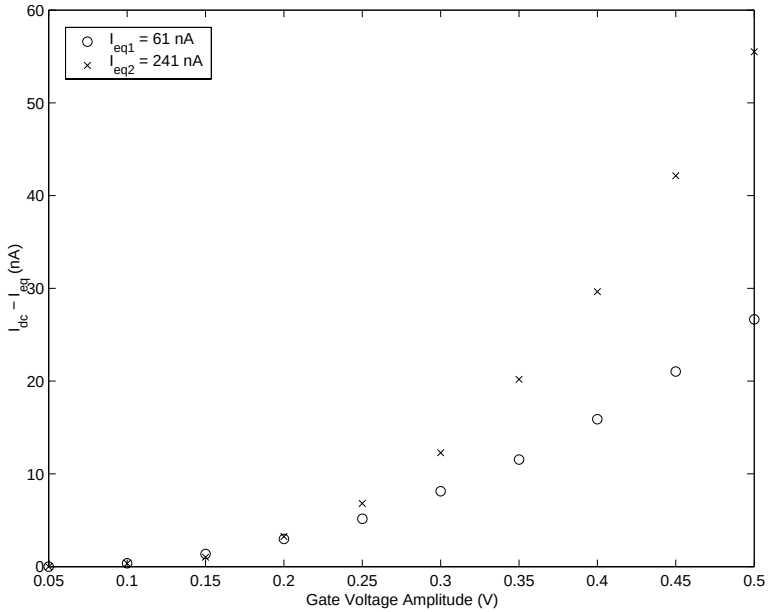

Figure 41: Plot of dc current values (weights) minus equilibrium values for two different source-degenerated floating-gate pFETs vs. gate voltage amplitude. Tunneling junction mismatch leads to significant differences in the gate-variance terms in the correlation learning rule. The legend demonstrates the large variation of equilibrium current values also due to mismatch.

gate variance, prohibiting this solution for even terms greater than order zero. However, the order zero, or DC, term can be easily cancelled in the floating-gate differential pair by programming the charge on one side of the pair to yield the same current as the equilibrium condition in the adapting floating-gate pFET, thus avoiding mismatch issues in DC bias current. Differences in weight offsets (due to differences in equilibrium current values) can be compensated using a differential floating-gate pair where one side continuously-adapts and the other side of the pair has its floating-gate charge programmed to cancel the offset.

Minimizing gate-variance effects will lead to significant improvements in the correlation learning rule. To eliminate the gate-variance problem without eliminating mismatch, we could force every input to a fixed amplitude using an automatic gain control circuit. The gate-variance term would then be a known constant to be programmed out along with the weight offset. To remove gate variance by eliminating mismatch, we recognize that mismatch is due to the tunneling junction of the floating-gate device. In our current devices, most tunneling occurs at the edge of the tunneling junction; increasing the length of this edge

for each device would tend to improve matching. Alternatively, we could attempt to tunnel directly through the well of the floating-gate device.

4.3 Concluding Comments

In this chapter, we have seen how source-degenerated pFET floating-gate synapses can be combined to form a single node by simply connecting their drain terminals together. This connection provides both a summation of the weighted output currents through Kirckhoff's Current Law, as well as allows every synapse in the node to share the same error signal. Preliminary results showed the ability of the system to converge to Fourier series coefficients for uncorrelated, harmonically related sinusoids. However, special attention had to be paid to avoid non-ideal behaviors in the multiplication computation and in the learning rule. Simple, yet elegant circuits were designed to compensate for these undesirable effects. Unfortunately, gate-signal variance remained a problem to overcome. In addition, another behavior have to compensate for was not observed due to the uncorrelated nature of the input signals — this was the effect of weight decay in the learning rule. In general, the input signals to an adaptive node will be correlated, and we will have to deal with weight decay and its effects. In the following, we discuss a new synapse design which will reduce the effects of weight decay and which will provide the added bonus of eliminating gate-signal variance, obviating the need for special restrictions on the input signals or resort to well-tunneling.

CHAPTER V

GENERAL LEAST-MEAN-SQUARES LEARNING IN FLOATING-GATE NODES

Figure 42 shows our improved LMS floating-gate synapse circuit. This core synapse circuit consists of three transistors: a floating-gate pFET in the middle computes the weight-signal product and another floating-gate and regular pFET pair on the right compose a source-follower circuit, enabling fairly ideal correlation behavior and minimal weight decay. The circuit is similar to our previous synapse element [30], except here we *linearize* and stabilize the synapse with a regular pFET current source rather than a dibl pFET. Unlike the dibl pFET, the regular pFET effectively increases the output resistance seen at the floating-node by the tunneling and injection mechanisms because this circuit behaves like a cascode configuration [2]. While the synapse presented here superficially resembles a previously proposed synapse [46], it operates in a fundamentally different way (continuous-time correlation) using analog instead of digital signals, and was inspired by different design considerations (minimal weight decay).

5.1 Issue of Weight Decay for Physical LMS computation structures

Least-mean-square (LMS) learning rules result from minimization of a least-square-error objective function. Some LMS algorithms intentionally incorporate weight decay — a form of *forgetfulness* exhibited by the learning system — for better learning generalization and tracking of non-stationary signals [8, 35, 36]. Analytical modeling of learning rules yields the following weight dynamics and steady-state solutions for an LMS algorithm with and

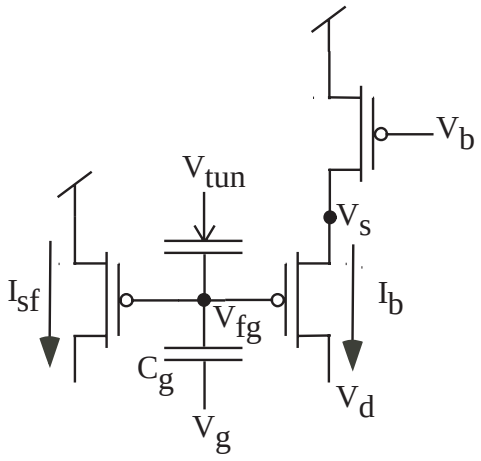

Figure 42: The Source-Follower Floating-Gate (SFFG) synapse.

without weight decay [35]:

$$\begin{array}{rll} & \text{No Weight Decay} & \text{Weight Decay} \\ \text{Dynamics:} & \tau\frac{d\mathbf{w}}{dt} = \mathbf{x}e & \tau\frac{d\mathbf{w}}{dt} = \mathbf{x}e - \epsilon\mathbf{w} \\ \text{Steady-state:} & \mathbf{w}_{ss} = \mathbf{Q}^{-1}\mathbf{r} & \mathbf{w}_{ss} = (\mathbf{Q} - \mathbf{I}\epsilon)^{-1}\mathbf{r} \end{array} \tag{62}$$

the error signal, $e = \hat{y} - y$, results from the difference between the output and a target signal, $\hat{y}$. The steady-state solution, $\mathbf{w}_{ss}$, depends on the input auto-correlation matrix, $\mathbf{Q} = E\left[\mathbf{x}\mathbf{x}^T\right]$ and the input-output cross-correlation vector $\mathbf{r} = E\left[\mathbf{x}y\right]$. The expected value operation, $E[\cdot]$, represents ensemble averaging, however we will assume our signals are ergodic and use time-averaging for this operation. The strength of the weight decay, ϵ, should be small ($\epsilon << 1$) relative to the input and learning-signal amplitudes to minimize deviation from the ideal LMS solution. However, a tradeoff exists between the stability of a physical synapse implementation and the amount of weight decay present in the LMS algorithm; the same mechanism responsible for weight decay results in high-gain and stability concerns in the circuit. Thus we require a synapse circuit that optimizes ϵ to produce sufficiently accurate LMS solutions while maintaining stability.

As a result, it would seem we need simply to implement (62) with $\epsilon = 0$; this solution requires an ideal, bidirectional current source to integrate current on a capacitor. In practice, we will get a weight decay term either through the leakage of a constant current source

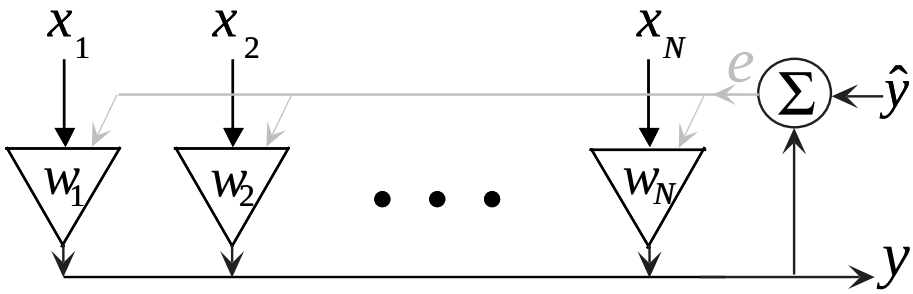

Figure 43: Block diagram of a single adaptive floating-gate node. Because the synapse elements far outnumber all of the other components, each synapse element must be as small as possible. Filters extract the output signal (y) from its slow-timescale and DC bias currents. Program circuitry on the periphery allows us to set the initial conditions and / or to restart the weights in a known state when experimenting with various algorithms.

(i.e. reversed-biased diodes) or an effective finite output resistance through our correlating element. For floating-gate circuit approaches, we have negligable leakage current (mV change over years); for non floating-gate circuit approaches, these issues put a practical lower bounds on the resulting adaptation rate (usually 1s to 10s range). The mathematical implications of both types of weight decay are similar if the effects are not large.

Further, approximating a bidirectional current source implies that we have a balance between one circuit element providing a negative current and another providing a positive current. For floating-gate circuit implementations, electron tunneling currents (I_{tun}) provide the positive current, and hot-electron injection currents (I_{inj}) provide the negative current. In equilibrium these two currents are equal, and for floating-gate implementations, we define this current as I_{fg0}.

The simple circuit of Fig. 44 exhibits the minimum requirements for a correlation-based learning rule. The nonlinear voltage-controlled current source provides the signal-error product for the correlation, while a capacitor provides the dynamic mechanism for the weight update; the resistor models weight decay. This classical correlator approach could utilize one of many voltage to current multiplying elements could be such as Gilbert multipliers [42], linearized resistance multipliers [10, 16, 28], or floating-gate based multipliers [1]. If we ignore Early effects of these transistors, then $\epsilon \rightarrow 0$. We see that τ is set similar

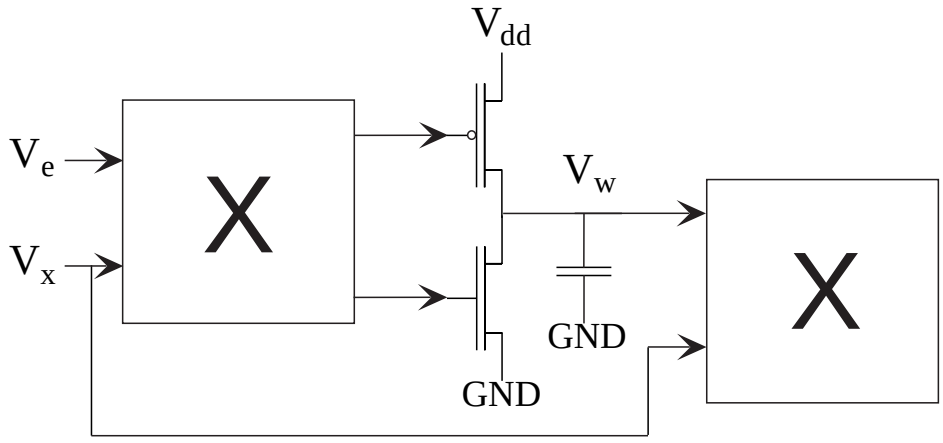

Figure 44: Basic block diagram for a weight update using transistors for the weight update element. The result is either non-negligible weight decay or a difficult analytical system to control. Therefore in practice, we would design our weight decay to be small enough for our algorithm but large enough to achieve the desired level of stability.

to the transconductance (g_m) for an amplifier generating the current to the integrating capacitor. and that τ/ϵ is set through the output resistance modulating current to the integrating capacitor. Considering the Early voltage effect of the nFET (V_{An}) and pFET (V_{Ap}) operating with subthreshold bias currents, we get that $\epsilon = U_T/(\kappa V_{An} \parallel V_{Ap})$, which results in fairly small weight decay values (< 0.01 even for minimum channel lengths). One could get even lower values by cascoding both the nFET and pFET transistor. Not all circuit elements, e.g. floating-gate devices at the gate terminal, will naturally have large enough gain or can be easily cascoded, and therefore additional circuit techniques will be necessary.

Low weight decay results from a high gain / impedance node, which also implies a system that becomes more difficult to control without feedback (which often increases ϵ), particularly since the resulting weight voltage will often have a small usable voltage range (like 100mV - 200mV). The one solution, even in digital solutions, is to initialize the weights before adapting and/or reseting the system when it drifts far off course. Similar weight decay issues occur in pulse-based systems, because ideal current sources are also needed to eliminate weight-decay components as well.

Our first version of a correlating floating-gate synapse, illustrated in Fig. 17 (chapter 2),

showed weight adaptation based upon correlations of the input and error signals [30], but these synapses exhibited sizable weight decay, where ϵ was on the order of 1. These synapses are closely related to our current synapse design, and these synapses are a good study in the difficulty of removing weight decay from floating-gate synapse design. The top transistor in the schematic of Fig. 17, M2, is a DIBL (Drain-Induced Barrier Lowering) transistor, which possesses a strong exponential Early voltage effect [30]. The added transistor provides stability and linearized the learning rule as well.

This circuit gives us an outer-product learning rule based on relative changes in the gate signal (x) and relative changes in the drain signal (y) as

$$\tau \frac{dw}{dt} = -\epsilon w + E[xy]. \tag{63}$$

where τ will depend upon device parameters and is inversely proportional to the equilibrium floating-gate current (I_{fg0}), and ϵ is calculated as [30]:

$$\epsilon = \frac{1}{\kappa_x} \frac{U_T}{\kappa V_x \parallel V_{inj}} - 1 \tag{64}$$

where ϵ was typically near 1 for most implementations.

One straightforward example is that we apply the target signal as a drain voltage, and therefore we mathematically and experimentally connect our adaptive floating-gate devices to a class of adaptive filters. The resulting steady-state solution: $w = \frac{1}{\epsilon} E[xy]$. Considering a typical LMS adaptation, we get the same solution if $\mathbf{Q}$ is an identity matrix, which requires the inputs to be uncorrelated and of the same input amplitude. Figure 30 shows experimental time-course measurements from this circuit, where the inputs are sinusoids at a fundamental and related third harmonic frequency, and the drain voltage is also another sinusoid corresponding to the learning signal. We observe that the circuit identifies a correlation between its input and the drain *learning* signal.

5.2 The LMS Floating-Gate Synapse Circuit

Implementing the system shown in Fig. 43 requires two basic functions: feedforward computation and weight adaptation. Feedforward computation consists of multiply and add

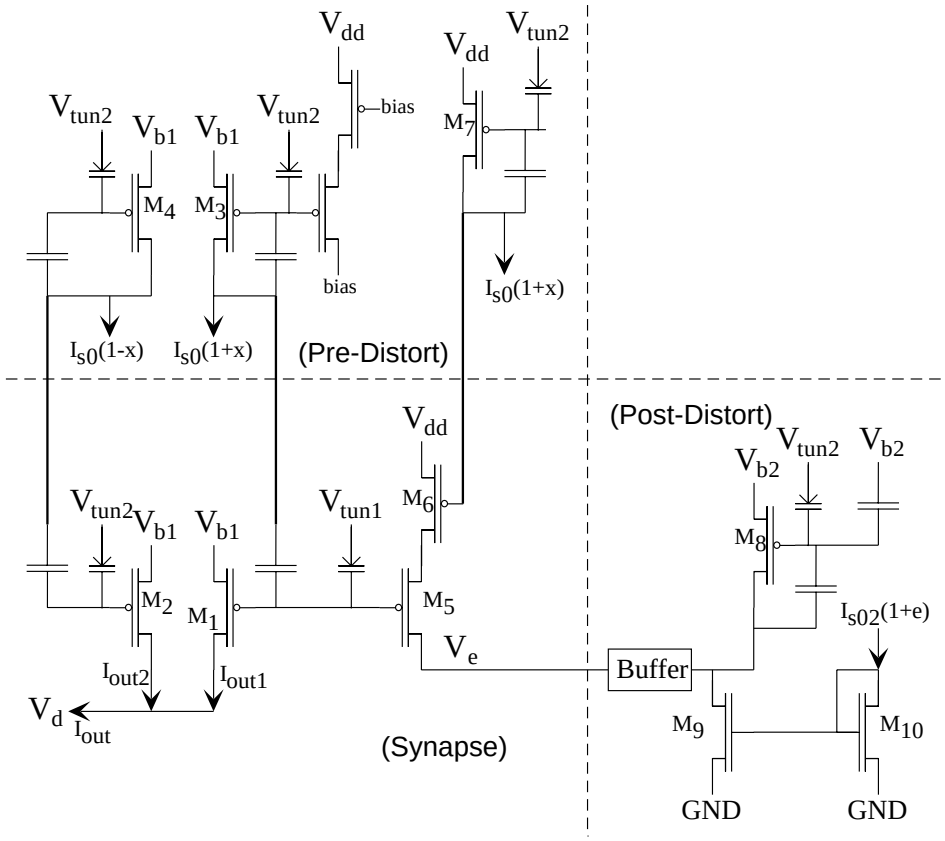

Figure 45: Building a node from a correlating synapse element. Full circuit with required peripheral interface circuitry. Interface circuitry linearizes synapse computation and adaptation. The figure also shows programming circuitry needed to set all of the floating-gate elements. V_{tun1} is the tunneling line for the adaptive and programmed floating-gate elements, where V_{tun2} is the tunneling line for the program only floating-gate elements.

operations. Weight adaptation requires circuitry to perform correlation operations on input and learning signals. In this section, we present circuit implementations and measured results illustrating this functionality.

5.2.1 Feedforward Synapse Computation

The floating node at the gate of each synapse transistor is completely surrounded by silicon dioxide, an excellent insulator, providing non-volatile weight memory when the adaptation mechanism is off. Operating pFET devices in subthreshold yields an exponential current-voltage relation, transforming the sum of the capacitively-coupled input voltage and the voltage due to charge stored on the floating-gate into a multiplication between the input signal and the weight. Figure 45 illustrates the circuitry used for our adaptive node which

includes the pre- and post-distortion circuitry discussed in chapter 4. We program negative synapses to eliminate the signal effect of the steady-state current of the positive synapses for similar input sizes (the resulting bias current increases). Also, since we use a differential input voltage, we get our output current by summation on one line, rather than taking the difference of two currents. This approach also compensates for the mismatch between synapse equilibrium points at the point where tunneling current equals injection current at each synapse. The goal is to develop a continuously-adapting multilayer neural networks using these floating-gate adaptive node circuits.

For a set of inputs, $\mathbf{x}$, (where we assume differential signals are available) the differential structure allows four-quadrant weight-signal multiplication and gives the synapse output current

$$I_{out} = I_{s0_i}(2 + w_i + w_i x_i). \tag{65}$$

For the synapse shown in Fig. 45, the floating gate of M1 adapts, giving w_i, and the floating gate of M2 is programmed to I_{s0_i}, providing the zero-reference for the differential amplifier. A typical circuit response illustrating the output of a single synapse appears in Fig. 47. Figure 47b shows the component of the single-synapse response due to w_i. The constant term and signal term have been removed. Figure 47c shows the envelope of the signal-only term, where it is observed that an amplitude change in the output signal follows the weight change in Fig. 47b, as expected. Summing individual synapse currents through KCL leads to the node output current

$$I_n = I_{n0}(2N + \sum_{i=0}^{N} w_i + \sum_{i=0}^{N} w_i x_i) \tag{66}$$

which, after low-pass filtering to remove the constant term ($2N$) and the slow-timescale term ($\sum_{i=0}^{N} w_i$), yields the standard weighted sum, represented symbolically in Fig. 43.

The computed synapse current and the equilibrium weight value of the synapse both depend on exponential functions of terminal voltages, hence we use logarithmic pre-distortion to linearize the input (eliminating unwanted signal harmonics) for feedforward computation, as well as idealizing the correlation learning rule. The resulting feedforward computation circuit is a current mirror, whose input is the leftmost transistor in Fig. 42; differences in

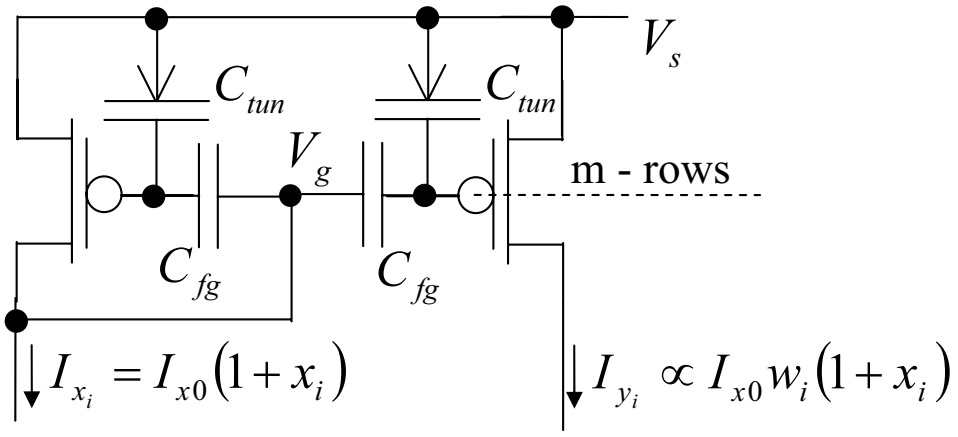

Figure 46: The bandwidth of the pre-distort input current mirror limits the speed of the feedforward computation. Bandwith depends on the bias current, I_{x0}, and the total capacitance seen at the node, V_g. One pre-distort mirror feeds the input current to each column of the adaptive matrix, driving the capacitance on m-rows of the matrix.

floating-gate charge set the gain of the current mirror [44, 1]. Because these circuits only add complexity to the periphery of the array, as shown in Fig. 45, they do not significantly increase the size of the synapse allowing for large arrays.

We can model the feedforward computation by a low-pass filter where the weight of the synapse corresponds to the passband gain. The speed of the feedforward computation is essentially equivalent to the time constant of the low-pass filter, which is determined by the capacitance on the input and output nodes of the system and by the currents driving these nodes. The output node is a very low-impedance point with small capacitances in comparison with the input node; therefore the input node contributes the dominant pole for the low-pass behavior. Figure 46 illustrates the details of how we determine the resulting time constant of the feedforward computation for m rows (network nodes) as

$$\tau = (m + 1)(C_T - C_{fg})U_T/\kappa I_{x0} \tag{67}$$

where C_T is the total capacitance at the floating gate of each transistor which includes C_{tun} and parasitic capacitance in addition to C_{fg}. For actual capacitor values of $C_{fg} \approx 25$ fF and $C_T \approx 30$ fF, we obtain $\tau \approx 0.2(m + 1)$ μs for a bias current of 1 nA. Therefore, if we take a 100 input (columns) by 100 node (rows) array, and a typical range of bias currents from 10 nA to 1 μA, we obtain a range of bandwidths governing the multiplication speed on the order of 10 kHz to 1 MHz, respectively. Finally, given a 5 V power supply, we estimate

that, operating at 0.2 ns, our system performs feedforward matrix-vector computations equivalent to about 100,000 MIPS/mW at 5 mW of power.

5.2.2 Weight Adaptation and the Floating-Gate Correlation Learning Rule

To realize continuous-time on-chip learning we take advantage of device physics, specifically, we take advantage of Fowler-Nordheim electron tunneling and hot-electron injection. The currents resulting from each of these processes are approximately six-orders of magnitude smaller than signal currents, yielding a slowly adapting weight parameter which can be considered approximately constant during weight-signal computation. Current due to Fowler-Nordheim tunneling depends on an exponential function of the voltage across the barrier of a MOS capacitor formed between the floating-gate and an external tunneling junction. We assume the affects on tunneling current due to changes in the *weight* and *signal* voltages on the floating-gate are small in comparison to those due to hot-electron injection, and therefore treat it as a nearly-constant bias current for the adaptation operation. Hot-electron injection current depends proportionally on a product of the channel current with an exponential function of the drain-to-source voltage of the device. Utilizing these nonlinearities inherent in the injection process provides a compact realization of the correlation learning rule at the heart of LMS, which is the key to implementing practical, large-scale on-chip learning adaptive systems. Electron tunneling and hot-electron injection allow us to add charge to and remove charge from the floating gate, providing us with a weight update mechanism. We extensively discuss the physics of floating-gate circuits elsewhere [31, 34]. Our previous work illustrated how the weight update mechanism leads to a learning rule based on correlations between the gate and drain voltages of these devices [14, 30, 34].

Next, we need to derive the floating-gate update equation; we base this derivation on rigorous treatments presented elsewhere [29, 30]. We start from writing KCL at the floating-gate node as

$$C_T \frac{dV_{fg}}{dt} = I_{fg0} \left(- \left(\frac{I}{I_{s0}} \right)^{\alpha} e^{-\tilde{V}_{es}/V_{inj}} + e^{-\tilde{V}_{fg}/V_x} \right) \tag{68}$$

where C_T represents the total capacitance connected to the floating-gate, I_{fg0} is the equilibrium bias current, and I and I_{s0} are the channel current and bias current, respectively, through M5. Terminal-voltage bias conditions determine V_{inj}, an injection device-dependent parameter, and V_x, a tunneling device-dependent parameter, which in turn define $\alpha = 1 - \frac{U_T}{V_{inj}}$ and $\beta = 1 + \frac{U_T}{\kappa V_x}$ [30].

We substitute for the I/I_{s0} term by noting that the channel current is set through a current mirror (M6 through M7) with a power slightly greater than 1 (we define as n). The power law of the current mirror (M6 and M7) is set by the ratio of the total capacitance at the floating-gate to the feedback capacitance for M7. We substitute for the $\tilde{V}_{es}$ term in two parts related to ΔV_e and $\tilde{V}_s$. First, we get ΔV_e through the circuit defined by M5, M6, and M7 (assume the current mirror is ideal):

$$I_{s0_2} e^{-\Delta V_e / V_{inj}} = I_{s0_2}(1 + e(t)), \tag{69}$$

where $e^{-\Delta V_e / V_{inj}} = 1 + e(t)$, and we call $e(t)$ the error signal for the synapse. We set the exponential dependence of V_e to the channel current of M5 by the ratio of its feedback capacitance to the total capacitance of its floating-gate node. The post-distort circuit of Fig. 45 transforms the exponential dependence of injection current on drain voltage into a linear dependence on error current. Buffering the error-signal output voltage removes interference between the desired error signal current and the currents out of the learning branches of the synapses.

Second, we substitute $\tilde{V}_s = \kappa \tilde{V}_{fg}$, because we have a follower circuit. We substitute for $\tilde{V}_{fg}$, noting that it is the sum of a fast timescale change in the floating-gate voltage (related to 1 + x) and a slow timescale change in the floating-gate voltage (related to w). We substitute for dV_{fg}/dt by again expanding into slow timescale and fast timescale components. We substitute w for the slow timescale component. If we integrate the entire equation over the fast timescale, defined as $E[\cdot]$, due to its relationship to finding statistical mean of piecewise ergodic signals, eliminating the fast timescale component for dV_{fg}/dt, we get

$$\tau \frac{dw}{dt} = (1 + w)^{\alpha}(1 + x)^{\alpha n - U_T / V_{inj}}(1 + e) - (1 + w)^{\beta}(1 + x)^{\beta - 1}, \tag{70}$$

where $\tau = C_T U_T/(\kappa I_{fg0})$. We choose n such that $\alpha n - U_T/V_{inj} = 1$. In practice, we use a bank of capacitors to digitally select the correct capacitor ratio for M4 and M5. A similar analysis leads to the weight update equation for the circuit in Fig. 42 and Fig. 45 as

$$\tau \frac{dw}{dt} = (1+w)^{\alpha}(1+x)(1+e) - (1+w)^{\beta}(1+x)^{\beta-1}, \tag{71}$$

which results from the definition of the weight due to the charge on the floating node and KCL analysis involving the tunneling and injection currents; both currents have been normalized to the equilibrium bias current. If we put a signal on each column of the tunneling line, one could eliminate the nonlinearities in the second term as well.

Assuming e is zero mean for this discussion, we can compute the steady state result for a constantly applied input and error signal. Note that one steady state occurs when $w = -1$, which is unstable for this configuration as long as the output of the follower circuit connected to the floating-gate is not at V_{dd}. Therefore, the synapse always has a bounded solution if x and e are driven at a constant amplitude. In practice, e will be defined by the learning dynamics; therefore we must consider the system as a whole to determine system stability.

Next, we evaluate and approximate (70) to get a weight update equation that fits within an adaptive filter or neural network viewpoint [35, 36]. We expand the input, x, in a second-order approximation, and evaluate $E[\cdot]$ to get

$$\begin{aligned} \tau \tfrac{dw}{dt} &= (1 + E[xe])(1+w)^{\alpha} \\ &= -\left(1 + \tfrac{1}{2}\tfrac{U_T}{\kappa V_x}\left(1 - \tfrac{U_T}{\kappa V_x}\right) E[x^2]\right)(1+w)^{\beta-1}. \end{aligned} \tag{72}$$

Furthemore, applying a first-order approximation to w leads to

$$\tau \frac{dw}{dt} = E[xe] - a_1 E[x^2]\left(-a_2 + \alpha(E[xe]) - a_3 E[x^2]\right) \tag{73}$$

where $a_1 = \frac{U_T}{\kappa V_x}\left(1 - \frac{U_T}{\kappa V_x}\right)$, $\epsilon = \frac{U_T}{V_{inj} \| \kappa V_x}$, and $a_3 = \frac{U_T}{\kappa V_x}\left(1 - \left(\frac{U_T}{\kappa V_x}\right)^2\right)$. By comparing typical device parameters, we see that ϵ, a_1, and a_3 are first-order terms ($\approx \epsilon$), and $\alpha \approx 1$. Depending on the technology, we obtain ϵ in the range from 0.03 to 0.3.

In the next section, we show that the input variance terms are small enough to be ignored in most cases, and for the remainder of the discussion we simplify (73) as

$$\tau \frac{dw}{dt} \approx -\left(\epsilon - \alpha E[xe]\right) w + E[xe] \tag{74}$$

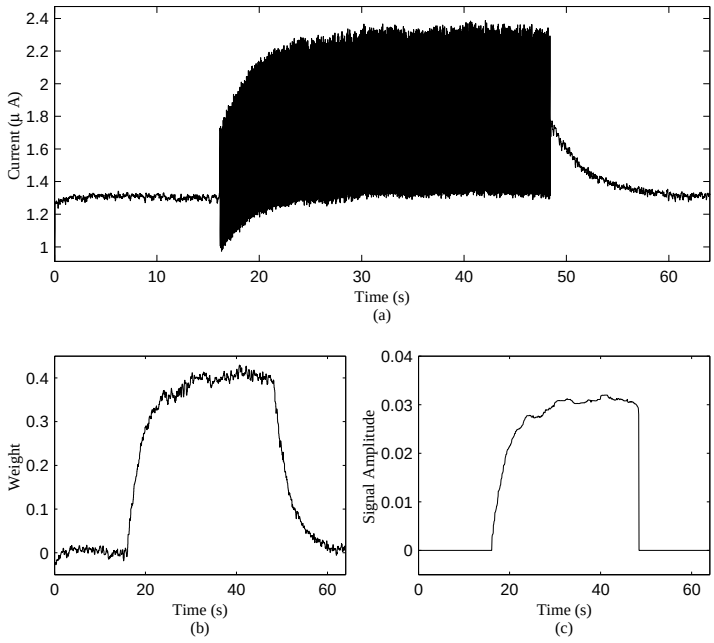

Figure 47: Amplitude correlation results for sinusoids of same frequency and phase with input and error signal amplitudes of 0.3. (a) Synapse output current vs. time. Sinusoidal signals are turned on at 16 seconds and turned off at 48 seconds. (b) Extracted dimensionless weight value vs. time, showing convergence to steady-state due to correlations affected by weight decay (c) Extracted dimensionless signal amplitude vs. time, which follows a path similar to the weight, increasing from zero to a steady-state value. Without an input signal, the output is zero due to multiplicative effects.

Note that the coefficient in front of w, is of first order size but the correlation is zeroth order.

5.2.3 Amplitude Correlation Experiments

Comparing our floating-gate correlation learning rule (74) with the LMS analysis of (62) we see that our rule exhibits two components: a correlation term and a weight decay term. We desire to characterize these terms and compare their relative sizes. Note that weight decay increases for large-error signals, which may improve generalization in the presence of large errors; weight decay decreases when the error becomes small. We verify the correlation learning rule approximation through amplitude-correlation experiments. In what follows, we present data to illustrate this learning rule; to simplify our analysis, we approximate the weight decay solely as ϵ and point out deviations where necessary.

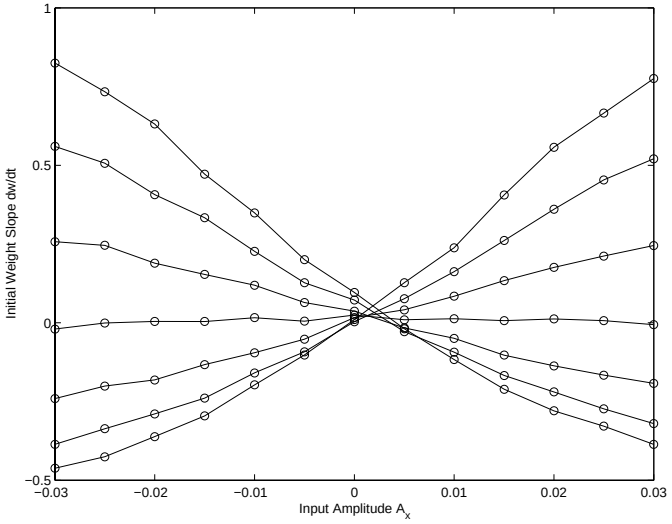

Figure 48: Basic correlating floating-gate synapse measurements for an individual isolated synapse element. We measured the initial slope from this synapse for multiple input (sinewave) amplitudes near the zero-input steady-state condition. This initial slope shows the correlation function used by this LMS synapse; steady-state solution has less meaning for an LMS synapse with small weight decay.

Making a step change in the input and error signal amplitudes from an initial condition of $w = 0$ yields the (of course, to get these results, we need the signal bandwidth to be much larger than the adaptation bandwidth / time-constant)

$$\tau \frac{dw}{dt} = E[xe] - a_1 E[x^2] \tag{75}$$

The input-variance term is typically much smaller than the correlation term, and typically smaller than a_2 when the amplitude of the input signal, x, stays within about less than half of its full range which is bounded between -1 and 1. Although we can roughly ignore $E[x^2]$ for the remainder of our discussions. it is important to recognize these terms, as they can be a limiting effect for some algorithms.

Each experiment we discuss involves sinusoidal signals since they provide an easily generated and commonly used set of test-signals that allow us to readily observe system nonlinearities via harmonic analysis and take advantage of Fourier theory in designing and analyzing experiments, and facilitate analysis of experimental results. Fig. 47, illustrates a typical synapse response for a simple amplitude correlation experiment, where we apply $x = A_x \sin(\omega t)$ and $e = A_e \sin(\omega t)$ as the input and error signals. The measured output

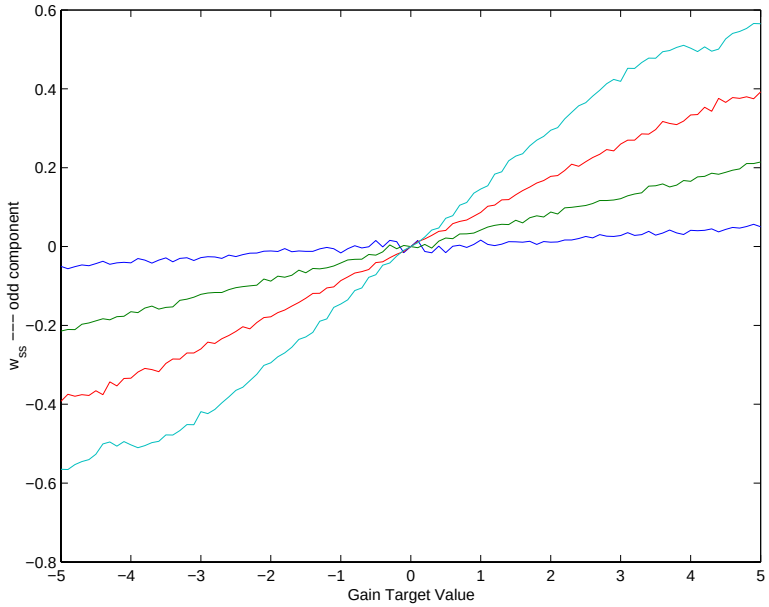

Figure 49: Results for a single-weight LMS experiment. The input signal is a sinusoid of fixed amplitude; the target signal has frequency and phase identical to the input, but varying amplitude. Steady-state weight values for the single-weight LMS experiment are plotted versus target signal amplitude. The steady-state weight value is a linear function of the target signal amplitude. We have plotted the results for several gain levels.

current of the synapse appearing in Fig. 47a shows the fast-timescale output signal riding on a slow-timescale deviation in the current bias due to weight adaptation. Normalizing the current to its equilibrium value and low-pass filtering yields the extracted weight value shown in Fig. 47b; the time-constant, τ, was fit to an RC-model and is approximately 3 seconds. Removing all correlating signals, leaves only weight decay, which returns the synapse to its initial equalibrium. Finally, Fig. 47c exhibits an increase in output signal amplitude from zero to a steady-state value of $w_{ss}A_x$. To illustrate weight dependency on statistical properties of applied signals, remaining figures plot either steady-state weight values, w_{ss}, or linear fits of initial weight dynamics, $\frac{dw}{dt}$, as functions of experiment parameters such as signal amplitude or phase angle.

We first observe that for $w \approx 0$, the initial slope of the weight dynamics, $\frac{dw}{dt}$, should yield correlations between the applied input and error scaled by the weight decay as $w_{ss} = E[xe]/\epsilon$. Fig. 48 demonstrates $\frac{dw}{dt}$ for $w \approx 0$ obtained from linearly fitting the initial slopes of each weight convergence. We see a compressive nonlinearity at high amplitudes due to

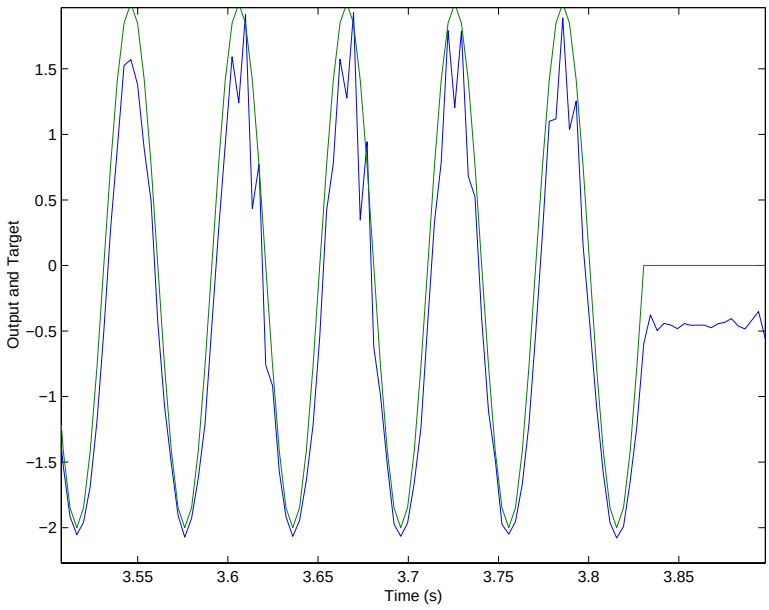

Figure 50: Results for a single-weight LMS experiment. The input signal is a sinusoid of fixed amplitude; the target signal has frequency and phase identical to the input, but varying amplitude. Transient solution for a given target signal amplitude of the single-weight LMS experiment shows the output signal for this trial tracking the target signal.

higher-order terms in the Taylor series expansion. Fig. 48 also shows an offset at $A_x = 0$ arising from variance of the error signal due to slight mismatch between the post-distort circuit and V_{inj} The size of this term tends to be similar to that of the weight decay for correlating synapses [14]. These results will be similar to the steady-state solutions when small amplitude signals are applied; for larger signals we will see nonlinearities due to the small amount of weight decay.

5.3 The Least-Mean-Square (LMS) Learning Rule

Our new synapse enables the first continuously-adapting floating-gate LMS algorithm. We previously demonstrated LMS learning for uncorrelated inputs [13]; we now demonstrate the effects of error-signal feedback in a single-synapse supervised-learning experiment and present results from a two-input single-node circuit for general inputs.

Unlike the pure correlation case, supervised learning requires feedback of an error signal defined as the difference between the computed output and a given target. However, before deriving the error from the single-node output current given in (66), we need to separate

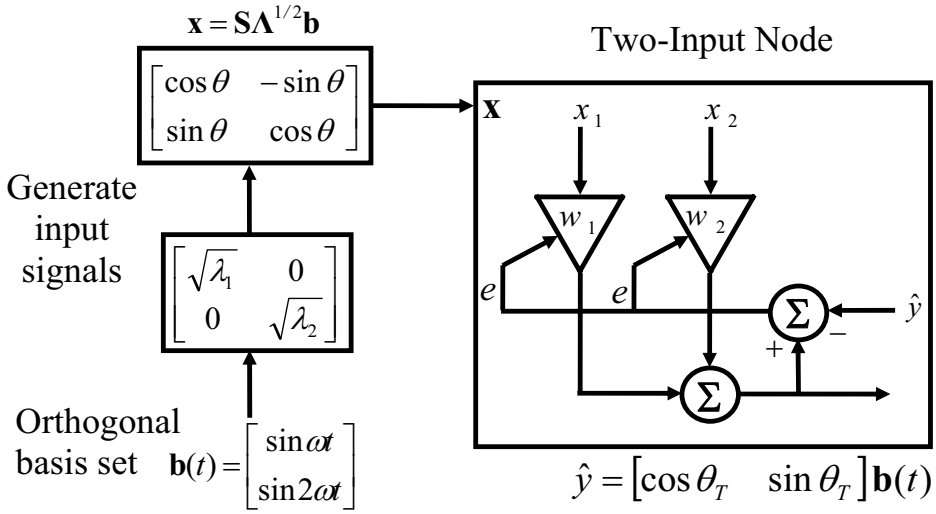

Figure 51: Experimental setup for examining Least-Mean-Squares behavior in a two-input node. A scaling operation followed by application of a rotation matrix to an orthogonal signal-space basis of harmonically related sinusoids yields the system input signals; the fundamental sinusoid is chosen as the target. The experiment runs for different values of θ, uniformly sampled from a circle.

the output signal, $y = \mathbf{w}^T\mathbf{x}$, from the bias and slow-timescale weight terms $(2N + \sum w_i)$ by low-pass filtering, as illustrated in Fig. 47. The first LMS case concerns supervised adaptation of a single weight, which illustrates circuit dynamics for a manageable number of parameters. The output and target signals in this experiment are given by $y = wx$ and $\hat{y} = Ax$, which lead to the error signal $e = (A - w)x$. An ideal LMS response would force w to equal A. Instead, we find that with weight decay, as exhibited in (62) and (74), the LMS rule yields $w_{ss} = A/(1 - \epsilon/E[x^2])$, when the term $\alpha E[xe]$ is neglected. Including this correlation term of the weight decay in (74) adds terms on the order of $O(\epsilon^2)$, justifying our first-order approximation. Figure 49 shows the weight value tracking various target gains, and Fig. 50 shows the output sinusoid tracking the target signal for a single trial.

5.4 The Two-Input Adaptive Node

Multiple-input LMS learning systems provide deeper results than the single-synapse case. Fig. 51 illustrates the setup for the simplest possible multiple-input LMS experiment comprising two-synapses. Two-dimensional input data allow visualization of the weight results

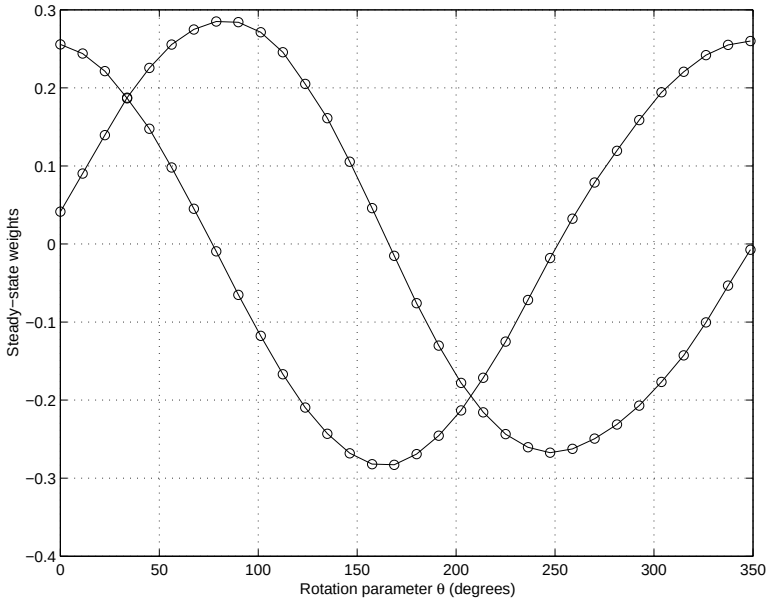

Figure 52: Two-input Least-Mean-Squares experimental results for the source-follower floating-gate synapse circuit. Measured data for the case $\theta_T = 0$ and $\lambda_1 = \lambda_2$ show steady-state weight dependence on the parameter, θ, of the two-dimensional input mixing-matrix. As expected from equation (84), we get a cosine curve for the first weight, and a sine curve for the second weight.

as a set of points in the plane, making it easier to observe the effects of weight decay on the *magnitude* and *direction* of the weight vector.

First, we choose an orthonormal basis for the signal space; in these experiments, we choose harmonically related sinusoids given by

$$\mathbf{b}(t) = \sqrt{2} \begin{bmatrix} \sin \omega t \\ \\ \sin 2\omega t \end{bmatrix} \tag{76}$$

which yield the desired orthonormal property

$$E[\mathbf{b}(t)\mathbf{b}^T(t)] = \mathbf{I}. \tag{77}$$

We construct the input signals from a linear combination of the basis signals as

$$\begin{aligned} \mathbf{x}(t) &= \mathbf{M}\mathbf{b}(t) \\ &= \mathbf{S}\mathbf{\Lambda}^{1/2}\mathbf{b}(t) \end{aligned} \tag{78}$$

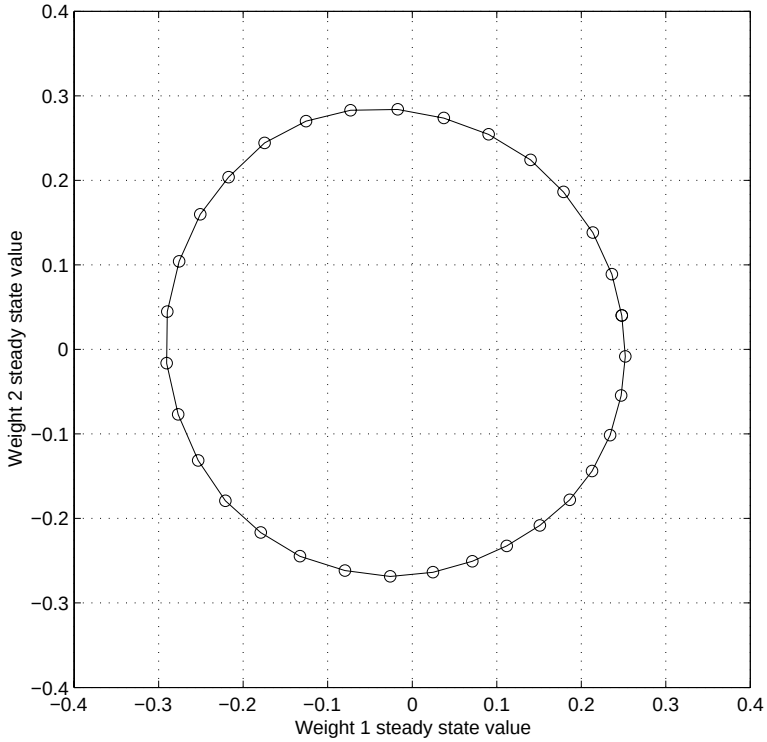

Figure 53: Two-input Least-Mean-Squares experimental results for the source-follower floating-gate synapse circuit. Measured data for the same case is plotted as one steady-state weight versus the other. An ideal LMS rule would produce a circle similar to our circuit results.

where the mixing matrix, $\mathbf{M}$, is composed of an orthonormal basis for the weight space determined by a particular value of a single parameter, θ, given by

$$\mathbf{S}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{79}$$

and an eigenvalue matrix

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{80}$$

which generates the input signals for the learning circuit according to

$$\mathbf{x} = \mathbf{S}\mathbf{\Lambda}^{1/2} \tag{81}$$

as illustrated in Fig. 51.

Finally, the target signal, $\hat{y}$, results from a linear combination of the signal-space basis, and will be dependent on a single parameter, θ_T, as shown by

$$\hat{y} = [\cos\theta_T \;\; \sin\theta_T]\mathbf{b} \tag{82}$$

Mathematical analysis of this experiment assuming constant weight-decay dynamics (the simplest case of (74) when the correlation term of the weight decay is ignored) and based on the results given in (62) yields the steady-state weight solution,

$$\mathbf{w_{ss}} = \mathbf{S}(\theta) \begin{bmatrix} \frac{\sqrt{\lambda_1}}{(\lambda_1+\epsilon)}\cos\theta_T \\ \\ \frac{\sqrt{\lambda_2}}{(\lambda_2+\epsilon)}\sin\theta_T \end{bmatrix}. \tag{83}$$

Measured results from the synapse circuit appear in Figure 52 and Figure 53. Each experiment comprises 32 trials of the rotation parameter, θ, uniformly sampled from 0 to 2π with θ_T held constant. We performed the first experiment with $\lambda_1 = \lambda_2$ and $\theta_T = 0$. Given these parameters, we expect the circuit to yield a weight vector given by

$$\mathbf{w_{ss}} = \begin{bmatrix} cos\theta \\ \\ sin\theta \end{bmatrix} \tag{84}$$

Figure 52, shows the steady-state weight values for each of the two weights as a function of the rotation angle, θ. Steady-state values for the first and second weights approximate $\cos\theta$ and $\sin\theta$ respectively, as expected; a plot of the second vs. the first component of $\mathbf{w_{ss}}$ results in a circle as shown in Fig. 53.

In the case where $\lambda_1 \neq \lambda_2$, from (83) we expect a weight-space plot of the first vs. the second node weight to yield an ellipse when θ is swept from 0 to 2π for a fixed value of $\theta_T = \pi/3$. Figure 54 exhibits the measured data compared with numerically computed results from (83) in a weight-space plot, which shows that the weights approximate an ellipse quite well. From the fit to the measured data, we estimate the weight decay, $\epsilon \approx 0.1$. The measured data exhibit a slight rotational difference from the computed results which is probably due to the higher-order terms in the weight decay which have been ignored in this analysis.

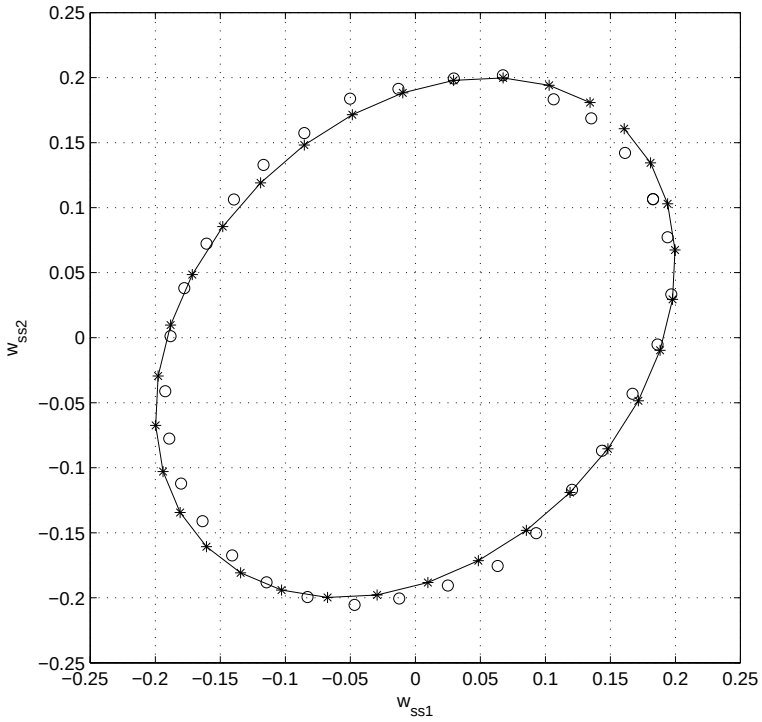

Figure 54: Two-input Least-Mean-Squares experimental results for the source-follower floating-gate synapse circuit. Illustration of measured data compared with model results computed from equation (83) assuming non-zero constant weight decay for the case where $\lambda_1 = 1$, $\lambda_2 = 2$, and $\theta_T = \pi/3$. The gain of the filter and the amplitude of the input signals determine the scale of the ellipse. The open circles show steady-state values measured from the actual circuit. The asterisks and the corresponding fit-line illustrate computed model results. We observe good agreement between the measured data and the computed values. As predicted in the text, the results form an ellipse.

5.5 Concluding Comments

We introduced a source-follower floating-gate synapse as an improvement over our earlier source-degenerated floating-gate synapse in terms of relative weight decay. While the ideal LMS solution does not exhibit weight decay, it is impossible to build a physical system that does not have some form of decay inherent in the system. However, zero weight-decay would result in a learning system which would be difficult to control, therefore we desire a circuit with a small, non-zero weight decay term. Additionally, weight decay adds some algorithmic benefits to the learning system, such as the ability to track non-stationary processes. Data for a general LMS experiment obtained from a two-input circuit exhibited good agreement with a mathematical model of the system.

CHAPTER VI

FROM SIMPLE NODES TO ADAPTIVE NETWORKS

Having investigated the properties of floating-gate synapses in two-input adaptive systems, we now turn to the more general n-input case. For these investigations we have designed and fabricated a multiple-input, multiple-node chip through MOSIS. Due to limitations of our current PC-based experimental set-up, we have developed an FPGA-controlled custom circuit board for real-time tests up to audio frequencies. This more advanced IC and test set-up, will allow for the study of LMS in the case of more than two inputs (for example, Fourier approximation and adaptive channel equalization), as well as exploration of multiple-node networks and unsupervised learning algorithms.

Because no good models of continuously-adapting floating-gate devices exist, we have developed an all-transistor non-floating-gate circuit model of the source-follower floating-gate synapse which has been designed to explore weight decay issues in large-scale networks. As noted in the previous chapter, *any* physical implementation of the least-mean-square learning algorithm will exhibit some weight decay. Due to the nature of this model, we are actually able to control the amount of weight decay present through design choices. Results from simulations of least-mean-square learning experiments demonstrate the learning properties of this simulation model.

We finish our discussion of systems by illustrating how arrays of floating-gate synapses could be useful in practical adaptive systems through an example of adaptive channel equalization.

6.1 The n-Input Node

In chapter 5, we derived the weight equation for a single source-follower floating-gate synapse to be

$$\tau \frac{dw_i}{dt} = -(\epsilon - \alpha E[x_i e])w_i + \alpha E[x_i e]. \tag{85}$$

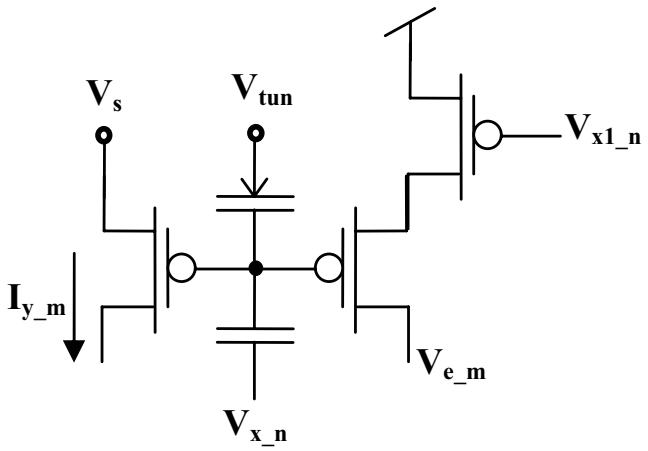

Figure 55: The source-follower floating-gate synapse (SFFG) that was introduced in the preceding chapter. The chip described in the text contains an array of 4 rows and 16 columns of this circuit.

In the following, we desire to express this learning rule in matrix-vector form. Additionally, in chapter 5, we examined how constant weight decay affects the steady-state weight solution; here we wish to examine how the correlation-dependent weight decay term affects the steady-state weight solution.

We begin our analysis by recognizing that the ideal solution, which we denote $\mathbf{w}^*$, is given by

$$\mathbf{w}^* = \mathbf{Q}^{-1}\mathbf{r} \tag{86}$$

where $\mathbf{r} = E[\mathbf{x}\hat{y}]$ is the cross-correlation between the input signals, $\mathbf{x}$, and the target signal, $\hat{y}$. Next, we observe that in the ideal case, our system output

$$y = \mathbf{w}^{*T}\mathbf{x}, \tag{87}$$

attempts to represent $\hat{y}$ in a manner that minimizes the mean-square-error between the two values, resulting in e_{min}. We further assume that

$$E[\mathbf{x}e_{min}] = 0, \tag{88}$$

i.e. the residual error signal is orthogonal to every input-channel signal. Now we would like

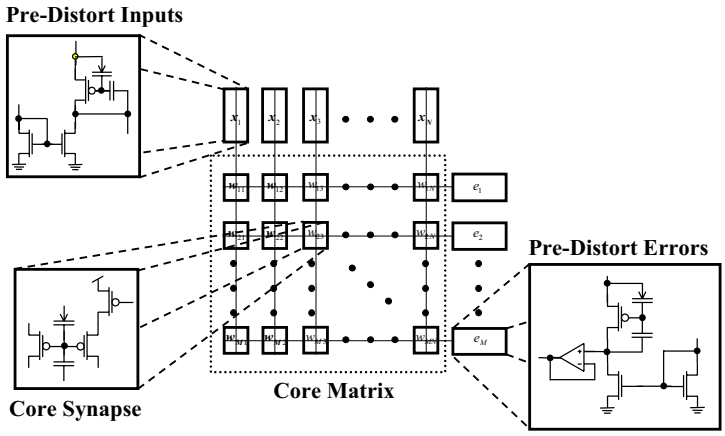

Figure 56: Functional diagram of the SFFG array chip described in this chapter. The chip, fabricated through MOSIS in a $0.5\mu m$ process, contains an array with 16 column inputs and 4 row outputs. Additional circuitry around the periphery allows for programming the array as desired. The pre- and post-distort circuitry particular to this design and the SFFG cell appear in the accompanying figures.

to solve

$$\tau \frac{dw_i}{dt} = -(\epsilon + \sum_{j \neq i}(w_j - \alpha w_j^*)q_{ij} - \alpha q_{ii} w_i^*)w_i - q_{ii}w_i^2 - \alpha(\sum q_{ij}w_j - r_i)) \qquad (89)$$

and observe how the correlation weight decay affects the ideal solution, $\mathbf{w}^*$. Experimental data presented in the preceding chapter shows that the weight converges close to the constant weight decay case; we wish to further understand how the signal-dependent weight decay affects this solution so results in higher dimensions can be predicted.

6.2 A Multiple-Input/Multiple-Node Adaptive Floating-Gate Integrated Circuit

Figure 56 illustrates an array with four nodes and sixteen inputs; this circuit has been fabricated through MOSIS in a 0.5μm process. The building blocks of this array, the SFFG synapse cell, and the pre- and post-distort circuits appear in Fig. 55, Fig. 57, and Fig. 58 respectively. Larger arrays are possible, but pin limitations led to the small size of the array for this design; there is still much room on the chip for more synapses and nodes. In an attempt to save pins, the 16 inputs of the array enter the chip through a single pin which feeds an analog multiplexer, and internal linear voltage-to-current converters

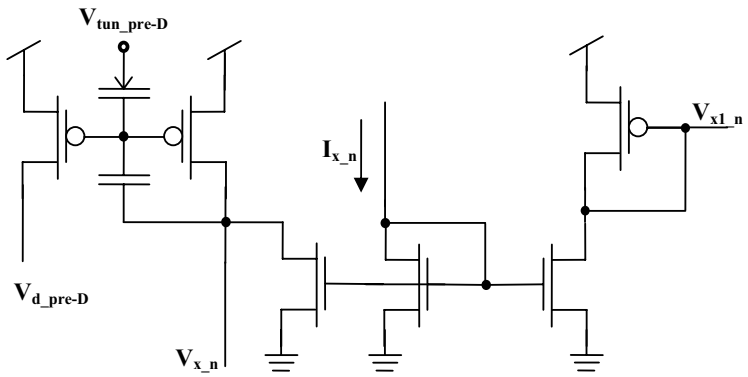

Figure 57: The pre-distort circuit for each column of the array includes an extra floating-gate pFET for adjusting the charge on the charge on gate of the pre-distort transistor. The nFET mirror provides the input signal to the pre-distort circuit as well as feeds the pFET input to the source-follower bias transistor in the SFFG synapse for the correlation learning rule.

generate current inputs for the pre-distort circuitry from the voltage inputs (see Fig. 59). The outputs of each node consume four pins total; the outputs are current values which will need to be transformed into voltages before analog-to-digital conversion. An additional 8 pins provide the four target signals necessary for supervised learning experiments. This chip also contains infrastructure for programming any element in the array to a desired value, as well as for programming the pre- and post-distort circuitry to desired bias conditions. A capacitor bank with individual capacitors which can be switched in and out allows the ability to select a desired post-distort value, as shown in Figure 58.

6.2.1 Hardware Test and Development System

Due to limitations on sampling speed and the number of channels available with the current PC-based measurement system, we designed a custom circuit board for test and measurement of the adaptive chip. The custom board interfaces to Matlab through an Altera FPGA board running a Nios processor with additional custom VHDL code written for more specialized I/O control. A photograph of the board appears in Figure 60.

The heart of the adaptive experiment circuit board consists of two of the array chips, one

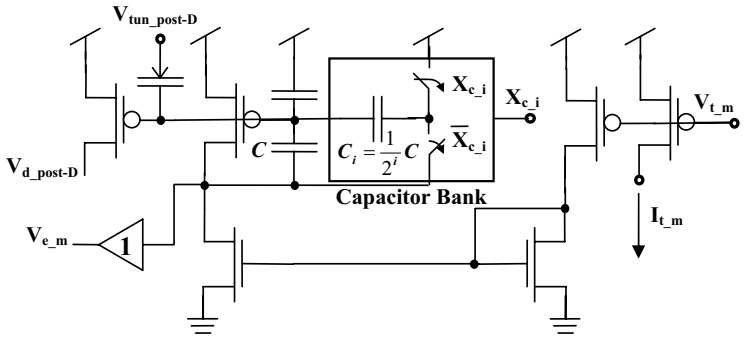

Figure 58: The post-distort circuit for each row of the array includes a switchable capacitor bank that allows for tuning the post-distort factor. The floating-gate pFET on the far left provides a means for setting the charge on the post-distort circuit gate node. The two pFET transistors and the nFET mirror on the right provide the error signal for the correlation learning rule.

programmed to provide the zero-point for the differential structure and the other adapting around this zero-point. Two single-channel parallel D/A converters provide input signals to the adaptive and programmed differential arrays. Five four-channel serial D/A converters control various bias voltages including chip power supply and tunneling voltages set through bipolar junction transistors (BJTs). Logic control signals applied directly from the FPGA control whether the system is in run or program mode, determine which row and column to select for programming, and reset the I-to-V converters during current measurments.

The adaptive chip generates output currents. These currents need to be transformed to voltages before analog-to-digital conversion. Current-to-voltage converters based on an op-amp integrator (illustrated in Figure 62) provides the necessary conversion. Twelve of these converters are divided among three quad-op amps; eight of these measure the four row-current outputs of each chip while the four remaining used for programming-current measurement Three parallel-output A/D converters, each with four channels, interface the current-to-voltage converters to the FPGA control board.

We plan to eventually move all of the current measurement and data converter functions currently performed on the custom circuit board on to the array chip with the goal of obtaining a standard, simple interface for adaptive system testing. Moving this functionality on-chip will also allow for much larger arrays to be designed and tested. In addition,

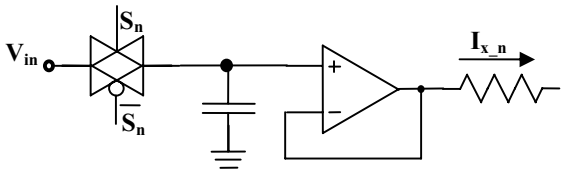

Figure 59: Sample-and-hold circuit formed by a capacitor and a unity-gain follower provides analog multiplexing allowing one input pin to drive sixteen synapse inputs.

the software which has been developed in conjunction with the custom circuit board for experimental testing provides the starting point for a seamless, user-friendly functional interface for chip testing. This functional interface will hide circuit-level details from the user and allows higher-level algorithm and system level experiments.

6.3 All-Transistor Synapse Circuit Model and Simulation

Figure 63 shows the all-transistor synapse (ATS) model of the source-follower floating-gate synapse. The ATS model allows us to more explicitly study and control weight decay by looking at the effects of different channel lengths for M_{a_bias} and M_{corr}. These channel lengths control the weight decay by determining the output resistance of the high-gain amplifier formed by M_{a_bias} and M_{corr}. We will refer to this pair of transistors collectively as the correlation amplifier. A pair of capacitively-coupled current mirrors, formed by the four transistors M_{i+}, M_{s+}, M_{s-}, and M_{i-} provides a four-quadrant multiplication of the weight and input signal. Differences between dc bias points on the gates of each transistor look effectively like differences in W/L-ratios in terms of current scaling factors. The bias on M_{s+} is set to an equilibrium bias voltage of V_{fg0} by the DC output value of the correlation amplifier The remaining three floating-gate transistors are programmed to the bias voltage, V_{fg0}. The difference due to $\bar{V}_{fg}$ on the gate of M_{s+} provides the weighting factor. In the correlation amplifier, M_{corr} generates the multiplication operation necessary for the correlation computation. The transistor, M_{x_corr}, mirrors the input into M_{corr} for multiplication, while M_{e1} pre-distorts the error signal for the source of M_{corr} to yield multiplication inear in the error signal. Transistor M_{e2} guarantees a bias condition on the

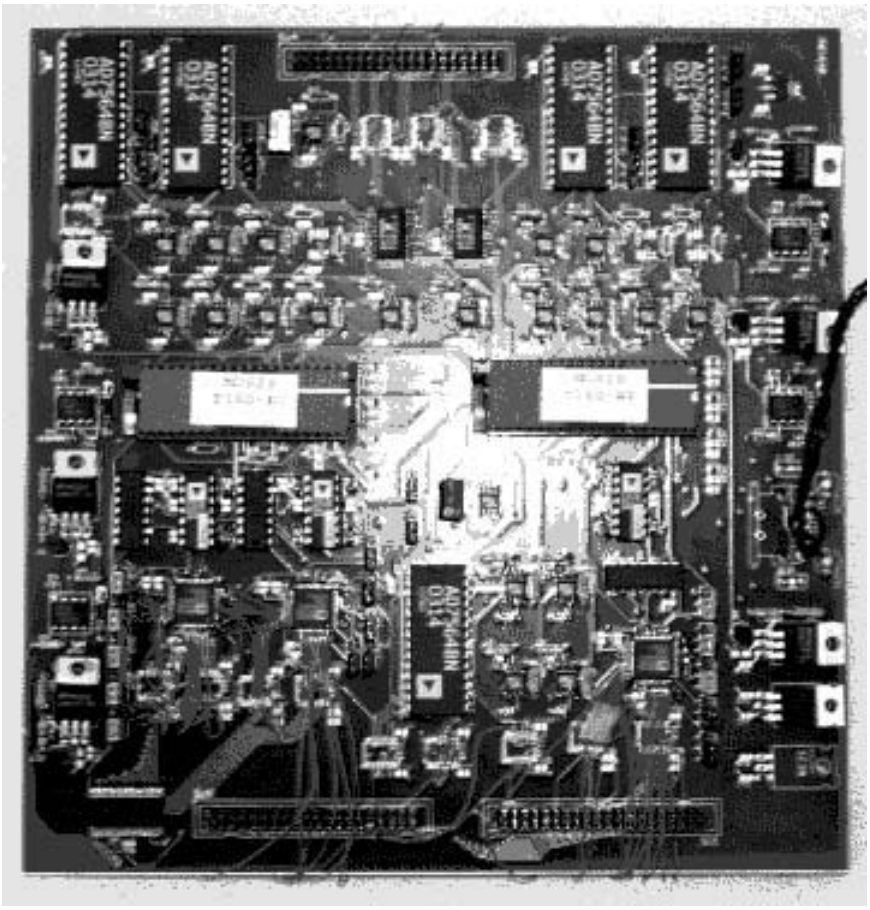

Figure 60: Photograph of the adaptive system test and measurment circuit board.

source of M_{x_corr} matching that on the source of M_{corr} due to M_{e1}. Unity-gain buffers guarantee that the source voltages on M_{x_corr} and M_{corr} are independent of their currents. Finally, M_{a_bias} provides a bias current for signed-integration; current deviations from this bias driven by M_{corr} are integrated on C to yielding correlation. We now show that this circuit does indeed provide a good all-transistor model of the SFFG synapse, how the constant weight decay term is controllable through drawn channel length, and we illustrate that the dynamics will always be similar for this class of adaptive circuits. As before, we define the weight for the adaptive circuit as

$$w = e^{-\kappa \bar{V}_{fg}/U_T} - 1 \tag{90}$$

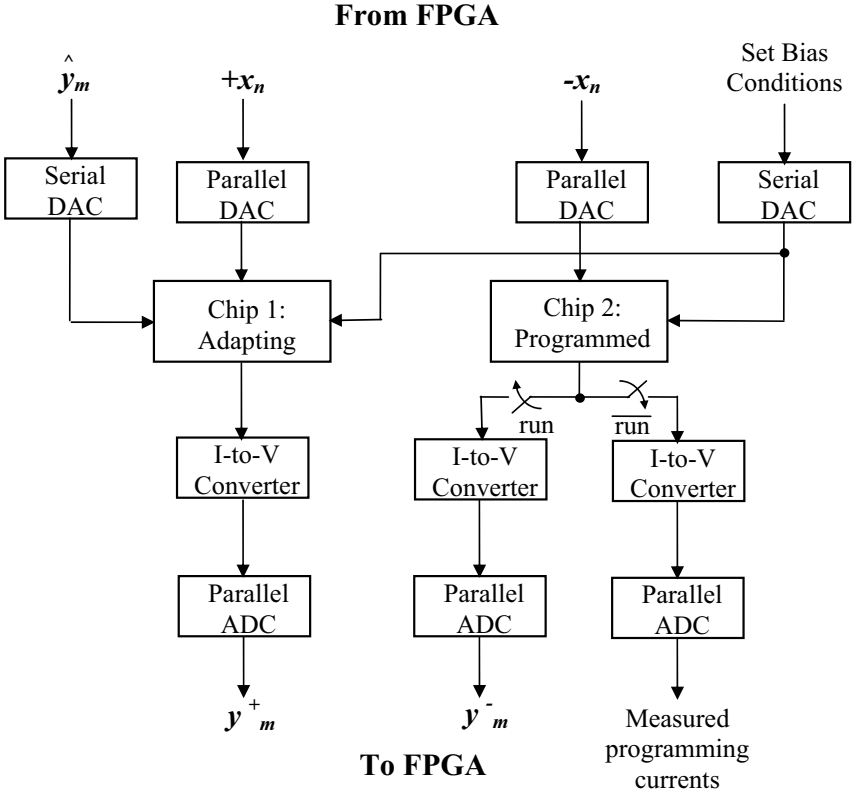

Figure 61: Functional diagram of the custom circuit board which appears in Fig. 60. Two floating-gate synapse array chips form the heart of the test system. One of the arrays has its floating-gate charge programmed to a bias point, while the second array is allowed to adapt. The programmed array provides a zero-point for for quadrant multiplication. The differential inputs, $+x_n$ and $-x_n$ are applied as a stream of time-division multiplexed samples from the FPGA through the parallel DACS. The target signals for learning, $\hat{y}_m$, are applied through through a serial DAC. The bias signals, including power-supply and tunneling voltages, are also supplied from the DACs. Finally, the output currents for each and programming currents are The signals determining the mode of operation, run and $r\bar{u}n$ are also provided by the FPGA.

or, alternatively, the floating-gate voltage can be expressed in terms of the weight as

$$\bar{V}_{fg} = -\frac{U_T}{\kappa} ln(1+w). \tag{91}$$

The derivative of this second expression proves useful in the development of the ATS learning rule and is thus

$$\frac{d\bar{V}_{fg}}{dt} = -\frac{U_T}{\kappa} \frac{1}{(1+w)} \frac{dw}{dt} \tag{92}$$

In the last chapter, it was shown that the effective resistance at the output node of a

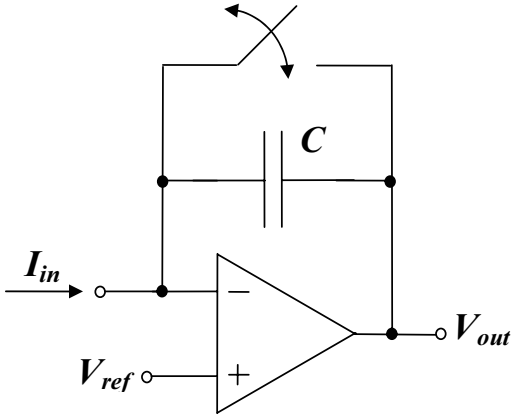

Figure 62: This integrator-based current-to-voltage converter transforms output currents from each adaptive node into voltages suitable for A/D conversion.

high-gain amplifier is given by

$$R = \frac{V_{An} \parallel V_{Ap}}{I_{fg0}}. \tag{93}$$

Performing nodal analysis on the gate node of M_{s+} substituting (92) for $\frac{d\bar{V}_{fg}}{dt}$ and using the definition in (93) leads to

$$\frac{dw}{dt} = -\frac{\kappa}{U_T C}(1+w)(\frac{1}{R}\frac{U_T}{\kappa} ln(1+w) + (I_{fg0} - I_{xe})). \tag{94}$$

Finally, we define the time constant of adaptation as

$$\tau = \frac{U_T C}{\kappa I_{fg0}} \tag{95}$$

and the constant weight decay factor, ϵ, as

$$\epsilon = \frac{U_T}{\kappa V_{An} \parallel V_{Ap}} \tag{96}$$

yielding the ATS learning rule

$$\begin{aligned} \tau\tfrac{dw}{dt} &= (1+w)(-\epsilon \ln(1+w) + E[xe]) \\ &= -\epsilon \ln(1+w) - (\epsilon \ln(1+w) - E[xe])w + E[xe] \\ &\approx \text{-}(\epsilon - E[xe])w + E[xe] \end{aligned} \tag{97}$$

where the final approximation assumes w is small and we have ignored (ϵ^2) terms. The dynamics of this equation closely resemble those of the source-follower floating-gate synapse

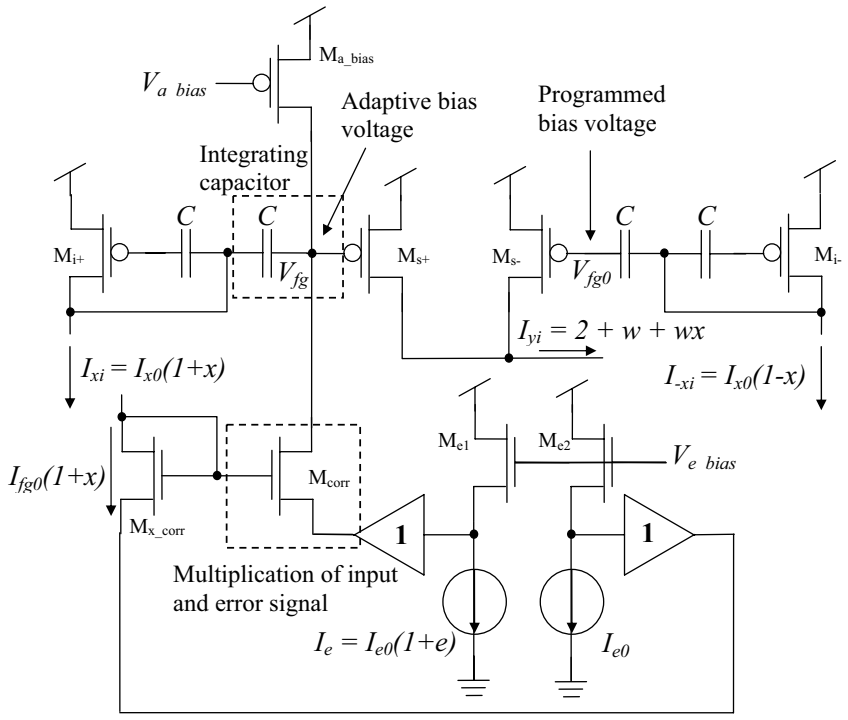

Figure 63: The differential synapse consists of two matched floating-gate current mirrors. The gate of M_{s+} adapts based on signal correlations while the gates of the remaining transistors in the mirror-pair are programmed to the bias condition, V_{fg0}. The adaptation circuit for each synapse comprises a pFET bias transistor and an nFET current mirror. The output transistor, M_{corr}, of the mirror provides multiplication of the input signal by an error signal. The source of each nFET is connected to a common buffered error-signal voltage bias. The source M_{corr} is also driven by a buffered error signal voltage which is generated as a logarithmic transform of a linear current error signal providing a multiplication linear in the error signal. All signals are represented by variation in current around a bias point.

with an unstable equilibrium at $w = -1$ and a stable equilibrium at $w = 0$. As before, our interest concerns how correlations between the input and error signals determine shifts in this stable equilibrium value and the weight decay of the system. Note that correlation-dependent weight decay terms will always arise due to our the weight definition given in (90). Because ϵ is dependent on the Early voltages, V_{Ap} and V_{An} of M_{a_bias} and M_{corr} respectively, the constant weight decay term can be controlled through design of the channel length of these devices; this property allows us to examine the effects of varying weight decay on the steady-state solution of (97). Simulation of a two-input node using the all-transistor

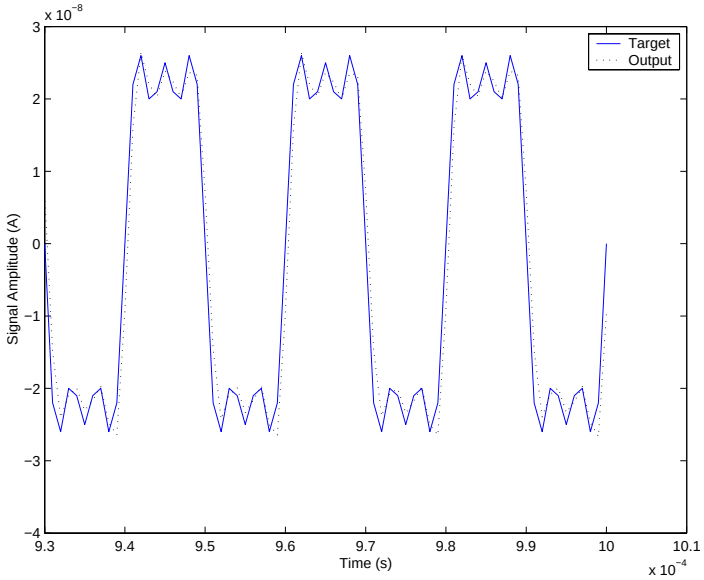

Figure 64: Comparison of system output and the square wave target signal for the ATS model Fourier LMS simulation.

synapse model yields results similar to those exhibited in the two-input node experiment described in the preceding chapter.

6.4 Fourier Series Example

Fourier series provide an excellent approach for examining LMS learning rules in both the source-follower floating-gate synapse and the all-transistor synapse model described in the preceding section. For the simulation results shown here, the inputs consist of pure harmonically related sinusoids, in particular we show a five-input case. For the target signal, we use a square wave. Each synapse in the ATS node attempts to learn the appropriate weight to reconstruct the square wave from its harmonic components. The system output and the target signals appear in Fig. 64, while their spectra are compared in Fig. 65. A look at both the time-domain signals and frequency spectra shows that the ATS system learns a good approximation to the appropriate Fourier coefficients. Discrepancies between the first, third, and fifth coefficients for the system output and the target signal are likely due to the constant weight decay term in (97). Notice, however, the introduction of small components for the second and fourth harmonics. We suspect that the signal-dependent

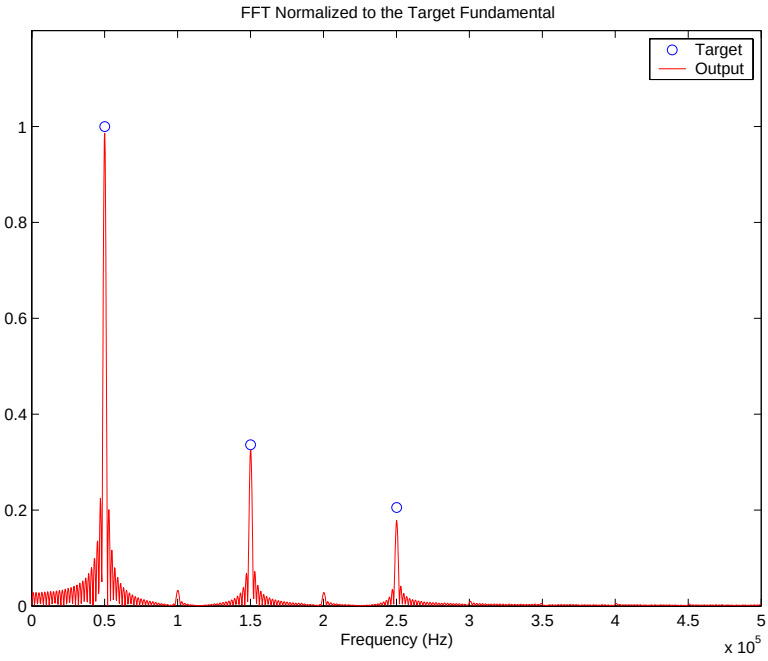

Figure 65: Fourier spectra comparing the learned system weights with the target signal square wave coefficients.

weight decay term in (97) introduces these components. Additional simulations are needed to prove or disprove this hypothesis.

6.5 Example Application: Adaptive Channel Equalization

Adaptive channel equalization provides an excellent example of a practical common use of adaptive filters. Every communication system possesses a channel representing the medium of transmission between a signal source and a receiver. In mobile communications, due to movement of the receiver and changes in the receiver's environment, this channel is time-varying, requiring an adaptive filter. In this section we propose how our adaptive floating-gate circuits could be used to solve this practical problem.

We first model the distorting channel as a linear filter with a rational transfer function as

$$H_{ch}(s) = \frac{\sum_{i=0}^{m} b_i s^i}{1 + \sum_{j=1}^{n} a_j s^j}. \tag{98}$$

The adaptive filter adjusts its parameters to approximate the inverse of the channel, thus

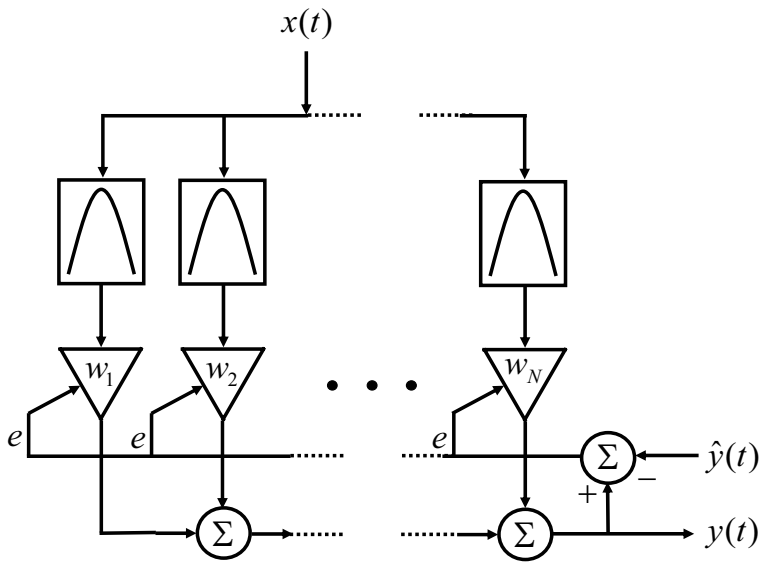

Figure 66: Adaptive channel equalization filter

minimizing channel distortion; ideally we desire that

$$H_{ch}(s)H_{af}(s) = 1 \tag{99}$$

where $H_{af} = H_{ch}^{-1}(s)$ is the adaptive filter response.

Figure 66 illustrates our proposed adaptive filter where inputs to the adaptive weights of a node are provided by a bank of bandpass filters whose transfer functions are given by

$$H_n(s) = K\frac{\tau_l s(1-\tau_f s)}{\tau_l\tau_f s^2 + (\tau_l + \alpha\tau_f)s + 1} \tag{100}$$

implemented using a simple circuit known as the capacitively-coupled current conveyor [23] which appears in Fig. 67. By tuning each of these filters to cover the frequency band of interest, we can find a weighted sum of these filters to approximate the inverse channel response as

$$H_{af}(s) = \sum_{n=0}^{N} w_n H_n(s). \tag{101}$$

The weights adapt to weight each frequency band to provide the desired overall frequency response of the filter just like an equalizer on a stereo, but instead of adjusting sound to fit the acoustics of a room, adjusting receiver to fit the radio transmission channel. The same

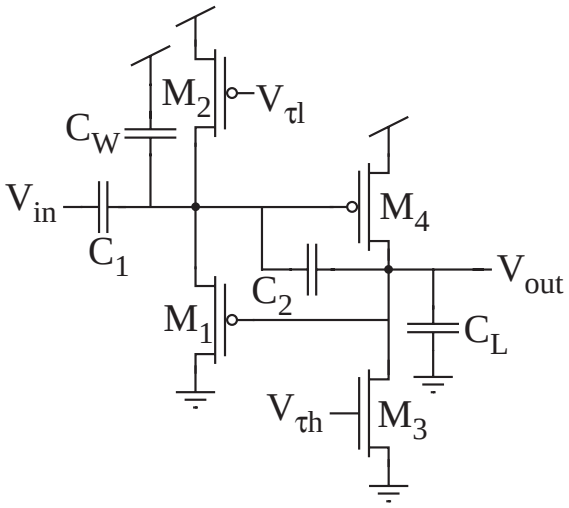

Figure 67: Circuit diagram of a capacitively coupled current conveyer (C^4). This is a bandpass filter with electronically tunable corner frequencies (controlled by $v_{\tau l}$ and $v_{\tau h}$) that can be moved independently of one another. The roll offs are first-order.

technique can be applied to system identification, which is an important problem in control theory.

6.6 Concluding Comments

In this chapter we have outlined the ideas necessary to extend the source-follower floating-gate synapse and simple nodes built from this synapse to larger-scale, multiple-input, multiple-output adaptive matrices. An array chip built for this purpose was introduced, and a custom circuit board for testing this system was described. In addition, we introduced a non-floating-gate all-transistor synapse model for simulation purposes. This circuit proves useful in its own right when faster adaptation rates are needed. Analysis of this circuit illustrates key features shared among adaptive analog circuits of this type based on both floating-gate and non-floating-gate devices. Simulation results obtained for this circuit showed close agreement with experimental data gathered from the SFFG synapse. Finally, we outlined a real-world application for which both the floating-gate and non-floating-gate synapses would be useful.

CHAPTER VII

CONCLUSIONS

Research presented in this thesis provides a substantial leap from the study of interesting device physics to fully adaptive analog networks and lays a solid foundation for future development of large-scale, compact, low-power adaptive parallel analog computation systems. This work builds on earlier investigations of floating-gate device physics in the quest to develop the smallest possible electronic synapse [26]. Previous work was inspired by earlier research which showed that floating-gate devices might not only be useful for non-volatile storage, but also useful as analog computational devices [37]. The result of these investigations led to the first useful adaptive floating-gate circuit, the auto-zeroing floating-gate amplifier (AFGA) [26]. Observation of signal-dependent adaptive behavior in the AFGA suggested this device could be useful in a learning system [26].

The investigation described here started with observation of this potential learning capability and led to the first derivation and characterization of the floating-gate pFET correlation learning rule. The learning rule was proven through experiments illustrating that the device was able to learn correlations between the amplitude and phase differences of sinusoidal voltages applied to the gate and drain terminals of the pFET [29, 31, 30]. Discovery and confirmation of the correlation learning rule led to experiments exploring the possibilities of using floating-gate learning device for adaptive systems, beginning with simple supervised experiments for uncorrelated inputs applied to a simple two-input node.

Starting with two synapses sharing the same error signal, we progressed from phase correlation experiments through correlation experiments involving harmonically related sinusoids, culminating in learning the Fourier series coefficients of a square wave [13]. These experiments conclusively demonstrated that the single transistor learning synapse could be utilitzed to implement an adaptive node with many inputs. During the course of these investigations, we recognized the need to develop circuitry to remove non-ideal terms from

the floating-gate synapse, developing circuits which eliminated harmonic distortion in the feedforward computation and error-signal energy terms in the correlation learning rule [14]. To cancel input-signal energy terms and dc offsets, we designed a differential structure. We concluded that finding a way to miminimize input-variance effects would lead to significant improvements in the correlation learning rule. All of the early two-input node experiments were restricted to uncorrelated input signals.

Extending these earlier two-input node experiments to the general case of correlated inputs required dealing with weight decay naturally exhibited by the learning rule. We introduced a source-follower floating-gate synapse as an improvement over our earlier source-degenerated floating-gate synapse in terms of relative weight decay [15]. While the ideal LMS solution does not exhibit weight decay, it is impossible to build a physical system that does not have some form of decay inherent in the system. However, weight decay does add some algorithmic benefits to the learning system, such as the ability to track non-stationary processes. We presented data from a two-input circuit and illustrated good agreement between the results and a mathematical model of the system. The new design had the added benefit of eliminating input-signal variance terms from the correlation learning rule.

A larger network of source-follower floating-gate synapses was fabricated and an FPGA-controlled testboard was designed and built. This more sophisticated system provides an excellent framework for exploring applications to multi-input, multi-node adaptive filtering applications. Adaptive channel equalization provided a practical test-case illustrating the use of these adaptive systems in solving real-world problems. The same system could easily be applied to noise and echo cancellation in communication systems and system identification tasks in optimal control problems. We envision the commercialization of these adaptive analog VLSI systems as practical products within a couple of years.

APPENDIX A

OJA'S RULE DERIVATION

A.1 Adaptive Node Computation

An adaptive node computes the inner product of an input signal vector and an adaptive weight as demonstrated by

$$y(t) = \mathbf{w}^T \mathbf{x}(t) \tag{102}$$

Because the weight, $\mathbf{w}$, is adaptive it is also a function of time, but the rate of adaptation is such that we can consider it a constant in computation.

A.1.1 General Observations on Weights

Two characteristics define a weight vector — *magnitude* and *direction*. We can thus represent the weight as $\mathbf{w} = ||\mathbf{w}||\mathbf{u}_\mathbf{w}$. Variations in weight vector magnitude merely affect the amplitude of the signal and reflect the *gain* of the system. The weight vector direction affects the variation of the weight components relative to each other, which reflects how much of each input signal shows up in the output. Hence, the direction of the weight vector affects the temporal *shape* of the output signal; (consider the effect of varying Fourier coefficients). The direction of the weight corresponds to a basis signal. Questions to consider when examining the results of learning rules are: 1. What function relates the weight magnitude to the input and learning signal statistics?; 2. What function relates the weight direction to the input and learning signal statistics? 3. What happens when the weight magnitude varies with weight direction?

A.1.2 Input Signals

For our experiments, we design the input signal by first generating a vector of orthonormal basis signals

$$E[\mathbf{s}(t)\mathbf{s}^T(t)] = \mathbf{I} \tag{103}$$

and then operating on these signals with a mixing matrix

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \tag{104}$$

The orthonormal basis signals can be harmonically related sinusoids or independent, identically distributed random variables, for example. All expected value operations are taken with respect to time

$$E[x(t)] = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t)dt \tag{105}$$

The autocorrelation matrix of the input signals is defined as

$$\mathbf{R} = E[\mathbf{x}(t)\mathbf{x}^T(t)] \tag{106}$$

We express the autocorrelation matrix in terms of the mixing matrix as

$$\begin{aligned} \mathbf{R} &= E[\mathbf{x}(t)\mathbf{x}^T(t)] \\ &= E[\mathbf{A}\mathbf{s}(t)(\mathbf{A}\mathbf{s}(t))^T] \\ &= E[\mathbf{A}\mathbf{s}(t)\mathbf{s}^T(t)\mathbf{A}^T] \\ &= \mathbf{A}\mathbf{A}^T \end{aligned} \tag{107}$$

We choose

$$\mathbf{A} = \mathbf{S}\mathbf{\Lambda}^{1/2}. \tag{108}$$

such that we can diagonalize $\mathbf{R}$ as

$$\mathbf{R} = \mathbf{S}\mathbf{\Lambda}\mathbf{S}^T \tag{109}$$

The matrix $\mathbf{S}$ has columns forming an orthonormal eigenvector basis, therefore we have $\mathbf{S}^T = \mathbf{S}^{-1}$. We represent the weight in terms of this eigenvector basis as,

$$\mathbf{w} = \mathbf{S}\hat{\mathbf{w}} \qquad \text{or} \qquad \hat{\mathbf{w}} = \mathbf{S}^T\mathbf{w} \tag{110}$$

to better study the solutions of various learning rules

$\mathbf{R}$ is a positive definite matrix; all of its eigenvalues are positive.

Extracting the weight from y

$$\begin{aligned} \mathbf{w} &= (\mathbf{A}^{-1})^T E[\mathbf{s}(t)y(t)] \\ &= \mathbf{S}\mathbf{\Lambda}^{-1/2} E[\mathbf{s}(t)y(t)] \end{aligned} \tag{111}$$

A.2 Unsupervised Learning and Oja's Rule

Almost all learning rules express a search for a weight vector that minimizes or maximizes some objective function, with or without a constraint condition [43, 50]. Hebb's rule results in a weight vector which yields the maximum output variance. Unfortunately, Hebb's rule is unstable. Oja's rule is a modification of Hebb's rule which obtains stability by constraining the weight vector to lie on the unit circle. This rule can be derived starting with the variance maximization of Hebb's rule and adding a Lagrange multiplier constraint given by

$$||\mathbf{w}||^2 = 1 \tag{112}$$

which requires the weights to lie on the unit circle. We add the effects of this constraint, scaled by a multiplier, to the objective function

$$\epsilon = \frac{1}{2}[E[y^2] + \lambda(1 - ||\mathbf{w}||^2)]; \tag{113}$$

note that this term adds zero to the overall error, but will introduce another condition to be met when we take the derivative to optimize this function.

$$\frac{\partial \epsilon}{\partial \mathbf{w}} = -\lambda \mathbf{w} + E[\mathbf{x}y] \tag{114}$$

$$-\lambda \mathbf{w} + E[\mathbf{x}y] = 0 \tag{115}$$

solve for λ $\lambda = \mathbf{w}^T \mathbf{R} \mathbf{w}$ or $\lambda = {\sigma_y}^2$

$$\tau \frac{d\mathbf{w}}{dt} = (\mathbf{x} - \mathbf{w}y)y \tag{116}$$

$$\tau \frac{d\mathbf{w}}{dt} = -{\sigma_y}^2 \mathbf{w} + \mathbf{R}\mathbf{w} \tag{117}$$

We observe that without the weight decay term, we obtain Hebb's rule which is unstable.

$$\begin{aligned} {\sigma_y}^2 &= \mathbf{w}^\mathbf{T} \mathbf{R} \mathbf{w} \\ &= (\mathbf{S}\hat{\mathbf{w}})^T (\mathbf{S}\mathbf{\Lambda}\mathbf{S}^T)(\mathbf{S}\hat{\mathbf{w}}) \\ &= \hat{\mathbf{w}}^T \mathbf{\Lambda} \hat{\mathbf{w}} \end{aligned} \tag{118}$$

$$\begin{aligned}
\tau\frac{d\hat{\mathbf{w}}}{dt} &= \mathbf{S}^T(\tau\frac{d\mathbf{w}}{dt}) \\
&= \mathbf{S}^T(\mathbf{R} - {\sigma_y}^2\mathbf{I})\mathbf{w} \\
&= \mathbf{S}^T(\mathbf{S}\mathbf{\Lambda}\mathbf{S}^T - {\sigma_y}^2\mathbf{S}\mathbf{S}^T)\mathbf{w} \\
&= \mathbf{S}^T\mathbf{S}(\mathbf{\Lambda} - {\sigma_y}^2\mathbf{I})\mathbf{S}^T\mathbf{w} \\
&= (\mathbf{\Lambda} - {\sigma_y}^2\mathbf{I})\mathbf{w} \\
&= (\mathbf{\Lambda} - \hat{\mathbf{w}}^T\mathbf{\Lambda}\hat{\mathbf{w}}\mathbf{I})\hat{\mathbf{w}}
\end{aligned} \tag{119}$$

A.3 The Least-Mean-Square Learning Rule and Weight Decay

$$\begin{aligned}
min_{\mathbf{w}}\tfrac{1}{2}E[e^2] &= min_{\mathbf{w}}\tfrac{1}{2}E[(\hat{y} - y)^2] \\
&= min_{\mathbf{w}}(\tfrac{1}{2}{\sigma_{\hat{y}}}^2 - E[\hat{y}y] + \tfrac{1}{2}{\sigma_y}^2)
\end{aligned} \tag{120}$$

$$\tau\frac{d\mathbf{w}}{dt} = -\mathbf{R}\mathbf{w} + \mathbf{p} \tag{121}$$

$$\mathbf{p} = E[\mathbf{x}\hat{y}] \tag{122}$$

This system is driven by the input, $\mathbf{p}$, which is the only term in which the target signal, $\hat{y}$, appears. Unlike Hebb's rule, this weight update converges without weight decay because $-\mathbf{R}$ is a negative definite matrix. The solution to this equation is

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{p} \tag{123}$$

$$\begin{aligned}
\hat{\mathbf{w}} &= \mathbf{S}^T\mathbf{w} \\
&= \mathbf{S}^T\mathbf{R}^{-1}\mathbf{p} \\
&= \mathbf{S}^T(\mathbf{S}\mathbf{\Lambda}\mathbf{S}^T)^{-1}\mathbf{p} \\
&= \mathbf{\Lambda}^{-1}\mathbf{S}^T\mathbf{p} \\
&= \mathbf{\Lambda}^{-1}E[\mathbf{S}^T\mathbf{x}\hat{y}] \\
&= \mathbf{\Lambda}^{-1}E[\mathbf{\Lambda}^{1/2}\mathbf{s}\hat{y}] \\
&= \mathbf{\Lambda}^{-1/2}E[\mathbf{s}\hat{y}]
\end{aligned} \tag{124}$$

A.3.1 Weight Decay

$$\tau \frac{d\mathbf{w}}{dt} = -\alpha \mathbf{w} - \mathbf{R}\mathbf{w} + \mathbf{p} \tag{125}$$

Note similarity to Oja's rule in (117). The solution to this equation is

$$\begin{aligned} \mathbf{w} &= (\mathbf{R} + \alpha \mathbf{I})^{-1} \mathbf{p} \\ &= (\mathbf{I} + \alpha \mathbf{R}^{-1})^{-1} \mathbf{R}^{-1} \mathbf{p} \end{aligned} \tag{126}$$

Diagonalizing (126)

$$\begin{aligned} \hat{\mathbf{w}} &= \mathbf{S}^T \mathbf{w} \\ &= \mathbf{S}^T (\mathbf{S}\mathbf{\Lambda}\mathbf{S}^T + \alpha \mathbf{I})^{-1} \mathbf{p} \\ &= (\mathbf{I} + \alpha \mathbf{\Lambda}^{-1})^{-1} \mathbf{\Lambda}^{-1} \mathbf{S}^T \mathbf{p} \\ &= (\mathbf{I} + \alpha \mathbf{\Lambda}^{-1})^{-1} \mathbf{\Lambda}^{-1} E[\mathbf{S}^T \mathbf{x} \hat{y}] \\ &= (\mathbf{I} + \alpha \mathbf{\Lambda}^{-1})^{-1} \mathbf{\Lambda}^{-1} E[\mathbf{\Lambda}^{1/2} \mathbf{s} \hat{y}] \\ &= (\mathbf{I} + \alpha \mathbf{\Lambda}^{-1})^{-1} E[\mathbf{\Lambda}^{-1/2} \mathbf{s} \hat{y}] \end{aligned} \tag{127}$$

All the elements of $\mathbf{\Lambda}$ are postive.

REFERENCES

[1] Farhan Adil, Guillermo Serrano, and Paul Hasler. Offset removal using floating-gate circuits for mixed-signal systems. In *SouthWest Symposium on Mixed Signal Design*, February 2003.

[2] Phillip E. Allen and Douglas R. Holberg. *CMOS Analog Circuit Design.* Oxford University Press, 2 edition, 2002.

[3] A.P. Almeida and J.E. Franca. Digitally programmable analog building blocks for the implementation of artificial neural networks. *IEEE Transactions on Neural Networks*, (2):506–514, March 1996.

[4] J. White B. Furman and A.A. Abidi. Cmos analog ic implementing the backpropagation algorithm. In *Abstracts of the First Annual INNS Meeting*, 1988, pages =.

[5] R.G. Benson and D.A. Kerns. Uv-activated conductances allow for multiple time scale learning. *IEEE Transactions on Neural Networks*, (3):434–440, 1993.

[6] Gert Cauwenberghs. Neuromorphic learning vlsi systems: A survey. In T. S. Lande, editor, *Neuromorphic Systems Engineering.* Kluwer Academic Publishers, Norwell MA, 1998.

[7] P.S. Churchland and T.J. Sejnowski. *The Computational Brain.* MIT Press, Cambridge, MA, 1992.

[8] Peter M. Clarkson. *Optimal and Adaptive Signal Processing.* CRC Press, Boca Raton, 1993.

[9] M. Cohen and A.G. Andreou. Current-mode subthreshold mos implementation of the herault-jutten autoadaptive network. *IEEE Journal of Solid State Circuits*, (5):714–727, 1992.

[10] Z. Czarnul. Novel mos resistive circuit for synthesis of fully integrated continuous-time filters. *IEEE Transactions on Circuits and Systems*, (2):277–281, 1986.

[11] P. Dayan, G. E. Hinton, R. M. Neal, and R. S. Zemel. The helmholtz machine. *Neural Computation*, pages 889–904, 1995.

[12] T. Delbr˘

ck. Bump circuits for computing similarity and disimilarity of analog voltages. In *Proc. IJCNN*, volume I, pages 475–479, Seattle, WA, July 8-12 1991.

[13] J. Dugger and P. Hasler. A floating-gate analog adaptive node. In *2000 IEEE Midwest Symposium on Circuits and Systems*, volume 3, pages 1058–1061, East Lansing, MI., August 8-11 2000.

[14] J. Dugger and P. Hasler. Improved correlation learning rule in continuously adapting floating-gate arrays using logarithmic pre-distortion of input and learning signals. In *2002 IEEE International Symposium on Circuits and Systems*, volume 2, pages II–536–II–539, Phoenix, AZ., 2002.

[15] Jeff Dugger and Paul Hasler. An analog floating-gate node for supervised learning. *IEEE Transactions on Circuits and Systems I*, Submitted for review.

[16] S.T. Dupuie and M. Ismail. High frequency cmos transconductors. In C. Toumazou, F. J. Lidgey, and D. G. Haigh, editors, *Analogue IC Design: the Current-Mode Approach*. Peter Peregrinus Ltd., 1990.

[17] D. Durfee and F.S. Shoucair. Comparison of floating gate neural network memory cells in standard vlsi CMOS technology. *IEEE Transactions on Neural Networks*, 3(2):347–353, 1992.

[18] I.A. Mack F. Kub, K.K. Moon and F.M. Long. Programmable analog vector-matrix multiplier. *Journal of Solid State Circuits*, (1):207–214, 1990.

[19] G. Cauwenberghs F.J. Pineda and R.T. Edwards. Bangs, clicks, snaps, thuds, and whacks: an architecture for acoustic transient processing. In M.I. Jordan M.C. Moser

and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, Cambridge, MA, 1997. MIT Press.

[20] O. Fujita and A Yoshihito. A floating-gate analog memory device for neural networks. *IEEE Transactions on Electron Devices*, 40(11):2029–2035, 1993.

[21] C. Neugebauer G. Cauwenberghs and A. Yariv. An adaptive cmos matrix vector multiplier for large scale analog hardware neural network applications. In *Proceedings of the International Joint Conference on Neural Networks*, pages 507–512, Seattle, 1991.

[22] R. Howard B. Howard B. Stranghn J. Denker W. Hubbard D. Tennant G. Graf, L. Jackel and D. Schwartz. Vlsi implementation of a neural network memory with several hundreds of neurons. In *AIP Conference Proceedings*, page 182, Snowbird, 1986.

[23] P. Graham, D.W.; Hasler. Capacitively-coupled current conveyer second-order section for continuous-time bandpass filtering and cochlea modeling. In *IEEE International Symposium on Circuits and Systems, 2002.*, pages 485–488.

[24] Gunhee Han and E. Sanchez-Sinencio. Cmos transconductance multipliers: a tutorial. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, (12):1550–1563, December 1998.

[25] P. Hasler. An implementation of a continuous-time trainable neural network. Master's thesis, Arizona State University, August 1991.

[26] P. Hasler. *Foundations of Learning in Analog VLSI*. PhD thesis, California Institute of Techonology, February 1997.

[27] P. Hasler and L. Akers. Circuit implementation of a trainable neural network using the generalized hebbian algorithm with supervised techniques. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1565–1568, Baltimore, 1992.

[28] P. Hasler and L.A. Akers. A continous-time synapse employing a multilevel dynamic memory. In *Proceedings of the International Joint Conference on Neural Networks*, pages I–563–I–568, Seattle, 1991.

[29] P. Hasler and J. Dugger. Correlation learning rule in floating-gate pFET synapses. In *1999 IEEE International Symposium on Circuits and Systems*, volume 5, pages 387–390, Orlando, FL., May 30 - June 2 1999.

[30] P. Hasler and J. Dugger. Correlation learning rule in floating-gate pFET synapses. *IEEE Transactions on Circuits and Systems II*, 48(1):65–73, January 2001.

[31] P. Hasler, B. Minch, J. Dugger, and C. Diorio. Adaptive circuits and synapses using pFET floating-gate devices. In G. Cauwenberghs and M. Bayoumi, editors, *Learning on Silicon*, pages 33–65. Kluwer Academic Press, Norwell MA, 1999.

[32] Paul Hasler. Continuous-time feedback in floating-gate MOS circuits. *IEEE Transactions on Circuits and Systems II*, 48(1):56–64, January 2001.

[33] Paul Hasler, Chris Diorio, and Bradley A. Minch. A four-quadrant floating-gate synapse. In *IEEE International Symposium on Circuits and Systems*, pages 29–32, Monterey, 1998.

[34] Paul Hasler, Chris Diorio, Bradley A. Minch, and Carver A. Mead. Single transistor learning synapses. In Gerald Tesauro, David S. Touretzky, and Todd K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 817–824. MIT Press, Cambridge, MA, 1995.

[35] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, 1999.

[36] John Hertz, Anders Krogh, and Richard G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[37] K. Kotani, T. Shibata, and T. Ohmi. Neuron-mos binary-logic circuits featuring dramatic reduction in transistor count and interconnections, December 1992.

[38] M. Lenzlinger and E.H. Snow. Fowler–nordheim tunneling into thermally grown si0_2. *Journal of Applied Physics*, (1):278–283, 1969.

[39] H. Castro M. Holler, S. Tam and R. Benson. An electrically trainable artificial neural network with 10240 'floating gate' synapses. In *Proceedings of the International Joint Conference on Neural Networks*, pages 191–196, Washington, D.C., 1989.

[40] C. Mead. Scaling of mos technology to submicrometer feature sizes. *Journal of VLSI Signal Processing*, pages 9–25, 1994.

[41] Carver Mead. Neuromorphic electronic systems. In *Proceedings of the IEEE*, volume 78, pages 1629–1636, Oct 1990.

[42] Carver A. Mead. *Analog VLSI and Neural Systems*. Addison-Wesley, 1989.

[43] K. D. Miller and D. J. C. MacKay.

[44] B. A. Minch, C. Diorio, P. Hasler, and C. Mead. Translinear circuits using subthreshold floating-gate MOS transistors. In *Analog Integrated Circuits and Signal Processing*, volume 9, pages 167–179, 1996.

[45] A. F. Murray. Pulse arithmatic in vlsi neural networks. *IEEE Micro*, (6):64–74, 1989.

[46] A. P. Shon, D. Hsu, and C. Diorio. Learning spike-based correlations and conditional probabilities in silicon. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[47] B. Linares-Barranco T. Serrano-Gotarredona and J.L. Huertas. A real time clustering cmos engine. In D.S. Touretzky G. Tesauro and T.K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 755–762, Cambridge, MA, 1995. MIT Press.

[48] Y. Tsividis and S. Satyanarayana. Analogue circuits for variable synapse electronic neural networks. *Electronics Letters*, (2):1313–1314, 1987.

[49] E.A. Vittoz. Dynamic analog techniques. In Y. Tsividis and P. Antognetti, editors, *Design of MOS VLSI Circuits for Telecommunications*, pages 145–170. Prentice-Hall, Englewood Cliffs, NJ, 1985.

[50] L. Wiskott and T. Sejnowski. Constrained optimization for neural map formation: A unifying framework for weight growth and normalization. *Neural Computation*, pages 671–716, 1998.

[51] Y. Xu. *Electron Transport through Thin Film Amorphous Silicon — A Tunneling Study.* PhD thesis, Stanford University, 1992.