VIEW Communicated by Joshua Alspector and John Wyatt

Analog Versus Digital: Extrapolating from Electronics to Neurobiology

Rahul Sarpeshkar
Department of Biological Computation, Bell Laboratories, Murray Hill, NJ 07974, U.S.A.

We review the pros and cons of analog and digital computation. We propose that computation that is most efficient in its use of resources is neither analog computation nor digital computation but, rather, a mixture of the two forms. For maximum efficiency, the information and information-processing resources of the hybrid form must be distributed over many wires, with an optimal signal-to-noise ratio per wire. Our results suggest that it is likely that the brain computes in a hybrid fashion and that an underappreciated and important reason for the efficiency of the human brain, which consumes only 12 W, is the hybrid and distributed nature of its architecture.

1 Introduction

We estimate that the human brain performs on the order of 3.6×10^{15} synaptic operations per second (appendix A.1 in Sarpeshkar, 1997). From measurements of cerebral blood flow and oxygen consumption, it is known that the brain consumes only 12 W (appendix A.2 in Sarpeshkar, 1997). Its efficiency of computation is thus about 3×10^{14} operations per joule. The human brain is capable of doing tremendously complex computation in real time despite the slow and noisy components in our heads and bodies.

An extremely fast microprocessor such as the DEC Alpha 21164 performs about 255×10^6 floating-point operations per second and consumes 40 W.[1] Its efficiency is thus about 6.25×10^6 operations per joule. It is incapable of solving even relatively simple behavioral tasks in real time in spite of its blazingly fast and precise transistors.

If we compare the computing efficiency of the human brain with that of a digital microprocessor, we observe that the brain is at least seven or-

[1] On the specfp92 Ear Program, which performs auditory computations similar to those in the human ear, the DEC 21164 running on an Alpha Server 8200 5/300 is 1275 times as fast as a VAX 11/780, which would run at about 0.2 MFLOPS for our computation. Thus, we estimate that it is equivalent to about $1275 \times 0.2 = 255$ MFLOPS. These numbers are for 1995.

Neural Computation **10, 1601–1638** (1998) © 1998 Massachusetts Institute of Technology

ders of magnitude more efficient.[2] Mead was the first scientist to point out the great discrepancy in the computational efficiencies of neurobiology and electronics (Mead, 1990). He also pioneered the field of neuromorphic computation—electronic computation inspired by and similar to that performed by neural systems (Mead, 1989).

How is efficient and complex computation with noisy components achieved in neurobiological systems? Mead attributed the enormous efficiency of neurobiological systems to their clever exploitation of the physics of the medium that they were built in, to their local wiring strategies, and to their enormous capabilities to adapt and learn. In this article we will focus on the trade-offs involved in using physics to do computation.

The three physical resources that a machine uses to perform its computation are time, space, and energy. Computer scientists have traditionally treated energy as a free resource and have focused mostly on time (the number of clock cycles required for the computation to terminate) and space (the amount of memory needed or the number of devices needed to perform the computation). However, energy cannot be treated as a free resource when we are interested in systems of vast complexity, such as the brain. With the current efficiencies of digital computation, it would take tens of megawatts to build a system like the brain, assuming we could do so at all. If we wanted to make this system portable as well, energy constraints would be very important indeed. Energy has clearly been an extremely important resource in natural evolution. (For an interesting discussion on energy constraints in biology and evolution, see Allman, 1990; and Aiello & Wheeler, 1995.) On a smaller scale, energy constraints are important in all portable applications, such as radio telephony, laptop computing, and hearing aids.

Biological systems typically compute constantly, rather than episodically, with the resource of time fixed by the computational requirements of the task. For example, for a sensorimotor task, we may need to respond within a few hundred milliseconds, whereas for the task of hearing a 1 K Hz tone, we will need to respond to cycle-by-cycle variations on a 1 msec time scale. Thus, throughout this article, we will assume that the bandwidth of the computational task is fixed and that the resource of time is not a degree of freedom (it will be a parameter in our equations but not a variable). The other two resources (energy and space) will be degrees of freedom; we shall

[2] It may be argued that our comparisons have not been fair since the floating-point computations that a microprocessor performs are more complex than are those that a synapse performs, and they are also more precise. However, in addition to multiplication, synaptic computations involve temporal filtering and adaptation, which are fairly complex operations in digital computation. We have also neglected several complex spatiotemporal correlations and additions that are performed in the dendrite of a neuron. Thus, for simplicity, we have chosen to compare just the efficiency of an “elementary operation” in digital computation and in neurobiology. There are so many orders of magnitude of discrepancy between neurobiology and electronics that such concerns will not alter our conclusions.

use the more natural resources of power (energy per unit time) and area (the spatial resource in a two-dimensional substrate such as nerve membrane or in VLSI) as our degrees of freedom.

Suppose that we are given two systems, *A* and *B*, that do a computation at the same bandwidth (in Hz), at the same output information rate (in bits / sec), and with the same input. *A* is more efficient than *B* if it consumes less power (and / or area) in doing this computation. In this article, we shall be interested in understanding the reasons for the efficiency of one system over another. In particular, we will study the reasons for differences in efficiency between analog and digital systems.

Electronic systems are far simpler to understand and analyze than are biological systems. So in sections 2 and 3, we begin by analyzing the differences between analog and digital electronic systems. In section 4, we use the insights gained by this analysis to outline how efficient, precise computation can be achieved by hybrid and distributed electronic architectures. In section 5 we extrapolate our ideas for electronic systems to neurobiological systems. Section 6 summarizes the article.

2 Analog Versus Digital: The Intuitive Picture

Electronic systems operate with continuous signals (CS) or discrete signals (DS), and in continuous time (CT) or discrete time (DT). Thus, there are four classes of systems: CSCT, CSDT, DSCT, DSDT (Hosticka, 1985). Figure 1 shows examples of systems, either electronic or biological, in each class. Typically, CS systems are referred to as analog, and DS systems are referred to as digital, irrespective of their representation in the time domain. In this article, we first concentrate on analog systems that are continuous in both the signal and time domains (CSCT), and on digital systems that are discrete in both the signal and time domains (DSDT). Such systems are the most common examples of analog and digital systems, respectively, and are also the most disparate from each other. Later, in section 4, we discuss why an alternation between the CSCT and DSCT domains can be advantageous over operation in the DSDT or CSCT domain alone. We shall ignore the CSDT domain in this article because its relevance to neurobiology is generally believed to be small.

Following is a comparison of CSCT and DSDT systems from a signal-processing viewpoint, emphasizing topics of importance in this article. It is by no means a comprehensive and exhaustive list of all the differences between analog and digital systems. For example, we completely omit all discussion of programmability and learning in these systems, although these issues are very important; also, we omit all discussion of temporal aliasing, which is an important source of distortion in discrete systems.

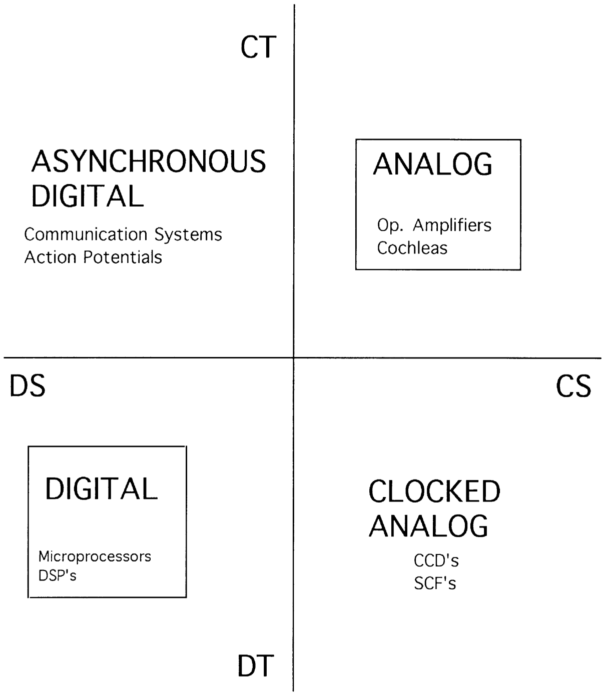

Figure 1: The four types of systems. The figure shows examples of electronic and biological systems that operate with continuous or discrete signals (CS or DS) and in continuous or discrete time (CT or DT). Analog systems that are continuous in both the signal and time domains (CSCT) and digital systems that are discrete in both the signal and time domains (DSDT) have been boxed in the figure. SCF stands for switched capacitor filter; CCD stands for charge coupled device.

	ANALOG	DIGITAL
1.	Compute with continuous values of physical variables in some range, typically voltages between the lower and upper power-supply voltages.	Compute with discrete values of physical variables, typically the lower and upper power supply voltages, denoted by 0 and 1, respectively.

2.	Primitives of computation arise from the physics of the computing devices: physical relations of transistors, capacitors, resistors, floating-gate devices, Kirchoff's current and voltage laws and so forth. The use of these primitives is an art form and does not lend itself easily to automation. The amount of computation squeezed out of a single transistor is high.	Primitives of computation arise from the mathematics of boolean logic: logical relations like AND, OR, NOT, NAND, and XOR. The use of these primitives is a science and lends itself easily to automation. The transistor is used as a switch, and the amount of computation squeezed out of a single transistor is low.
3.	One wire represents many bits of information at a given time.	One wire represents 1 bit of information at a given time.
4.	Computation is offset prone since it is sensitive to mismatches in the parameters of the physical devices. The degradation in performance is graceful.	Computation is not offset prone since it is insensitive to mismatches in the parameters of the physical devices. However, a single bit error can result in catastrophic failure.
5.	Noise is due to thermal fluctuations in physical devices.	Noise is due to round-off error.
6.	Signal is not restored at each stage of the computation.	Signal is restored to 1 or 0 at each stage of the computation.
7.	In a cascade of analog stages, noise starts to accumulate. Thus, complex systems with many stages are difficult to build.	Round-off error does not accumulate significantly for many computations. Thus, complex systems with many stages are easy to build.

2.1 Physicality: Advantage Analog. Items 1 through 3 show that analog computation can be far more efficient than digital computation because of analog computation's repertoire of rich primitives. For example, addition of two parallel 8-bit numbers takes one wire in analog circuits (using Kirchoff's current law), whereas it takes about 240 transistors in static CMOS digital circuits. The latter number is for a cascade of 8 full adders. Similarly an 8-bit multiplication of two currents in analog computation takes 4 to 8 transistors, whereas a parallel 8-bit multiply in digital computation takes approximately 3000 transistors. Although other digital implementations could make the comparisons seem less stark, the point here is simply that exploiting physics to do computation can be powerful. The advantage of an analog machine over a digital machine is especially great when there is a straightforward mapping between the operations needed in the computation and the primitives of the technology. For large-scale systems, as in the implementation of silicon cochleas (Sarpeshkar, Lyon, & Mead, 1998), depending on the nature of the digital implementation, the advantage can range from a factor of 300 to 10^5 in power consumption.

Because the number of devices required to perform a computation is greater in digital systems, there is more wiring and communication overhead. The presence of more devices and more communication overhead causes digital circuits to have typically higher area consumption than that of analog circuits. The switching energy dissipation due to the large number of devices and the communication overhead also causes the power consumption to be higher in digital circuits. If the number of devices switching per clock cycle is N, the clock frequency is f, the average load capacitance that a device has to drive is C, and the power supply voltage is V_{DD}, then the power consumption P_D of digital circuits is given by the simple formula (Rabaey, 1996),

$$P_D = NfCV_{DD}{}^2. \tag{2.1}$$

Unlike digital CMOS circuits, whose power dissipation occurs only during switching and is entirely dynamic, many analog circuits have standby or static power dissipation and little or no dynamic power dissipation.[3] Thus their power dissipation is given by the simple formula,

$$P_A = NV_{DD}I, \tag{2.2}$$

where N is the number of computational stages, V_{DD} is the power supply voltage, and I is the average bias current flowing through each computational stage.

We can make digital computation more power efficient by using architectures that operate on a slow-and-parallel paradigm. Such architectures conserve power by allowing the use of lower–clock-frequency and lower–supply-voltage operation, although they require increased area consumption (Chandrakasan, Sheng, & Brodersen, 1992). Bit-serial digital implementations are area efficient because they use time multiplexing to perform several computations on the same circuit (Denyer & Renshaw, 1985). The rapid evolution of digital technology has shrunk the efficiency gap between analog and digital computation. However, the inefficiency of ignoring the physical computational primitives inherent in the technology and the inefficiency of encoding only 1 bit per wire is always present in digital computation. Consequently, analog computation still retains its advantage.

[3] Of course, class AB analog systems have dynamic power dissipation, but we are focusing on only general trends.

2.2 Noise and Offset: Advantage Digital. Although the use of physics made analog systems much more efficient than digital systems, items 4 through 7 reveal that the very same physics causes analog systems to be much more sensitive to noise and offset than digital systems. The use of continous signal variables precludes analog systems from having any discrete attractor state to which they can be restored. Thus, for a sufficiently complex computation, the noise accumulation in analog systems becomes severe, not enough precision can be maintained at the output of the system, and analog systems clearly emerge as the losers.

Adaptation can help to compensate for offset in analog systems. However, performance is still ultimately limited by residual offsets due to the finite loop gains of the compensating circuits and by the offsets introduced by the compensating circuits themselves. If the compensation of offset is done periodically or continuously, such that the offset remains bounded throughout the computation, then the problem of offsets may be alleviated in analog systems. However, offset compensation is achieved at the cost of increased complexity, area, or power consumption; also, care must be taken to ensure that the feedback loops do not cause unwanted dynamics due to interactions with the rest of the analog system.

We can attenuate noise if we are willing to spend a large amount of power (and / or area) resources. However, as we shall show in section 3, by this point a digital solution would be more efficient than an analog solution. Parasitic capacitances and resistances in physical devices set a lower bound on the achievable noise floor in practical analog systems.

3 Analog Versus Digital: The Quantitative Picture

In this section we quantify the intuitive picture of section 2. We need to have an understanding of what causes noise in the devices with which we compute and how the noise accumulation from the various devices in a system degrades the output signal-to-noise ratio of an analog system.

3.1 Noise in MOS Transistors. We usually treat current as though it is the flow of a continuous fluid, although it is the flow of discrete charged electrons. Due to thermal fluctuations, these electrons have random, diffusive motions that are uncoordinated with one another. These incoherent motions give rise to shot-noise currents and cause white noise in the device. The noise is called white because its power spectrum is flat. Intuitively, by simple $\sqrt{N}$ law-of-large-numbers arguments, we might expect that shot noise would be less important at larger current levels because we average over the motions of more electrons per unit time. This intuition is indeed borne out. (For further details of noise in transistors, see Sarpeshkar, Delbrück, & Mead, 1993, and Sarpeshkar, 1997.) White noise is fundamental and is present in all physical devices at room temperature. The input-referred white noise of an MOS transistor is given by

$$v_n^2 = \frac{K_w(p)}{I^p}\Delta f, \tag{3.1}$$

where $p = 1.0$ in the subthreshold region of operation of the MOS transistor, and $p = 0.5$ in the above-threshold region of operation of the MOS transistor; I is the DC current through the transistor; v_n^2 is the expected value of the square of a band-limited white noise voltage signal, applied between the transistor gate and source; $\Delta f = f_h - f_l$ is the bandwidth of operation, with f_h and f_l being, respectively, the highest and lowest frequencies of operation; the technology-dependent parameter $K_w(p)$ increases with temperature and with thick gate oxides, and is given by

$$K_w(1.0) = \frac{4kTU_T}{2\kappa^2} \tag{3.2}$$

in the subthreshold regime, and by

$$K_w(0.5) = \frac{4kT(2/3)}{\sqrt{\left(2\mu C_{ox}\frac{W}{L}\right)}} \tag{3.3}$$

in the above-threshold regime. The parameter κ is the subthreshold exponential coefficient; kT is a unit of thermal energy; $U_T = kT/q$ is the thermal voltage where q is the charge on the electron; μ is the mobility of the electron; C_{ox} is the oxide capacitance per unit area; and W and L are the width and length of the transistor, respectively. Note that K_w is independent of transistor geometry in the subthreshold regime, but is dependent on the transistor geometry in the above-threshold regime. The parameter $K_w(p)$ is an important parameter of MOS technology.

Another kind of noise in the transistor is called $1/f$ noise because its power spectrum varies inversely with the frequency. It is widely believed that this form of noise arises from electrons in the channel going into and out of surface states, and into and out of impurities or defect traps in the gate oxide of the transistor. It is known that the mean square $1/f$ noise voltage at the gate input of the transistor v_{nf}^2 scales inversely with the area of the transistor $A = WL$. The noise is approximately independent of the current flowing through the transistor,

$$\begin{aligned} v_{nf}^2 &= \frac{K_f}{A}\int_{f_l}^{f_h}\frac{df}{f}, \\ &= \frac{K_f}{A}\ln\left(\frac{f_h}{f_l}\right). \end{aligned} \tag{3.4}$$

The parameter K_f is given by

$$K_f = \frac{B}{C_{ox}}, \tag{3.5}$$

where B is a measure of the number of surface states, impurities, or defects in the gate oxide of the transistor.

The electronic fluctuations just described dynamically modulate the surface potential and thus the threshold voltage of the transistor. Hence, $1/f$ noise can be viewed as noise due to a dynamically varying threshold voltage. Since the current in a transistor depends on the difference between the gate voltage and its threshold voltage, independent of where the transistor is operating, the input-referred $1/f$ noise is independent of current. The larger the area of the transistor, the greater the oxide capacitance of the transistor, and the smaller the effect of any one fluctuating electronic charge on the transistor's threshold voltage. However, since the trap and defect densities are approximately constant, the larger the area of the transistor, the greater the number of fluctuating charges. The increased capacitance effect reduces the noise power like $1/A^2$, and the increased total charge effect increases the noise power like A, such that the input-referred noise scales like $1/A$.

The parameter B also determines the magnitude of typical offsets in MOS technology. Offsets between transistors are mainly due to mismatches in threshold voltage caused by charges in impurities, surface states, defect traps, and so on. By applying the reasoning of the previous paragraph, we can show that offsets scale inversely with the area of the transistor as well. Thus, the R.H.S. of equation 3.4, which models the magnitude of $1/f$ noise in MOS technology, also models the magnitude of the typical offsets in this technology. Actually, the total $1/f$ noise would be affected by f_l and f_h, but the offsets would not be. So to model offsets, we should add another term proportional to K_f but independent of f_l and f_h. However, this added complication neither affects nor adds to our conclusions. Thus, we model $1/f$ noise and offsets with one term. For similar reasons, we do not discuss other less important sources of offset such as geometric mismatches which scale like $1/L$ or $1/W$ or some function of L and W.

Adaptation may help lower the effective value of K_f in a circuit, but it cannot make it zero. Area is expended in the adaptation circuitry and in improving the residual offsets after adaptation.

Thus, we observe that the noise and offset (or $1/f$ noise) in an MOS transistor decrease with an expenditure of power and area resources in the transistor, respectively. The total input-referred noise is given by

$$v_n^2 = \frac{K_w(p)}{I^p}\Delta f + \frac{K_f}{A}\ln(f_h/f_l). \tag{3.6}$$

We call such an equation a *noise resource equation* for the transistor. In any technology, each device will have its own noise resource equation that illustrates how the noise in the device decreases with an increase in the resources consumed by the device. In this case, we consume the resource of power (current) to reduce thermal noise, and the resource of area to reduce $1/f$ noise (or offset). In general, in any technology, by similar law-of-large-numbers arguments, the thermal noise reduces with power consumption, and the offsets reduce with the increased consumption of some spatial resource like length, area, or volume.

The resource of time is implicitly represented in equation 3.6 as the Δf and $\ln(f_h/f_l)$ variables. A small bandwidth (f_l and f_h are near each other) implies that we have a lot of time at our disposal, which we may trade off for lower noise or lower power / area consumption. Thus, equation 3.14 also captures trade-offs between maintaining low noise with few resources and maintaining bandwidth. Averaging is an example of a technique that reduces the bandwidth of a computation while lowering noise.

3.2 Noise in Analog Systems. Figure 2 shows a cascade of M analog computational stages with an input V_{in} and output V_{out}. Each stage i has a certain number of devices n_i, has a gain g_i, consumes a current I_i, consumes an area A_i, and adds a certain amount of noise v_{ni}. The cascade is representative of many analog computations that involve distributed gain amplification. In neurobiology, distributed gain amplification occurs in the dendrites of neurons or in the traveling wave amplifier architecture of the cochlea.

The complexity of the computation sets a lower bound on the number of devices in each stage n_i and on the total number of stages M. The ingenuity of the analog designer determines how close to the bound a realization of this system is. Depending on the details of the computation, the bound may be on $M \times \Sigma n_i$, on all the n_i, on n_1 and $M \times \Sigma n_i$, and so on. We assume that the power supply voltage V_{DD} is fixed and is equal to or slightly greater than the linear voltage range of the system, otherwise power is unnecessarily wasted, with no increase in output signal-to-noise ratio. So, we choose not to operate the system in this nonoptimal situation. We also make two simplifying assumptions. We assume that the current I_i and area A_i of stage i are divided equally among all the n_i devices in the stage. We also assume that each of the n_i devices contributes equally to the noise of the stage v_{ni} and is amplified by the full gain g_i of that stage. In practice, the circuit topology of a stage determines the amount of current through a device. The circuit topology also determines the noise contribution of that device to the noise of the stage. In spite of our simplifying assumptions, our model captures the general trend of the noise in each stage to increase with increasing n_i, and the noise at the output of the cascade to increase with increasing M.

The total mean square noise at the output of the cascade, v_{no}^2, is made up of noise from each of the computational stages. The noise at the first stage is amplified by the cascaded gain of all the stages, whereas noise at the output of the ith stage is amplified by the cascaded gain of all stages from i to M.

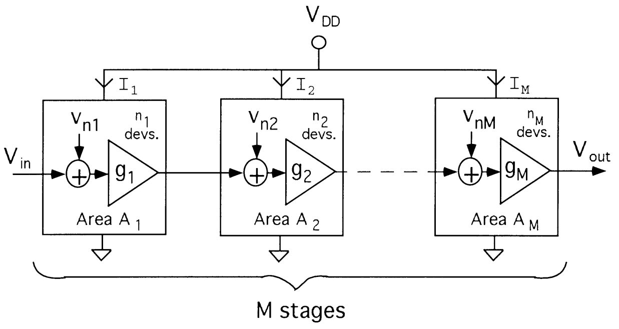

Figure 2: Noise accumulation in an analog system. The figure shows a cascade of M analog computational stages, each of which contributes some noise to the output. The common power supply is represented by V_{DD}. If we want to minimize noise at the final output, V_{out}, subject to fixed constraints on total current consumption (sum of the I_i's) and total area consumption (sum of the A_i's), then equations 3.7, 3.11, and 3.13 show that the complex stages (stages with large values of n_i) and the early stages (stages with large amounts of accumulated gain) should get most of the system's resources of current and area.

Therefore, the early computational stages typically contribute more noise than do the later stages (Haus & Adler, 1959). We define the noise gain from stage i to the output as G_i, with

$$G_i = \prod_{k=i}^{k=M} g_k. \tag{3.7}$$

Then, from equation 3.6, the assumptions of the previous paragraph, and Figure 2, we have the total noise at the output given by[4]

$$\begin{aligned} v_{no}^2 &= \sum_{i=1}^{i=M} v_{ni}^2 G_i^2 \\ &= \sum_{i=1}^{i=M} \left(n_i \frac{K_w(p)}{(I_i/n_i)^p} \Delta f + n_i \frac{K_f}{(A_i/n_i)} \ln(f_h/f_l) \right) G_i^2. \end{aligned} \tag{3.8}$$

The nature of the computational task determines the requirements on f_l and

[4] For simplicity, we assume that p is the same across all stages. In a practical situation, the first few amplifiers may be operated above threshold ($p = 0.5$) to reduce noise, and the last few amplifiers may be operated in subthreshold ($p = 1.0$).

f_h. The bandwidth of the system, $\Delta f = f_h - f_l$, is the overall bandwidth of the system at the output. Any individual computational stage may have a bandwidth higher than this, but that is not the bandwidth that is relevant for noise calculations at the final output.

Suppose that we have a total amount of current I_T, or equivalently power $P_T = V_{DD} I_T$, at our disposal; suppose that we also have a total amount of area A_T—that is,

$$\sum_{i=1}^{i=M} I_i = I_T,$$
$$\sum_{i=1}^{i=M} A_i = A_T. \tag{3.9}$$

We now ask how we should distribute our current and area resources among the various stages to minimize the output noise given by equation 3.8. The answer to this question is a simple exercise in multivariable minimization through a Lagrange-multiplier technique. We find that the currents I_i and areas A_i should be distributed such that

$$I_i = \frac{w_i I_T}{\sum w_i}, \tag{3.10}$$
$$w_i = G_i^{2/(1+p)} n_i, \tag{3.11}$$
$$A_i = \frac{z_i A_T}{\sum z_i}, \tag{3.12}$$
$$z_i = G_i n_i. \tag{3.13}$$

With the optimal allocation of resources, the total noise at the output is given by

$$v_{no}^2 = \frac{\left(\sum_{i=1}^{i=M} w_i\right)^{1+p} K_w(p) \Delta f}{I_T^p} + \frac{\left(\sum_{i=1}^{i=M} z_i\right)^2 K_f \ln(f_h/f_l)}{A_T}. \tag{3.14}$$

This equation is the noise resource equation for our system. We find that the noise resource equation for the device equation 3.6 and the noise resource equation for the system equation 3.14 are very similar. The noise resource equation for the device modeled the technology with the p, $K_w(p)$, and K_f parameters. The noise resource equation for the system added the effects of the complexity of the task and the ingenuity of the analog designer in the $\sum w_i$ and $\sum z_i$ terms. Both equations reveal that power and area resources lower thermal noise and 1 / f noise (or offset), respectively. (Further subtleties of noise in analog systems are discussed in Sarpeshkar, 1997.)

To first order, equation 3.14 quantitatively captures all the intuitive ideas about noise and offset that we expressed in items 4–7 of our analog-versus-digital list. Equation 3.14 reveals how noise accumulates in analog systems; if M and / or n_i are large, as would be the case for a complex computation, then the output noise can be large indeed. Equations 3.11, 3.13, and 3.14 show that if noise is to be minimized, more resources should be distributed to the parts of a system that affect all other parts of it (the initial stages) and to those parts of it that are complex (high n_i). Above threshold, the weighting of power resources toward the early stages is more severe than is that for subthreshold ($G_i^{4/3}$ versus G_i).

It is convenient to rewrite equation 3.14 as

$$v_{no}^2 = \frac{C_w}{{P_T}^p} + \frac{C_f}{A_T}, \tag{3.15}$$

where $P_T = V_{DD} I_T$. The parameter C_w is simply the numerator of the first term of equation 3.14 multiplied by V_{DD}^p, and the parameter C_f is the numerator of the second term of equation 3.14.

3.3 The Costs of Analog Precision. In an analog system, the maximum possible amplitude of an output sinusoidal signal Y is $V_{DD}/2$. The power of this signal is $V_{DD}^2/8$. For such a signal, the maximum possible signal-to-noise ratio is given by

$$S_N = \frac{V_{DD}^2}{8 v_{no}^2}, \tag{3.16}$$

where v_{no}^2 is the noise power at the output. The parameter S_N is important because the information $H(Y)$ that we can observe at the output of our system is a monotonically increasing function of S_N. The larger the value of S_N, the more finely can we distinguish among states at the output, and the greater is the output precision. The exact form of the function depends on the amplitude distribution of the output signal and the output noise. For many practical situations, $H(Y) \approx (\log_2 (1 + S_N))/2$ is a good approximation to the number of bits of information present at the output; this formula is exact if the amplitude distributions of the signal and noise are gaussian. The information at the output is an upper bound on the mutual information between the function of the input implemented by the computation and the output (Cover & Thomas, 1991).

By using the expression for system-level noise from equation 3.15 in equation 3.16, solving for P_T at constant S_N and A_T, and solving for A_T at constant S_N and P_T, we get,

$$P_T = \left(\frac{C_w S_N}{V_{DD}^2/8 - \left(C_f/A_T\right) S_N} \right)^{\frac{1}{p}}, \tag{3.17}$$

$$A_T = \left(\frac{C_f S_N}{V_{DD}^2/8 - \left(C_w/P_T{}^p\right) S_N} \right). \tag{3.18}$$

We refer to these equations as the *resource precision* equations for analog computation; they tell us how the resource utilization is a function of S_N, the variable that determines the output precision. For small values of S_N, the denominator is constant in both expressions and $P_T \propto S_N{}^{1/p}$, while $A_T \propto S_N$. Since p = 1.0 in the subthreshold regime and p = 0.5 in the above-threshold regime, the scaling laws of power versus S_N are $P_T \propto S_N$ in the subthreshold regime and $P_T \propto S_N{}^2$ in above-threshold regime. The scaling laws for area, $A \approx S_N$, are similar in both regimes. The power cost P_T diverges when S_N is limited by $1/f$ noise (or offset); we must spend area in this situation to reduce $1/f$ noise (or offset). Similarly, the area cost A_T diverges when S_N is limited by thermal noise; we must spend power in this situation to reduce the thermal noise. Actually, these conclusions of divergence are true only for the subthreshold regime, where we cannot trade the power and area resources of a transistor to obtain a certain value of S_N. Sarpeshkar (1997) shows how to trade between power and area in the above-threshold regime.

3.4 The Costs of Digital Precision. In many digital systems, the power and area costs are proportional to the number of bits b used in the computation. In such cases, a 12-bit computation consumes one-half as much area and one-half as much power as does a 24-bit computation if all parameters—such as clock frequency f, average switching capacitance C, and power supply voltage—remain fixed. If we do allow the clock frequency and power supply voltage to scale with the number of bits, as in a bit-serial implementation, then the power costs scale as a polynomial function of the number of bits. Some computations like multiplication have power and area costs that scale like the square of the number of bits. In general, most tractable computations scale as a polynomial function of the number of bits. For simplicity, we assume that the power and area costs are proportional to the number of bits. It is straightforward to extend the arguments that follow to the polynomial-scaling case, although a quantitative solution may not be possible for any general polynomial. Thus, the resource precision equations for digital computation are given by

$$P_T = L_p \log_2 (1 + S_N), \tag{3.19}$$

$$A_T = L_a \log_2 (1 + S_N), \tag{3.20}$$

where b is defined from the relationship $b \approx (\log_2(1+S_N))/2$. The parameter L_a would scale like NWL where W and L are the widths and lengths of a small transistor, and N represents the complexity of the task and the ingenuity of the digital designer. The parameter L_p would scale like $NfCV_{DD}{}^2$.

3.5 Precision Costs: Analog Versus Digital. Figure 3 shows power and area resource-precision curves for subthreshold analog computation (equations 3.17 and 3.18 with $p = 1$) and for digital computation (equations 3.19 and 3.20). We see that analog computation is cheaper than digital computation at low values of S_N and more expensive than digital computation at high values of S_N. Note also the divergence in power and area costs when S_N is limited by $1/f$ noise (area) and thermal noise (power), respectively. The exact location of the crossover point will depend on the task, technology, and ingenuity of the analog and digital designers. We have chosen values for C_w, C_f, L_p, and L_a such that the crossover happens near 10 bits (60 dB in S_N). For many common computations in today's CMOS technology, the crossover happens near 8 bits.

Curves such as the ones in Figure 3 were first proposed for comparisons of delay operations in a seminal paper (Hosticka, 1985). Recently, there has been additional work on comparing analog and digital systems for delay operations (Furth & Andreou, 1996). Vittoz (1990) compared filtering operations in analog versus digital systems, and Kearns (1993) compared analog and digital systems for their performance on the tasks of comparing two N-bit numbers, and also for the construction of delay lines. To our knowledge, the comparison presented in this article is the first to generalize the prior results to a broad class of analog and digital systems and to include the effects of $1/f$ noise and offset along with the effects of thermal noise.

3.6 Caveats. A / Ds and D / As are analog systems, and the costs of operating these systems at high precision (high S_N) are high. In a digital system with analog inputs and outputs, the precision costs of the A / D and D / A are paid once at the front end and once at the back end, respectively. The cost of the high-precision processing between the front end and back end is determined by the digital system in between. Thus, the total cost of the overall system is made up of an analog part for the A / D and D / A, and a digital part for the rest of the processing. Our comparisons between analog and digital computation ignored the additional A / D and D / A costs of a digital system. In a sufficiently complex computation, the A / Ds and D / As represent a small fraction of the total cost of the computation. In an analog system doing the same high-precision processing, the high-precision analog costs are paid throughout all parts of the system, rather than only at the front and back ends; that is why, for a sufficiently complex task, a digital system with an A / D and D / A would still be more efficient than an analog system.

In practice there is a minimum area or power consumption for both technologies that is independent of the value of S_N—for example, the minimum feature size of a technology determines the minimum possible area that may be expended. Thus, both analog and digital curves flatten out to constant values at low S_N. We have ignored such overhead costs in our simple analysis.

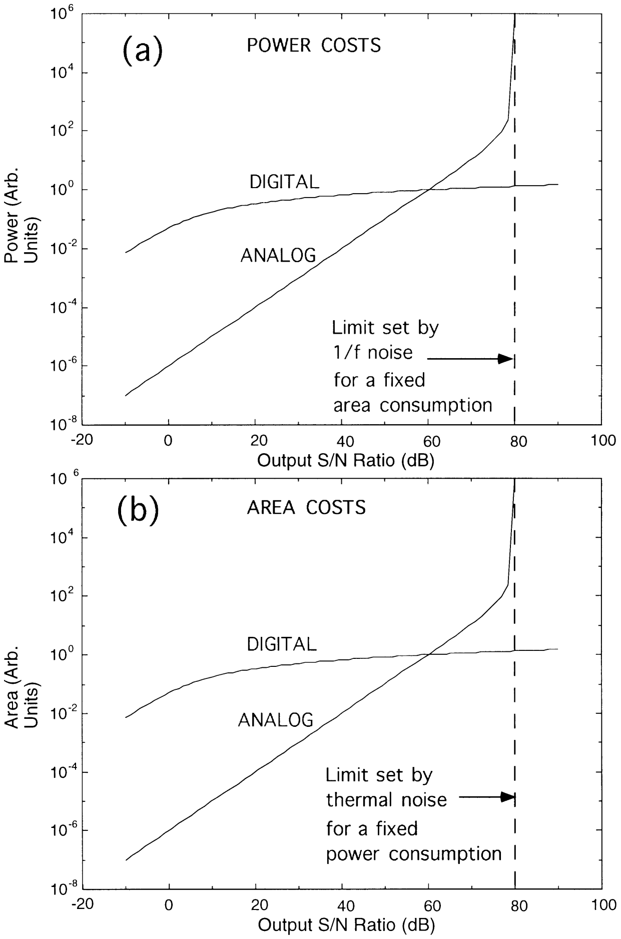

Figure 3: Resource precision curves—analog versus digital. Plots of the resource precision equations for analog computation (equations 3.17 and 3.18) and digital computation (equations 3.19 and 3.20) for subthreshold technology ($p = 1$). The plots show how the resource utilization—power in (a) and area in (b)—is a function of S_N, the output signal-to-noise ratio (a measure of precision).

3.7 Summary of the Analog-Versus-Digital Analysis. Before we begin our discussion of hybrid systems in section 4, it is worth recapitulating the lessons learned from our analysis. Physical primitives are more efficient at computing than are logical primitives as long as we do not attempt to compute with low noise on one wire. Thus, the analog constants C_w and C_f and the digital constants L_p and L_a are such that the analog curves lie below the digital curves at low S_N. At high S_N, however, the multiwire representation of information by digital systems divides the information processing into independent bit parts that many simple processing stages can collectively handle more efficiently than can one precise single-wire analog processing stage. This intuition is mathematically expressed by a logarithmic scaling of digital computation with S_N, and a power law–like scaling of analog computation with S_N. Furthermore, the lack of signal restoration in analog systems causes the noise accumulation for complex analog systems to be much more severe than that for complex digital systems. Thus, we have large values of C_w and C_f for complex analog computations (large M, w_i, or z_i in equation 3.14), whereas L_p and L_a remain of reasonable size for the equivalent complex digital computation.

4 The Best of Both Worlds

It is attractive to combine the best of both computing paradigms to make a hybrid paradigm that is better than either one. In this section, we suggest a framework for such a paradigm. In section 4.1 we show that analog computation that distributes its precision and processing resources over many wires is maximally efficient at a certain signal-to-noise ratio per wire. In section 4.2, we propose a hybrid architecture that combines the advantages of discrete-signal restoration with the advantages of continuous-signal continuous-time analog computation. In section 4.3 we describe a computing architecture that illustrates the simultaneous workings of distributed and hybrid computation.

4.1 Distributed Analog Computation. Figure 4a shows an example that illustrates the idea behind distributed analog computation. Instead of the usual analog paradigm that represents 8 bits of information on one wire, or the usual digital paradigm that represents 8 bits of information on 8 wires, in distributed analog computation, we represent 8 bits of information on two wires that carry analog signals; instead of one analog processor maintaining 8 bits of precision on its output wire, we now have two processors that interact with each other and maintain 4 bits of precision on their respective output wires. The analog signals each have a signal-to-noise ratio of 24 dB in order to encode 4 bits of information. For example, we could encode the four most significant bits of a digital number as an analog signal on one

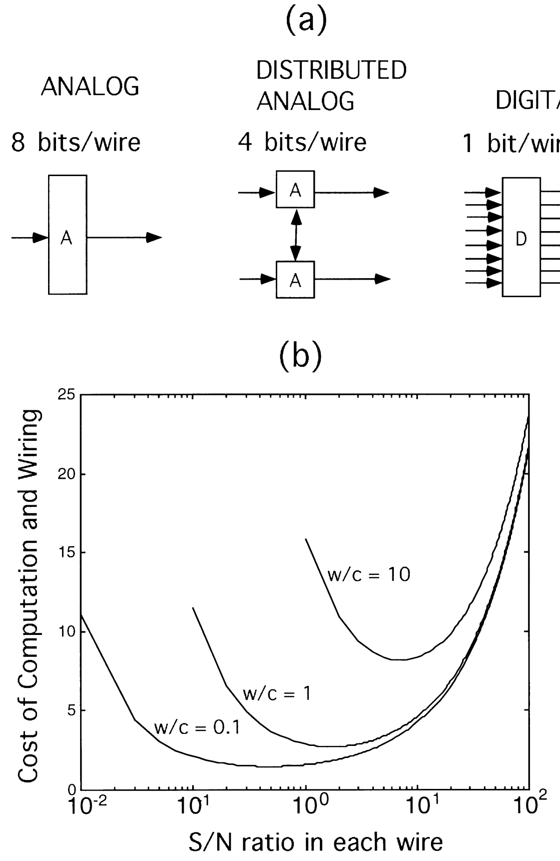

Figure 4: Distributed analog computation. (a) The idea behind distributed analog computation is illustrated by contrasting it with purely analog and purely digital computation. In distributed analog computation, analog processors interact with one another and maintain only a moderate amount of precision on their respective output wires. (b) Plots of the total cost of computation and communication as a function of S_N in each wire, for $c = 1$, and for various w/c ratios in equation 4.1 are shown.

wire by doing a 4-bit D / A operation on the four most significant bits of the digital number. Similarly, we could encode the four least significant bits of the number as an analog signal on another wire.

If the original signal was an analog signal present on a single wire, then an 8-bit A / D encoder must first convert the single-wire analog representation into a digital number. The precision of this front-end A / D operation will be at a resolution of 8 bits. However, once we have a distributed representation (2 wires × 4 bits), all subsequent analog operations may be done at a resolution of 4 bits. As in digital computation, where overflows in one channel are handled via carry propagation to an adjacent channel, the analog processors must interact with each other appropriately to preserve their distributed representation. The interaction between analog processors necessarily involves interaction between their signal-restoration circuitry as well (signal-restoration circuitry is described in section 4.2).

Because each analog processor operates at a low precision, its power consumption and area consumption requirements are low. We are interested in knowing whether the total costs in power consumption and area consumption are lower for two 4-bit processors than for one 8-bit processor. We therefore ask the following question: Suppose we want to output N bits of information by outputting b bits of information from N/b analog processors on N/b wires. What is the optimal number of bits b on each wire such that the total power or area consumption of all circuitry is minimized?

To answer the question, we will have to take the costs of wiring (communication) and computation into account. Wires cost area and add capacitance. In order to keep the bandwidth of the system constant as capacitance is added, the power consumption in the system rises. The wiring costs for area increase in linear proportion to the number of wires.[5] If bandwidth is to be maintained, the power consumption must rise in linear proportion to the total capacitance in the analog processor. Thus, the power costs of wiring also increase in linear proportion to the number of wires. In neurobiological systems, the power costs of wiring include the costs of active restoring circuitry in axons as well. Thus, wiring costs are a function of the technology.

From equations 3.17 and 3.18, for relatively small S_N where analog computation is more effective than digital computation, the power consumption and area consumption are power law functions of S_N in the subthreshold and above-threshold regimes. Thus, the analog cost function for computation per processor is well described by $c{S_N}^l$, where $l = 2$ for above-threshold power consumption, and $l = 1$ in all other cases of interest; here, c is a computation cost constant that accounts for all computation costs at each channel, including those necessary for interactions with adjacent channels, and the cost of signal restoration circuitry in the channel. We will discuss only the

[5] The linear proportionality of area cost with the number of wires accounts for only the area occupied by the wires themselves. In practice, area costs for wiring will involve the area between wires and the area between computational elements as well. Such considerations cause the area cost function to be supralinear in the number of wires. For simplicity, we assume a linear function as the supralinear case will not alter the basic nature of our conclusions.

case for $l = 1$ since the $l = 2$ case follows by straightforward extension. The cost function for wiring is given by a constant cost of w per wire. The number of bits per wire $b = \left(\log_2(1 + S_N)\right)/2$. Thus the total cost function for computation and communication is given by

$$\begin{aligned} \text{Cost} &= (cS_N + w)\left(\frac{N}{b}\right) \\ &= (cS_N + w)\left(\frac{N}{0.5\left(\log_2(1 + S_N)\right)}\right). \end{aligned} \tag{4.1}$$

Figure 4b shows plots of the total cost of computation and communication as a function of S_N in each wire, for $c = 1$, and for various w/c ratios. We see that when wiring is expensive ($w/c = 10$), the optimal signal-to-noise ratio is high, b is high, and we have few wires. When wiring is cheap ($w/c = 0.1$), the optimal signal-to-noise ratio is low, b is low, and we have many wires.

By simple calculus, we can show that the optimal S_N occurs when

$$\ln(1 + S_N) = \left(\frac{S_N + w/c}{1 + S_N}\right). \tag{4.2}$$

The optimal value S_N^o has the following limiting solutions:

$$S_N^o = \sqrt{w/c} \text{ if } w/c \ll 1, \tag{4.3}$$

$$S_N^o \ln S_N^o = w/c \text{ if } w/c \gg 1. \tag{4.4}$$

At the optimal value, the total cost of computation and communication is $2Nc \ln 2(1 + S_N)$. For the case where $w/c \ll 1$, the cost is $2Nc \ln 2$. The cost of outputting all N bits from one single analog processor is $c2^N$. Thus, if N is sufficiently big, $2Nc\ln 2 \ll c2^N$. Therefore, if the amount of output information is large, it is better to distribute the information and information processing on many wires.

4.1.1 Caveats. In general, there may be overlap in the information distributed among the channels; for example, one wire may encode the six least significant bits of an 8-bit digital number, and the other wire may encode the six most significant bits of the 8-bit number. In the latter case, we have a redundant and correlated representation of amplitude information between the two wires. We do not analyze such cases here for they are technically harder and do not illustrate the point any better.

In our analysis we have ignored the front-end costs of distributing the information from a single wire onto many wires. As we described in section 3.6, this operation is analogous to an A/D encoding cost that we pay once at the front end. For a sufficiently complex computation where we do a lot of distributed computation, this cost is negligible. Similarly, if we must eventually collapse distributed information back onto a single wire (e.g., at the output end of the system), then we will have to pay a high-precision decoding cost, as in an output D/A.

If the encoding and decoding costs are a significant part of the computation, then we have another trade-off in having our representation be highly distributed. An excessively distributed representation may require very complex encoding and decoding operations (such as A/Ds and D/As) that grow in an exponential fashion with the number of wires. The optimization of resources must then include the costs of encoding and decoding in addition to those of computation and communication.

4.2 Hybrid Computation. Noise always accumulates in a cascade of analog processing stages. If a computation is sufficiently complex, then at some point, an analog system simply cannot maintain enough precision at its output to do anything useful. Even if we require the system to maintain only 1 bit at its output, it will be unable to do so. We now show how to use a building block called the A/D/A, and an architecture that uses A/D/As for solving the noise accumulation problem in analog systems. The A/D/A is an A/D converter that is immediately followed by a D/A converter. However, its most efficient circuit implementation does not involve explicit implementation of an A/D converter and a D/A converter. The A/D/A has been proposed as a useful building block for various analog and digital storage and processing applications (Cauwenberghs, 1995).

The basic ideas are illustrated in Figure 5. A *hybrid link* is a set of analog processing stages (denoted A_i in the figure) followed by an A/D/A that restores the analog signal to one of M discrete attractor states. A *hybrid chain* is composed of a sequence of hybrid links. Each chain can maintain analog information to a precision of $N = \log_2(M)$ bits with a low probability of error, provided that we meet the following constraint: The net input-referred noise of the A/D/A, due to all processing stages in a link and the restoration circuits in the A/D/A, must be significantly lower than the minimum distance between attractor states. In section 4.2.1, we show that an error probability of 10^{-12} can be achieved in an N-bit hybrid link if the input-referred noise is low enough such that we operate with a precision of $N + 4$ bits. Thus, in order to restore signals reliably, we need four redundant bits of precision. To keep the error probability low in a hybrid chain composed of many links, the requisite precision before restoration needs to grow only very slowly with the number of links in the chain (like the log(log(size of the chain)).

Thus, a hybrid chain can do an extremely large amount of analog processing and still maintain a precision of $\log_2(M)$ bits at its output. Effectively, we can operate with the precision and complexity characteristic of digital systems, while doing efficient analog processing. If we assume that we do not want to have more than 8 bits of precision at the input to the A/D/A, then the best A/D/A that we can build would restore a signal to 4 bits of

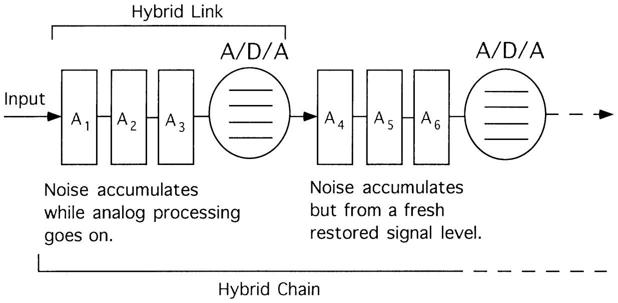
Hybrid Link
A/D/A
A/D/A
Input
A1
A2
A3
A4
A5
A6
Noise accumulates while analog processing goes on.
Noise accumulates but from a fresh restored signal level.
Hybrid Chain

Figure 5: Hybrid computation. In this form of computation, analog processing is followed by restoration of the analog signal to a set of M discrete attractor states. As discussed in section 4.2, hybrid chains allow us to operate with the precision and complexity characteristic of digital systems, while doing efficient analog processing.

precision. Using A/D/As is probably not a good technique for maintaining anything more than 4 bits of precision on an analog input. As we shall discuss in section 4.3, the main use for A/D/As is in distributed analog computation, where it is unnecessary to maintain too much precision on one wire.

To maximize the efficiency of information processing in a hybrid chain, there is an optimal amount of analog processing that must occur before signal restoration in a hybrid link; that is, hybrid links should not be too long or too short. If the link is too long, we expend too much power (or area, or both) in each analog stage to maintain the requisite precision at the input of the A/D/A. If the link is too short, we expend too much power (or area or both) in frequent signal restorations. In section 4.2.2, we analyze the optimal length of a hybrid link quantitatively. Needless to say, if we are unconcerned about efficiency, then the link can be as long or as short as we like, as long as we meet the A/D/A constraint.

4.2.1 The A/D/A. To restore a signal, we must have discrete attractor states. In digital signal restoration, the input signal is compared with a threshold, and high-gain circuits restore the output to an attractor state that is a function of the input attractor state. The input may deviate by a fairly large amount from its attractor state, and the output will still be very close to its attractor state. The noise immunity of digital circuits arises because the typical distance in voltage space between an input attractor-state level and a threshold level is many times the variance of the noise or the offset in the circuit. We can generalize this two-state restoration to an M-state restoration by having $M-1$ input threshold levels and M output state levels. The input signal is compared with $M-1$ threshold levels and is rounded off to that attractor state level that it is closest to. Systems like these have been proposed for multistate logic systems. Figure 6a shows the threshold levels V_{Ti} and restoration levels V_{Li} for a four-state or 2-bit system. The arrows converge on restoration levels and diverge from threshold levels.

The A/D/A modifies the digital restoration scheme for M states to an analog restoration scheme for M states. In the analog restoration scheme, M can be arbitrary and does not have to be 1, 2, 4, 8, 16, 32, and so on. It can be any arbitrary number that we choose because, unlike multistate logic, we do not do any digital computation with our inputs or outputs. The input V_{in} is an analog signal that may have been processed by many analog stages. The output V_{out} is a restored and filtered analog signal that can serve as an input to future analog-processing stages. Figure 6b shows a circuit for one possible implementation of a four-state A/D/A.[6] The analog signal is compared with three thresholds, and zero, one, two, or three currents are switched onto a resistor, whose voltage then equilibrates at V_{L1}, $V_{L1}+IR$, $V_{L1}+2IR$, or $V_{L1}+3IR$, respectively. The RC circuit acts as a filter and removes sharp edges in the signal. The capacitance is chosen such that $1/RC$ is at or near the desired bandwidth of the input. Figure 6a shows that if an input analog signal happens to be exactly at a threshold level V_{Ti}, then it will be constantly restored at random to the attractor state above or below it. However, since we are always within half a bit of the analog input, this random restoration still preserves the input information to within 1 bit, as desired. All other analog inputs are restored to within a half-bit of their input values as well. Thus, we preserve information in the analog signal to a precision of $\log_2 M$ bits.

Now we analyze how large the input noise and offset of the A/D/A can be if we need to preserve a precision of $\log_2 M$ bits in the output analog signal. Suppose that because of noise and offsets, the input signal is described by a gaussian probability distribution with variance σ^2, as shown in Figure 6c.[7] If the analog input is situated at a threshold level V_{Ti}, then it needs to deviate by a full 1-bit distance from this level for a bit error to occur. If, on the other hand, the analog input is situated at a restoring level V_{Li} that is not at the extremes such as V_{L1} or V_{L4}, but rather is midway such as V_{L1} and V_{L2}, then a deviation from this level by half-bit distance is sufficient for

[6] There are vastly more efficient circuit representations that we can use to construct an A/D/A. However, we do not discuss these here because they are of a more technical nature and require a background in analog circuit design.

[7] The gaussian assumption is not essential to the qualitative nature of our arguments, although it does affect our quantitative answers. If the probability distribution was not gaussian, we may still perform the calculations outlined below, although closed-form answers may not be possible.

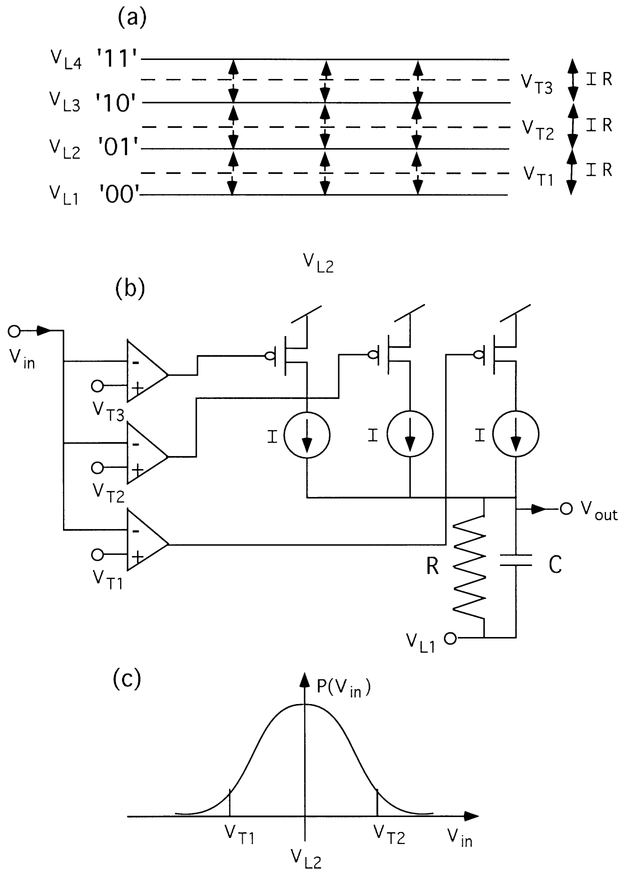

Figure 6: The A / D / A. (a) The threshold levels V_{Ti} and restoration levels V_{Li} for a four-state or 2-bit A / D / A system. (b) A circuit for one possible implementation of a four-state A / D / A. (c) The probability of a bit error for a worst-case situation when the input is at V_{L2} is given by the area under the gaussian tails—to the left of V_{T1} and to the right of V_{T2}. Section 4.2.1 provides further details.

a bit error to occur. Thus, we analyze this worst-case situation for the input situated at V_{L2}.

Let the variance of the noise be σ^2. The distance between a threshold level and a restoration level is $b_d/2$, where b_d is a bit distance given by $(V_{LM} - V_{L1})/(M - 1)$ in an M-state A / D / A. The probability of a bit error P_e is then given by the area under the gaussian tails in Figure 6c, that is, to the left of V_{T1} and to the right of V_{T2}. Thus, P_e is given by

$$P_e = \operatorname{erfc}\left(\frac{b_d/2}{\sigma\sqrt{2}}\right), \tag{4.5}$$

where erfc(x) is defined by

$$\operatorname{erfc}(x) = \frac{2}{\sqrt{\pi}}\int_x^{\infty} e^{-u^2}du \approx \frac{e^{-x^2}}{\sqrt{\pi}x}. \tag{4.6}$$

Now $P_e = 1 \times 10^{-12}$ if $b_d/(2\sigma\sqrt{2}) = 5.04$. Thus, $b_d = 2\sqrt{2} \times 5.04\sigma = 14.3\sigma$. Hence, to restore the signal faithfully, with a low bit-error rate, an N-bit A / D / A requires that the precision at its input be $\approx N + 4$ bits ($\log_2(14.3) \approx 4$).

4.2.2 The Optimal Length of a Hybrid Link. For simplicity, assume that our computation is a cascade of N identical analog processing stages, as in a many-pole filter. By the reasoning of the last paragraph of section 4.2, if the stages are not identical, we can show that an optimal length still exists. However, the closed-form solution is hard to obtain. The simplest case with all identical gains for which we may obtain a closed-form solution corresponds to all the stages having unity gain. Thus, we shall discuss only the case with identical unity-gain stages to avoid complexity that does not add much insight. For similar reasons, we shall analyze only the simple case of current (power) optimization assuming that the $1/f$ (or offset) terms in the resource noise equation (3.6) are negligible. Other simplifying assumptions also include that $p = 1$(subthreshold) and that we pay a fixed cost in power per A / D / A restoration stage.[8] Suppose that there are M computational stages and 1 A / D / A in every hybrid link. Then, there will be N/M links with a total of N computational stages, and N/M A / D / As in the chain. Suppose that the complexities of the A / D / A stage and of each computational stage correspond to n_r and n_c devices, respectively. By equation 4.5, corresponding to whatever error criterion we pick, the input-referred noise σ at every A / D / A must be less than or equal to some value σ_t. The value of σ_t depends on only b_d, the distance between attractor states in the A / D / A, which is fixed by the precision desired for a given hybrid chain. Thus, from equations 3.14, 3.11, and 3.13, with $n_i = n_c$ for all i, and $G_i = 1$ for all i, the noise due to the computational stages in a link is given by

$$v_c{}^2 = \frac{(Mn_c)^2 K_w(1)\Delta f}{I_C}, \tag{4.7}$$

[8] There is a possible variant of the problem, where we simultaneously optimize the power allocated between the A/D/A stages and the computation stages, as well as the number of stages per link.

where I_C is the total power consumption in the computational stages. Similarly, the noise due to an A / D / A stage in a link is given by

$$v_r^2 = \frac{(n_r)^2 K_w(1)\Delta f}{I_R}, \tag{4.8}$$

where I_R is the fixed current consumption of the restoration stage. The A / D / A constraint gives us

$$v_c^2 + v_r^2 = {\sigma_t}^2. \tag{4.9}$$

Algebraic manipulation of equations 4.7, 4.8, and 4.9 then yields

$$I_C = M^2 \left(\frac{n_c^2 K_w(1)\Delta f}{{\sigma_t}^2 - \frac{n_r^2 K_w(1)\Delta f}{I_R}} \right), \tag{4.10}$$

$$= M^2 C_c, \tag{4.11}$$

where C_c is defined by the preceding equations. The total current consumption due to N/M links in the entire chain is then given by

$$I_{CH} = \left(\frac{N}{M}\right)(I_C + I_R),$$

$$= N\left(C_c M + \frac{I_R}{M}\right). \tag{4.12}$$

Figure 7 shows a plot of the current consumption for differing values of C_c = 2.06 pA, 5.11 pA, 9.52 pA, and 15.34 pA; I_R is fixed at 100 pA. The parameter C_c was changed by varying σ_t in equation 4.10. Thus, as we increase the precision of the hybrid link, the costs of computation rise with respect to the costs of signal restoration, and the optimal length of the link decreases. The mathematics is in accord with the intuition expressed in the last paragraph of section 4.2. The curves in Figure 7 were drawn for $\Delta f = 100$ Hz, $K_w(1) = 4.38 \times 10^{-22}$, $n_r = 3$, and n_c = 150. It is easy to show that the location of the optimum in equation 4.12 is given by

$$M = \sqrt{\frac{I_R}{C_c}}. \tag{4.13}$$

4.3 Distributed and Hybrid Computation. Figure 8 combines the ideas of sections 4.1 and 4.2. The information from a single-wire analog input is encoded onto many wires by an analog encoder. Typically, the encoder might be more redundant and thus might distribute the information over

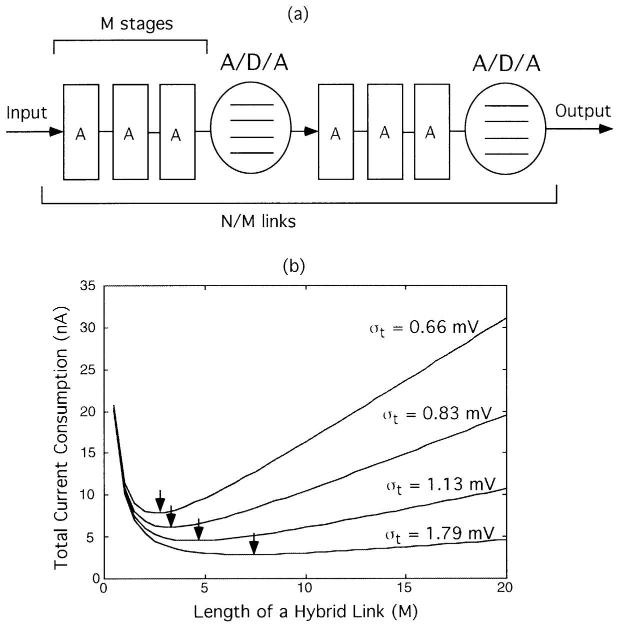

Figure 7: Optimal length of a hybrid link. (a) A hybrid chain with M stages of computation per link, N/M links, and N total stages of analog computation. (b) A plot of the current consumption (obtained from equation 4.12) versus link length (M) for differing values of precision, parameterized by σ_t, the input-referred noise at the A/D/A. As the precision increases, the optimal length of the hybrid link is shortened. Section 4.2.2 provides further details.

many more wires, but for simplicity, we have shown a nonredundant encoder. A cochlea, retina, and A/D are all good examples of encoders that distribute information from one wire onto many wires. In this example, we have an analog encoder, so if we used an A/D, we would have to follow it with a D/A. In the example of Figure 8, the distributed information is preserved in the first stage of processing by 2-bit A/D/As. In the next stage of processing, the analog processors or the A/D/As, or both, make decisions based on the information and reduce the output information to 1 bit. Thus, the analog circuits in the second half can afford to be noisier, since the A/D/A restoration has a precision of only 1 bit. The use of distributed analog computation and low-precision A/D/A signal restoration makes this architecture ideal for efficient precise computation.

Mixed-signal circuits that involve analog and digital techniques have

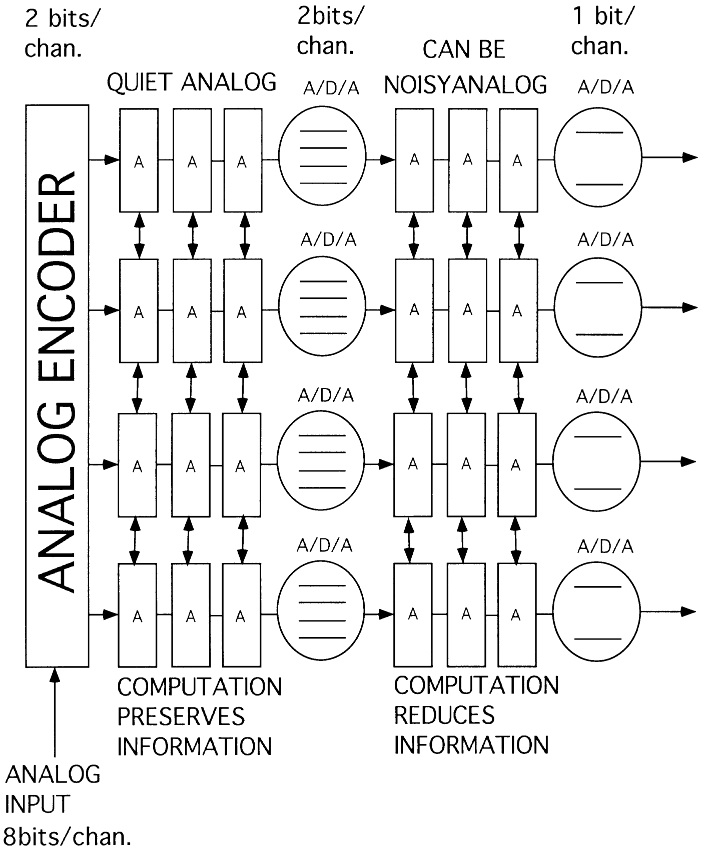

Figure 8: Distributed and hybrid computation. The information from a single-wire analog input is encoded onto many wires by an analog encoder such as a cochlea, retina, or A/D. Interacting hybrid chains process the information on these wires. Section 4.3 provides further details.

been proposed for efficient low-precision sensory data processing (Martin, 1996). Distributed-and-hybrid schemes, such as ours, illustrate how mixed-signal circuits can be architected to be suited for high-precision processing as well. For example, it is possible to implement efficient high-precision arithmetic circuits using distributed-and-hybrid architectures. The results from several low-precision analog addition and multiplication operations are appropriately combined via carry and A/D/A interactions. A more detailed description of such arithmetic architectures is outside the scope of this article and a topic of our research; these architectures may represent the first practical applications of the ideas described in this article.

5 Extrapolating to Neurobiology

Our analysis for electronic systems suggests why neuronal information processing is distributed, that information processing in the brain is likely to be hybrid, and how signal restoration in neurons may be implemented. In sections 5.1 through 5.3 we discuss these suggestions in more detail. In sections 5.4 through 5.5 we shall discuss how our arguments about noise in electronic systems can be extrapolated to neurobiology.

5.1 Why Neuronal Information Processing Is Distributed. Information processing in networks of neurons is accomplished in a tremendously distributed fashion. It has often been pointed out that this distribution results in fault-tolerant behavior, since the destruction of any one neuron or synapse hardly affects the operation of the overall network. However, we suggest that the primary reason for the distributed nature of neuronal information processing is not fault tolerance but efficiency. We showed in section 4.1 that if the costs of computation are to be cheap, then the information and information processing must be distributed across as many wires as possible. However, if the costs of communication are to be cheap, then the information and information processing must be localized among as few wires as possible. The trade-off between these two constraints, as revealed in equation 4.1, results in an optimal number of wires and an optimal signal-to-noise ratio per wire, as revealed in Figure 4. In neurobiological systems, where communication costs are relatively low compared with communication costs in silicon, the optimal signal-to-noise ratio is lower than that in silicon.[9] Thus, we believe that nature was smart to distribute computational resources over many noisy neurons (dendrites and somas) and communicate that information between neurons over many noisy fibers (axons). The noisiness of the brain is due to the wisdom of millions of years of evolution, and is not a reflection of the incompetence of biology. We believe that the "use" of neuronal noise in phenomena such as stochastic resonance, or in phenomena that prevent trapping in a local minima, may be valuable in certain special cases, but the primary reason for the noisy nature of the brain is efficiency.

Experimentally based estimates of the energy required to transmit a bit of information in various stages of the blowfly retina are rather large (Laughlin,

[9] In today's electronic technology, it would be unthinkable even to dream of wiring on the scale of neurobiology. For example, the million fibers of the optic nerve or the 35,000 fibers of the auditory nerve would simply be too expensive to implement.

van Stevenick, & Anderson, 1998a; Laughlin, Anderson, O'Carroll, & van Stevenick, 1998b). Therefore, these authors have independently arrived at conclusions very similar to ours: Distributed coding of information among multiple pathways is important for energy efficiency in noise-limited systems.

5.2 Information Processing in the Brain Is Likely to Be Hybrid. Action potentials are all-or-none discrete events that usually occur at or near the soma or axon hillock. In contrast, dendritic processing usually involves graded synaptic computation and graded nonlinear spatiotemporal processing. The inputs to the dendrites are caused by discrete events. Thus, in neuronal information processing, there is a constant alternation between spiking and nonspiking representations of information. This alternation is reminiscent of the constant alternation between discrete and continuous representations of information in Figure 5. Thus, it is tempting to view a single neuron as a D / A / D. However, although the firing of a spike is a discrete event, it does not imply that it encodes information about a discrete state. The information encoded by a spike is meaningful only in relation to spikes in different neurons, or in relation to earlier or later spikes in the same neuron. If these relationships are analog, then all-or-none events do not imply the encoding of discrete states. So how do we know whether the brain is analog (continuous signal) or digital (discrete signal) or hybrid (both)? Almost everybody accepts that the brain does a tremendous amount of analog processing. The controversy lies in whether there is anything digital about it.

We know, from the arguments of this article, that the noise accumulation in complex systems is simply too high for purely analog processing to be efficient in such systems. Given that the brain is made up of a large number of physical devices that exhibit noise at room temperature and is yet extremely efficient (12 W power consumption and 300 ms response time for complex tasks), we may hypothesize that it must be mixing continuous-signal and discrete-signal processing to compute in a hybrid fashion. In section 5.4 we review noise in biological devices, and in Sarpeshkar (1997) we review numbers on the interconnectivity and complexity of the brain's architecture. These reviews suggest that although it is theoretically possible that the brain's complexity is small enough that a purely analog brain could be efficient, a purely analog brain seems unlikely. However, more quantitative studies need to be done on noise in biological devices and on the architecture of the brain before we can conclusively rule out the possibility of a purely analog brain. Thus, the suggestion that the brain is hybrid is only a hypothesis supported by our quantitative arguments from electronics and by some qualitative facts from our current knowledge of neurobiology.

5.3 How Signal Restoration May Be Implemented. To implement signal restoration, there must be a set of discrete states that the continuous signal is periodically restored to. How are the discrete restorative states of neurons encoded in the firing of action potentials? Conceptually, at the level of a single neuron, the discrete states of a spike train may be encoded in the number of spikes that occur in a given window of time (the mean-firing-rate code), or in a discrete set of firing patterns that occur within that same window of time (the timing-pattern code). Such codes are scalar codes since they involve only one neuron. As experimental (Abeles, Bergman, & Gat, 1995) and theoretical (Hopfield & Herz, 1995) work indicate, it is more likely that discrete states involve a vector code that is implemented in a collective fashion across many neurons. The window of time over which we count spikes or detect temporal patterns within the spikes is determined by the integration time constants of the neurons.

The mean firing rate and timing pattern scalar codes have direct analogies in vector codes. In the mean firing rate case, instead of counting the number of spikes in one neuron within a window of time, we count the number of spikes across many neurons that are present within some time window. In the timing pattern case, instead of a discrete set of firing patterns of one neuron that occur within some time window, we have a discrete set of cross-correlated firing patterns of many neurons within some time window. For simplicity, we shall assume that our time window is short enough that each neuron contributes at most one spike within that time window. It is easy to generalize our ideas to multiple spikes within one time window.

The key building block of our electronic signal restoration schemes was the A/D/A, which was basically an A/D followed by a D/A. In the signal-representation scheme of neurons, how might we build a A/D/A? We shall discuss only signal restoration for vector codes.

5.3.1 Von Neumann Restoration for Spike Counting. Suppose we have $3N$ neurons. We group them into N sets of three each. For each of the N sets we perform a simple majority vote and regenerate three signals, each of which encodes the result of the majority vote. Thus, if we have (spike, spike, no spike) across the three neurons, we restore this signal to (spike, spike, spike). If we have (no spike, spike, no spike) across the three neurons, then we restore this signal to (no spike, no spike, no spike). Thus, we restore the original $3N + 1$ possible states (ordering of neurons does not matter) into $N + 1$ possible states. Just as in the A/D/A, if we want to have low rates of error, we must compute with more redundancy (20N instead of 3N). The majority-vote scheme was first proposed by John Von Neumann (Von Neumann, 1952) as a way for doing signal restoration. Note that in this scheme, we are really restoring a fine-grained discrete quantity to a coarse-grained discrete quantity. In the A/D/A, we restore a continuous analog quantity with a fine-grain size determined by analog noise into a coarse-grained discrete quantity.

5.3.2 Restoration for Spike Timing. Here, we detect the presence of a discrete timing pattern by building suitable delays in the dendrites or synapses or input axons of a "matched-filter" neuron such that the inputs from the N neurons that encode the timing pattern arrive in synchrony at the axon hillock (Abeles, 1991). The matched-filter neuron regenerates the timing pattern by fanning out collaterals to a set of N output neurons with appropriate axonal or synaptic delays such that the timing pattern is regenerated. The restoration in pattern timing will occur if the matched-filter neuron is configured to respond to inputs with somewhat skewed timing patterns; this is accomplished by setting its threshold so it is not too high. If we want to restore M timing patterns that are encoded on N input axons, then we need M matched-filter neurons and N output neurons. Each of the N output neurons could receive inputs in parallel from the M matched-filter neurons, as, in a good design, only one of the M matched-filter neurons would be active at any given time. As in the A / D / A, if we want to ensure low error rates, M should be significantly less than the possible number of timing patterns encoded among the N neurons. It is also crucial that the delays involved in regeneration be precise enough to maintain a precision that is a few bits above $\log_2(M)$ bits.

In digital electronic circuits, an inverter performs restoration and computation at the same time. It inverts its input (1 goes to 0, and 0 goes to 1), but it is also restorative since a "bad 1" is restored to a "good 0." Similarly, in a time-delay restoration scheme, we could have the regenerated pattern be a different timing pattern such that a somewhat skewed input temporal pattern is restored to a clean-output temporal pattern. In a pattern-recognition computation, such as that performed by an associative memory, computation and restoration are intermingled because the nature of the computation inherently requires a discrete set of outputs.

5.3.3 Caveats. We have made many simplifying assumptions such as treating computation and restoration as distinct entities, and similarly treating computation and communication as separate entities. It is likely that such entities are more deeply intertwined in the brain and that the rather sharp digital restorations that we propose are really soft restorations in the brain, such that a more accurate description would need to involve the language of complex nonlinear dynamical systems. The processing of information in a single dendrite, let alone the whole brain, is enormously complex. Such processing could be very useful in performing signal restoration within the level of the dendrite itself. Thus, we do not, by any means, claim that the brain is implementing the particular architectures, and the particular restorative schemes that we have proposed. We have merely offered our schemes as a possible way in the hope that it will stimulate further discussion and work on the subject.

The importance of action potentials for avoiding temporal dispersion and signal attenuation over long communication distances is well known.

That is not the issue under discussion in this article since we take that issue to be resolved. Rather, the issue under discussion revolves around the importance of action potentials for signal restoration in complex local networks of neurons. In the latter case, the complexity of computation degrades the signal-to-noise ratio due to the large number of processing steps, and restorative action-potential codes serve to preserve the signal-to-noise ratio.

In this article, we have emphasized that the hybrid and distributed nature of the brain's signal processing is likely to be an important and underappreciated reason for its efficiency. Other reasons for the efficiency of the brain are discussed in detail in Sarpeshkar (1997). They include the marvelous technology of devices and interconnect available to the brain, its nonlinear and adaptive signal processing strategies, and its strategies for processing only the information that is useful for solving a given computational task.

5.4 Noise in Biological Devices. In any technology, the starting point for an analysis of the information costs of computing is the noise resource equation of that technology. It was the noise resource equation for MOS technology (see equation 3.6) that enabled us to construct a set of resource precision equations (equations 3.17 and 3.18). The resource precision equations evaluated the costs of a computation as a function of the output information or precision. What might the noise resource equations for neurobiological devices look like? Due to the great diversity of biological devices and the incomplete knowledge that we have about their functioning, a quantitative theory for the technology of neurobiology seems premature. However, we can make qualitative statements that reveal how the noise can be decreased with an increase in resource consumption.

The limiting form of noise in biological devices is typically the randomness in ion channel openings and closings (DeFelice, 1981) and the unreliability of synaptic vesicle release (Allen & Stevens, 1994). Channels transition between discrete closed and open states with certain finite probabilities per unit time. The transition probabilities depend on the membrane voltage or the chemical concentration of a substance. For a good discussion of the kinetics of ion channels, see Weiss (1996). The noise can be reduced by $\sqrt{N}$ law-of-large-numbers averaging over several ionic channels (i.e., through the increase of ion channel densities). Similarly, the noise of synaptic transmission may be reduced through the use of averaging over many vesicles, many synaptic contacts, and so on. Such averaging costs area and also turns up power consumption since the power per unit channel, vesicle, or contact is approximately constant. It is intuitive to expect that averaging over large areas of membrane would improve offsets and $1/f$ noise, but we are unaware of any actual experimental measurements that address whether they do. Interestingly, as in electronics, the magnitude of the $1/f$ noise in biology is highly unpredictable. It is dependent on the concentrations in the cellular environment of substances that alter transport properties of nerve membrane (DeFelice, 1981). In electronics, $1/f$ noise is also strongly dependent on the concentrations of impurities in an insulating membrane, the gate oxide.

Averaging strategies were at the root of a reduction in noise in electronic systems as well. In electronics, we averaged over more electrons per unit time (to reduce thermal noise by increasing power), or over more traps and impurity defects (to reduce $1/f$ noise and offset by increasing area). In biology, we average over more ion channels or over larger stretches of membrane. Indeed, a simple $\sqrt{N}$ averaging would yield noise resource equations that are similar to equation 3.6, with $p = 1$. However, there are suggestions that neurobiological systems may be even smarter and may attain noise reductions that scale like $1/N$ rather than like $1/\sqrt{N}$ (Salman, Soen, & Braun,1996); such scaling laws require the use of interactions between channel kinetics and membrane kinetics through the use of membrane voltage feedback.

In situations where it is important to maintain reliability and precision, such as at a neuromuscular junction, there is a lot of averaging over numerous synaptic connections and synaptic vesicles. In situations where it is not that important to be very reliable, such as in the highly distributed architecture of cortex, there is little averaging over synapses or vesicles (Koch, 1997). When timing must be precise, synapses are typically large (Zhang & Trussell, 1994). From numerous examples, it is qualitatively clear that the reduction of noise is accomplished through resource consumption in neurobiology, as it is in electronics. Neurobiology and electronics behave similarly because physical and mathematical laws like the laws of thermodynamics and the law of large numbers do not change with technologies.

5.5 Noise in Neurobiological Systems. In section 3.2, we abstracted the mapping from computational task to circuit topology in the parameters M and n_i. The ingenuity of the designer lies in mapping the task to the primitives and architecture of the technology, such that M and n_i are as small as possible. Consequently, when function is well mapped to structure, noise is minimized; the wiring of the architecture also is more efficient. Computational architectures where function and structure are well matched amplify the computational information above the noise in the components. This amplification is analogous to the way a matched filter amplifies the signal above the background noise.

Two topologies that may be completely equivalent functionally may have markedly different noise properties. For example, suppose that in topology A we take the difference between a large, positive current and a large, negative current to output a small differential current; in topology B we just output a small differential current. The noise of topology A will be much higher than that of topology B even though the two topologies may be indistinguishable as far as outputs go. Thus, the mapping from function to structure must be done with care.

It is clear that natural structures have evolved to match structure and function. The architectures of the cochlea, the retina, the hippocampus, the cerebellum, the neocortex, and various other regions have patterns of connectivity, cellular organization, and cell differentiation that indicate a close relationship between structure and function. Cells of various anatomical types are specialized to have certain functional characteristics. (For a good review, see Shepherd, 1990.)

For noise minimization, resource allocation should be increased in the initial and complex stages of a computation, as discussed in section 3.2.

6 Summary

We conclude by reviewing the main points of the article:

1. Analog computation is efficient at low-precision processing, and digital computation is efficient at high-precision processing. The resource precision equations for analog computation (equations 3.17 and 3.18) and the resource precision equations for digital computation (equations 3.19 and 3.20) quantify the costs of computing at a certain precision in MOS technology. Figure 3 shows a plot of the costs of analog and digital computation at different levels of precision. The noise resource equation of a technology (equation 3.6 for MOS technology) determines the form of the resource precision curves for that technology.

2. The advantages of analog computation arise from its exploitation of physical primitives for computation. The advantages of digital computation arise from its multiwire representation of information and information processing, and from its signal restoration properties.

3. Analog computation that distributes its precision and processing resources over many wires is maximally efficient at a certain signal-to-noise ratio per wire, due to the trade-offs between computation and communication. Equation 4.1 and Figure 4 illustrate this fact in more detail.

4. We proposed a hybrid architecture that combines the advantages of discrete-signal restoration with the advantages of continuous-signal, continuous-time analog computation. The key building block of such a hybrid scheme is a restoration circuit called an A/D/A, which is described in section 4.2.1. Figures 5 and 6 illustrate the workings of the hybrid scheme. For maximum efficiency in a computation, there is an optimal amount of continuous analog processing that must be done before a discrete signal restoration; Figure 7 and equation 4.12 illustrate how this optimum can be determined.

5. We described a computing architecture that illustrates the simultaneous working of distributed and hybrid computation in section 4.3 and Figure 8. Distributed and hybrid computation combines the best of the analog and digital worlds to create a world that is more efficient than either.

6. In neurobiological systems, where communication costs are relatively low compared with communication costs in silicon, the optimal signal-to-noise ratio per wire is lower than that in silicon. Thus, we believe that nature was smart to distribute computational resources over many noisy neurons (dendrites and somas) and communicate information between neurons over many noisy wires (axons).

7. Since the brain appears to be extremely efficient in its information processing and hybrid representations are the most efficient representations in massively complex systems, it is likely that the brain uses hybrid representations.

8. Experiments suggest that discrete states in the brain are encoded in the cross-correlated firing patterns of neurons in a network (Abeles et al., 1995). Neuronal information processing is thus most likely to involve vector signal restoration. In section 5.3, we discussed how signal restoration in networks of neurons may be implemented using A / D / A-like schemes.

9. From numerous examples, it is qualitatively clear that in neurobiology, the reduction of noise is accomplished through resource consumption as it is in electronics. Neurobiology and electronics behave similarly because physical and mathematical laws such as the laws of thermodynamics and the law of large numbers do not change with technologies. It is such laws that, with a few technology-dependent parameters, determine noise resource equations. Since our conclusions depend only on general properties of noise resource equations such as a polynomial reduction in noise with resource consumption, we suggest that our extrapolation from electronics to neurobiology is correct to leading order.

Acknowledgments

I thank Carver A. Mead, John Allman, Christof Koch, Yaser Abu Mostafa, Kwabena Boahen, Sanjoy Mahajan, Mike Levene, David Kewley, and a couple of anonymous reviewers for useful discussions. As well, I thank Lyn Dupre for her detailed editing of the manuscript, and Shih–chii Liu for a thorough review of this work.

References

Abeles, M. (1991). *Corticonics* (pp. 227–259). Cambridge: Cambridge University Press.

Abeles, M., Bergman, H., & Gat, I. (1995). Cortical activity flips among quasi-stationary states. *Proceedings of the National Academy of Sciences of the United States of America, 92,* 8616–8620.

Aiello, L. C., & Wheeler, P. (1995). The expensive-tissue hypothesis: The brain and digestive system in human and primate evolution. *Current Anthropology, 36*,199–221.

Allen, C., & Stevens, C. F. (1994). An evaluation of causes for unreliability of synaptic transmission. *Proceedings of the National Academy of Sciences, 91,* 10380–10383.

Allman, J. (1990). Evolution of neocortex. *Cerebral Cortex, 8A*, 269–283.

Cauwenberghs, G. (1995). A micropower CMOS algorithmic A / D / A converter. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 42,* 913–919.

Chandrakasan, A., Sheng, S., & Brodersen, R. W. (1992). Low-power CMOS digital design. *IEEE Journal of Solid-State Circuits, 27*, 473–484.

Cover, T., & Thomas, J. (1991). *Elements of information theory* (p. 20). New York: Wiley.

DeFelice, L. J. (1981). *Introduction to membrane noise*. New York: Plenum Press.

Denyer & Renshaw. (1985). *VLSI signal processing: A bit serial approach*. Reading, MA: Addison-Wesley.

Furth, P. M., & Andreou, A. G. (1996). Bit-energy comparison of discrete and continuous signal representations at the circuit level. *Proceedings of the 4th Physics of Computation Conference*. Boston.

Haus, H., & Adler, R. (1959). *Circuit theory of linear noisy networks.*

Hopfield, J., & Herz, A. V. M. (1995). Rapid local synchronization of action potentials: Toward computation with coupled integrate-and-fire neurons. *Proceedings of the National Academy of Sciences, 92*, 6655–6662.

Hosticka, B. J. (1985). Performance comparison of analog and digital circuits. *Proceedings of the IEEE, 73*, 25–29.

Kearns, D. A. (1993). Experiments in very large-scale analog computation. Unpublished doctoral dissertation, California Institute of Technology.

Koch, C. (1997). *Biophysics of computation: Information processing in single neurons*. Unpublished manuscript.

Laughlin, S., de Ruyter van Stevenick, R., & Anderson, J. C. (1998a). The metabolic cost of neural information. *Nature Neuroscience, 1*.

Laughlin, S., Anderson, J. C., O'Carroll, D., & de Ruyter van Stevenick, R. (1998b). Coding efficiency and the metabolic cost of sensory and neural information. In *Information theory and the Brain*. R. Baddeley, P. Hancock, & P. Foldiak (Eds.), Cambridge: Cambridge University Press.

Martin, D. A. (1996). *ADAP: A mixed-Signal array processor with early vision applications*. Unpublished doctoral dissertation, Massachusetts Institute of Technology.

Mead, C. A. (1989). *Analog VLSI and neural systems*. Reading, MA: Addison-Wesley.

Mead, C. A. (1990). Neuromorphic electronic systems. *Proceedings of the IEEE, 78*, 1629–1636.

Mead, C. A., & Conway, L. (1980). *Introduction to VLSI systems*. Reading, MA: Addison-Wesley.

Rabaey, J. (1996). *Digital integrated circuits*. Englewood Cliffs, N.J.: Prentice Hall.

Salman, H., Soen, Y., & Braun, E. (1996). Voltage fluctuations and collective effects in ion-channel protein ensembles. *Physics Review Letters, 77*, 4458–4461.

Sarpeshkar, R. (1997). Efficient precise computation with noisy components: Extrapolating from an electronic cochlea to the brain. Unpublished doctoral dissertation, California Institute of Technology. Chapter 5: Section 6, Appendix A, Appendix B, and Appendix C; Postscript copy available on http://www.pcmp.caltech.edu/anaprose/rahul/thesis/.

Sarpeshkar, R., Delbrück, T., & Mead, C. (1993). White noise in MOS transistors and resistors. *IEEE Circuits and Devices, 9*, 23–29.

Sarpeshkar, R., Lyon, R. F., & Mead, C. A. (1998). A low-power wide-dynamic-range analog VLSI cochlea. *Analog Integrated Circuits and Signal Processing, 16*, 3; Postscript copy available on
http://www.pcmp.caltech.edu/anaprose/rahul/cochlea/.

Shepherd, G. M. (1990). *The synaptic organization of the brain*. Oxford: Oxford University Press.

Vittoz, E. A. (1990). Future of analog in the VLSI environment. *Proceedings of the International Symposium on Circuits and Systems, 2*, 1347–1350.

Von Neumann, J. (1952). *Probabilistic logics and the synthesis of reliable organisms from unreliable components*. Lecture delivered at the California Institute of Technology, Pasadena, CA, January 4–15.

Weiss, T. F. (1996). *Cellular biophysics* (2 vols.) Cambridge, MA: MIT Press.

Zhang, S., & Trussell, L. O. (1994). A characterization of excitatory postsynaptic potentials in the avian nucleus magnocellularis. *Journal of Neurophysiology, 72*, 705–718.

Received February 3, 1997; accepted March 4, 1998.