Parameter Estimation of a Known Chaotic Time Series Corrupted by White Gaussian Noise

John DiCecco
Electrical, Computer, and Biomedical Engineering Department
University of Rhode Island
Kingston, Rhode Island, 02881

May 1, 2007

Abstract

The subject of parameter estimation in linear signals embedded in white Gaussian noise has been extensively studied. The subject of nonlinear choatic signals, however, has received much less attention. This paper will examine some of the known techniques for estimating the parameters of chaotic signals such as iterative maps, including the tent map and the logistic map.

1 Introduction

Chaotic signals present a special difficulty in parameter estimation. The difficulty arises from the definition of a chaotic system: sensitive dependence on initial conditions. Very slight changes in the initial condition(s) cause significant effects in the evolution of the time series [1]. While chaotic systems are nonlinear by definition and apparently random, they are deterministic. Once the intial condition is found, the whole time series is known. For the purposes of this paper, two particular chaotic systems will be investigated. The logistic map, given by

$$s[n+1] = As[n](1 - s[n]) \tag{1}$$

for $0 < A < 4$, is a 1-D map representing exponential growth in discrete time [1]. A somewhat simpler map is given by the tent map,

$$s[n+1] = \begin{cases} 2s[n] & s[n] \leq 0.5 \\ -2s[n] + 2 & s[n] > 0.5 \end{cases} \tag{2}$$

which is conjugate to the logistic map, and so named because of its shape (fig. 1) [1][2]. The critical issue of parameter estimation when confronted with 1-D chaotic unimodal iterative maps is noninveribility.

Each iterate $s[n]$ has two preimages $s[n-1]$ given by

$$s[n-1] = 0.5 \pm \sqrt{0.25 - \frac{s[n]}{A}} \tag{3}$$

for the logistic map. These images typically are different, resulting in a loss of one bit (factor of 2) of information for each iteration, since there is no way to determine which of the preimages generated the value $s[n]$ [1].

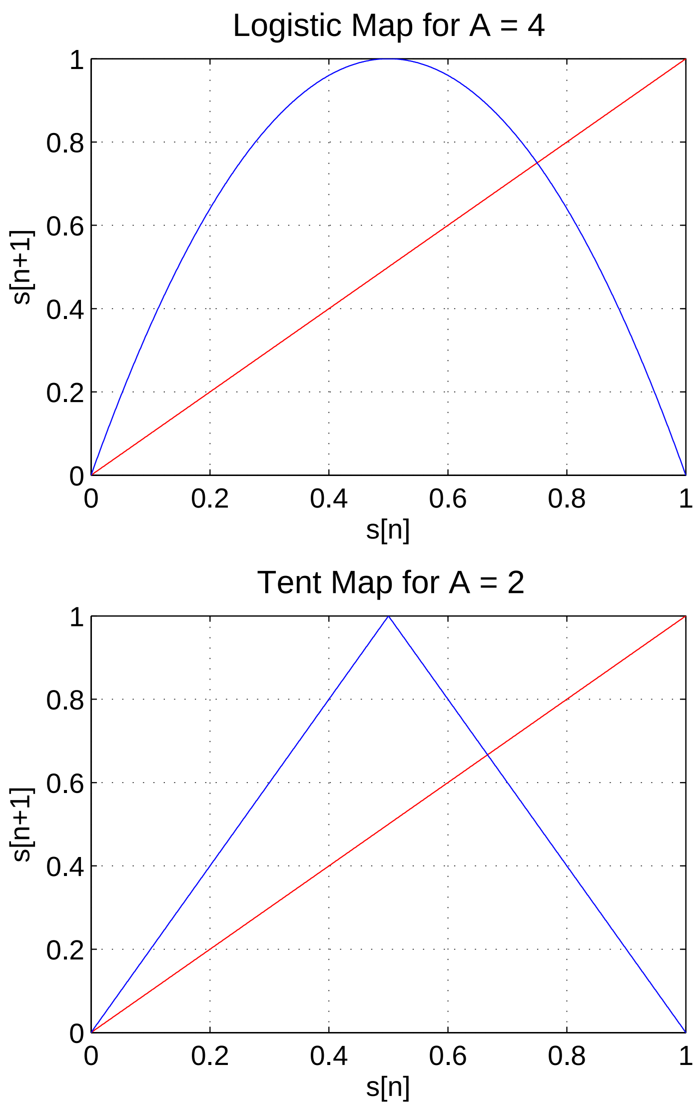

Figure 1: The Logistic Map (Top) and Tent Map (bottom)

2 Cramer-Rao Lower Bound

Given that an arbitrary signal (chaotic or not) is embedded in noise, the ability to estimate any of the values that parameterize such a signal is directly dependent on the probability density function of the noise. For this paper, the noise model will be based on zero mean white Gaussian noise (WGN). The signal dependence on the parameter θ can then be modeled using the likelihood function, given by

$$p(\mathbf{x};\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma)^{\frac{N}{2}}} \exp\left[-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x[n]-s[n;\theta])^2\right]. \tag{4}$$

The Cramer-Rao lower bound (CRLB) provides the achievable lowest limit of the variance of an unbiased estimator [3]. To achieve the CRLB for a signal parameterized by θ, the variance of any unbiased estimator must satisfy

$$var(\hat{\theta}) \geq \frac{1}{-E\left[\frac{\partial^2 \ln p(\mathbf{x};\theta)}{\partial\theta^2}\right]} \tag{5}$$

The expression in the denominator of the CRLB is known as the Fisher information, which is expressed as

$$I(\theta) = -E\left[\frac{\partial^2 \ln p(\mathbf{x};\theta)}{\partial\theta^2}\right]. \tag{6}$$

For the general signal parameterized by theta ($s[n;\theta]$) embedded in WGN, the CRLB is given by

$$var(\hat{\theta}) \geq \frac{\sigma^2}{\sum\limits_{n=0}^{N-1}\left[\frac{\partial s[n;\theta]}{\partial\theta}\right]^2}. \tag{7}$$

In the case of estimating the initial condtion of the chaotic signal $s[0]$, the CRLB can be shown to be

$$I^{-1}(\hat{s}[0]) = \frac{\sigma^2 s[1]}{\sum\limits_{n=0}^{N-1} 4^n s[n+1]} \tag{8}$$

where $I(\hat{s}[0])$ is the Fisher information for $\hat{s}[0]$ [5].

3 Maximum Likelihood Estimation

The Maximum Likelihood Estimator (MLE) is perhaps the most widely used estimator for practical data problems [3]. The heuristic nature of the method makes it well suited for complex data sets. For the problem of estimating parameters of a chaotic signal embedded in WGN, the model is

$$x[n] = s[n] + w[n] \qquad n = 0, 1, ..., N-1 \tag{9}$$

for $s[n]$ a deterministic, but chaotic, signal based on the nonlinear noninvertible map

$$s[n] = f(s[n-1]). \tag{10}$$

The likelihood function can then be written as

$$p(\mathbf{x};\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma)^{\frac{N}{2}}} \exp\left[-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}(x - s[n;\theta])^2\right] \tag{11}$$

where θ is the parameter to be estimated. This paper is primarily concerned with addressing the estimation of the initial condtion $s[0]$ of a chaotic time series. The intitial condition of a chaotic system is of great importance because the time series, though chaotic and seemingly random, is deterministic, so once the initial condition is determined, the entire time series is known [1].
The MLE for the initial condition is found by minimizing

$$J(s[0]) = \sum_{n=0}^{N-1}(x[n] - s[n])^2 \tag{12}$$

where $s[n] = f^{(n)}(s[0])$ and $f^{(n)}$ is the n^{th} order iterate of the map [2][5]. This is essentially a least squares estimator[5]. There are, however, inherent difficulties with this approach to estimating parameters from known chaotic maps. The likelihood function for such maps is highly irregular and is known to have fractal (self similar) characteristics [5].
The MLE of the initial condition of the logistic map is known to be consistent, unbiased, achieves the lower bound, and assumes a Gaussian distribution, under certain conditions, as the variance of the noise goes to zero [4][5]. For instance, under normal circumstances, two chaotic signals with different initial conditions must diverge as the map is iterated. However, the logistic map will yield the same signal for the initial condition $s[0]$ and the initial condition $1 - s[0]$. But since the logistic map is twice continuously differentiable and satisfies the condition

$$\sum_{n=0}^{N-1}(s[n;\theta_1] - s[n;\theta_2])^2 = 0, iff \quad \theta_1 = \theta_2 \tag{13}$$

it can be shown that $\hat{s}[0] \rightarrow s[0]$, with probabilty one [3]. This means that the properties of consistency and normality do not hold as $N \rightarrow \infty$, but rather they hold as the variance of the noise goes to zero.

4 The Logistic Map

The logistic map has very interesting properties for varying values of the parameter A in (1). It is known that the map is chaotic once A reaches the accumulation point at $A = 3.5699456718...$[1]. The map goes in and out of chaos from this value to $A = 4$, where it is known that any intial condition results in chaotic orbits (figs. 2 and 3) [1]. For the remainder of this paper, it is assumed that the map is chaotic with parameter $A = 4$ unless specifically noted otherwise.

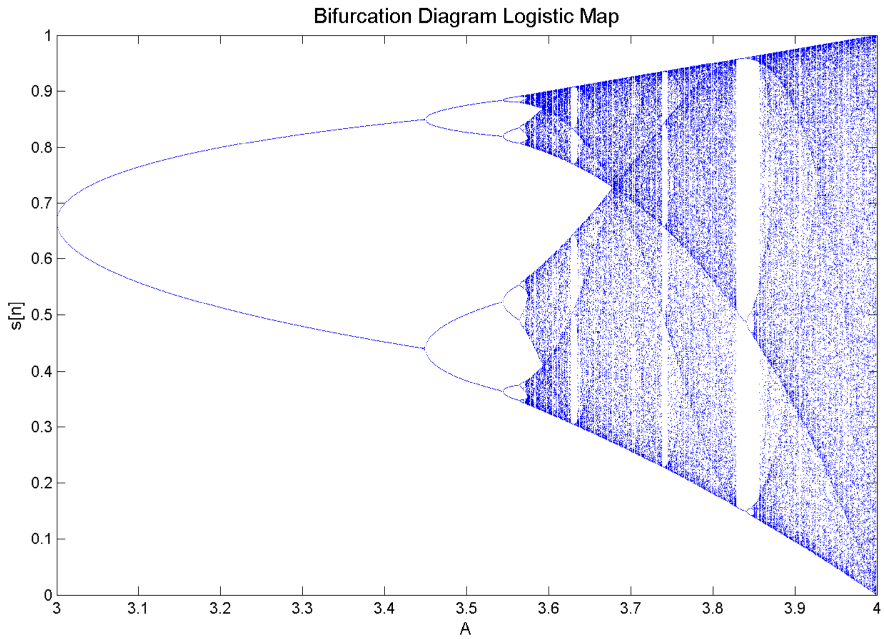

Figure 2: Bifurcation map for $3 < A < 4$. Beyond $A = 3.5699456718...$, chaos.

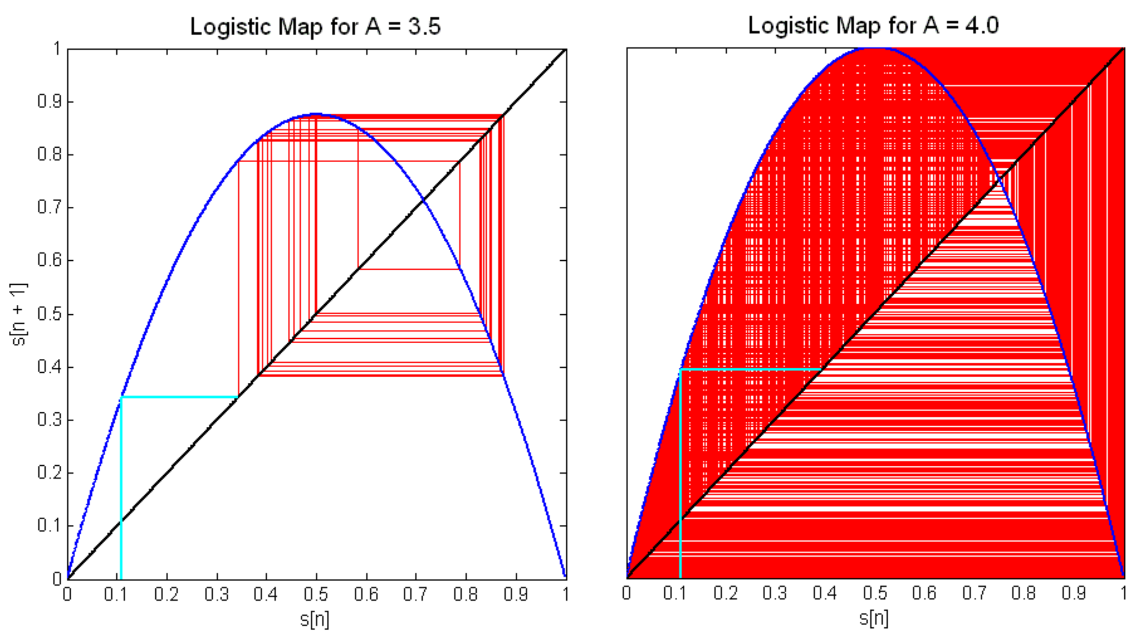

Figure 3: Cobweb evolution of the logistic map. Each trip around one box of the web corresponds to one iteration of the equation. In the case for $A = 4$, any initial condition leads to chaos.

The notion of chaos implies three things: 1) the system is bounded 2) it is aperiodic and 3) has sensitive dependence on initial condions. Of course boundedness and periodicity are relatively straightforward things to prove. The dependence on initial conditions, however, is a bit more difficult to show. Graphically, however, this dependency is easy to see. Consider fig. 4. Clearly, after just 4 iterations, the signal resulting from an initial condition of 0.5 and the signal resulting from an initial condition of 0.501 are showing signs of diverging. After 10 iterations, the divergence is more pronounced. In fact, the divergence is exponential. This is quantified by the Lyapunov exponent and it is used to prove the existence of chaos [1]. There are a host of estimation schemes to determine the value of the Lyapunov exponent from a time series, most of which assume that the initial condition is known.

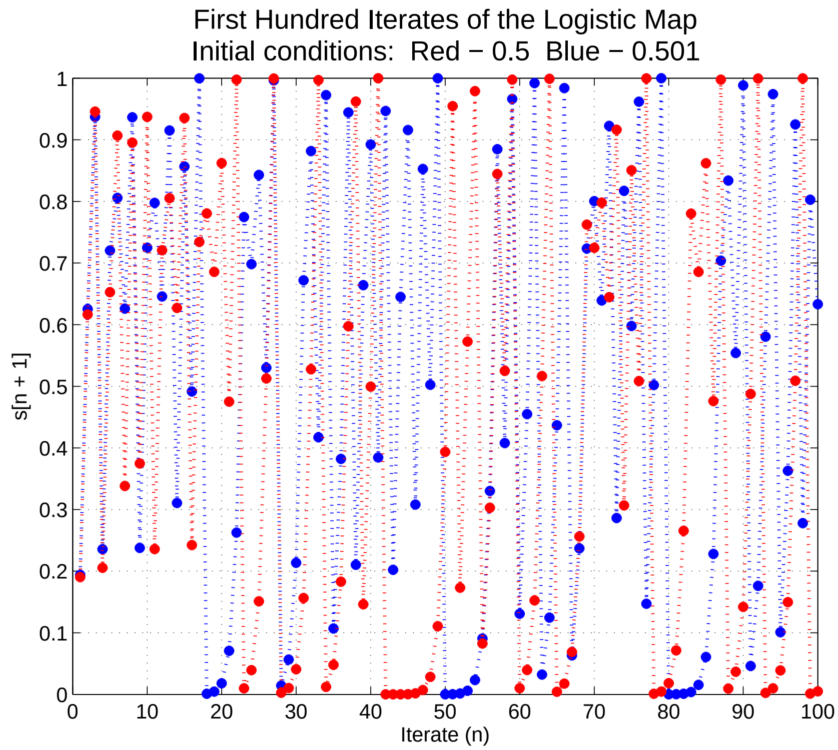

Figure 4: Sensitive dependence on initial conditions.

It is tempting to use the iteration function in (10) in the forward direction to estimate the initial condition. However, as demonstrated in fig. 4, the propagation of errors is extremely unforgiving and this approach will fail.

4.1 The Itinerary

In order to work around the propagation of errors difficulty, the data sequence $\{s[0], s[1], ..., s[N-1]\}$ is replaced by the itinerary $\{p_n\}_{n=0}^{N-1}$, where for the logistic map

$$p_n = \begin{cases} 0 & 0 \leq s[n] < 0.5 \\ 1 & 0.5 \leq s[n] < 1 \end{cases} \tag{14}$$

since the map reaches a maximum at 0.5. In this way, it is possible to account for the fact that each iteration generates two preimages $s[n-1]$ of the next value $s[n]$. Without the itinerary, it would be impossible to know which of the two preimages preceeded the current sample (fig. 5). In fact, it is precisely this loss of information that makes the forward iteration scheme impractical for parameter estimation.

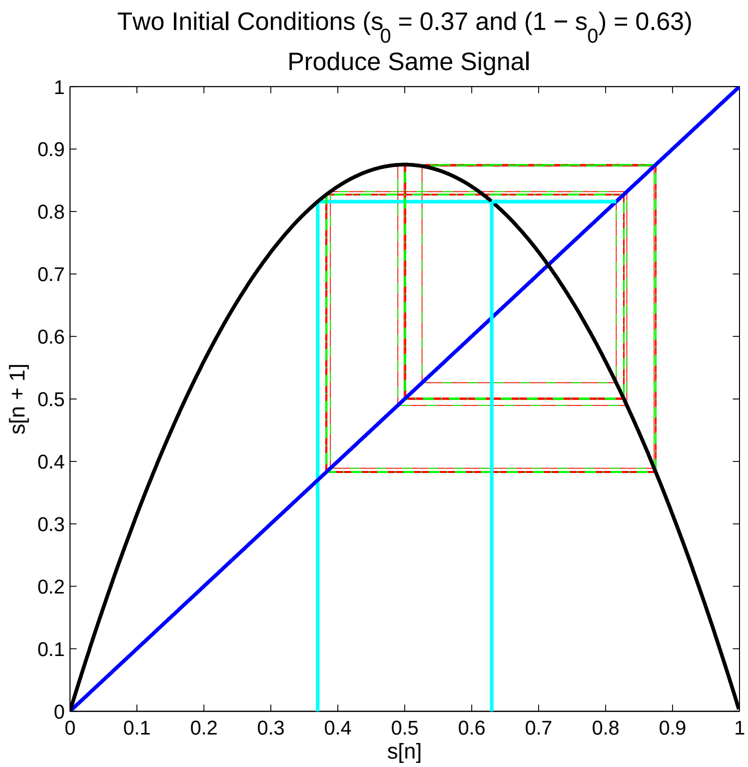

Figure 5: Two different initial conditions $s[0] = 0.37$ and $s[0] = 0.63$ produce the exact same signal. The green and red trajectories trace right on top of one another. Note that the height of the map does not reach 1 for this graph. The parameter A has been changed from 4 (chaotic) to 3.5 (periodic) for the purpose of demonstrating that two different initial conditions lead to the same signal.

4.2 Topological Conjugacy

A particularly useful relationship exists between the tent map (2) and the logistic map (1); they are topologically conjugate. Mathematically, two functions are topologically conjugate if there exists a homeomorphism h that allows one function to map to the other on their native intervals [2]. That is, let $f : x \to x$ and $g : y \to y$. If

$$h(f(x)) = g(h(y)) \tag{15}$$

then the maps associated with the two functions f and g are said to be topological conjugates, and h is the homeomorphism that maps the function f to the function g [2]. This is easy to see by considering fig. 6. The conjugate function (homeomorphism) that maps the tent map to the logistic map is

$$h(x) = \sin^2\left(\frac{\pi}{2}x\right). \tag{16}$$

An inverse function h^{-1} is easily derived to yield

$$h^{-1}(x) = \frac{1}{\pi}\arccos^2(1-2x) \tag{17}$$

where x is on the interval $0 \rightarrow 1$.

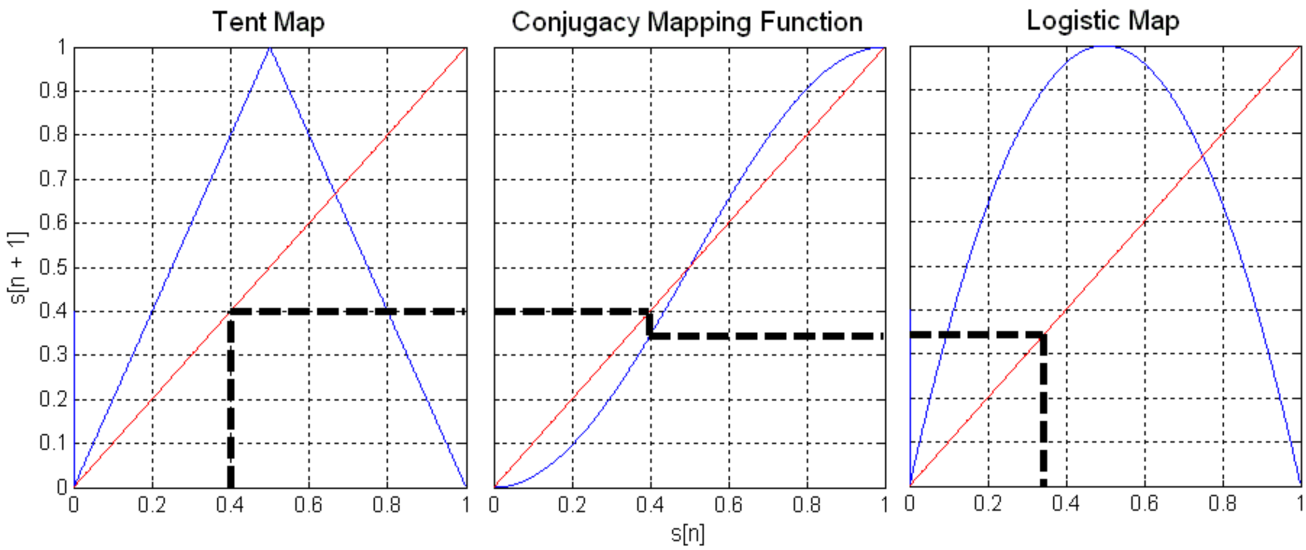

Figure 6: Conjugacy relationship between the tent map and the logistic map.

5 The Halving Method

As noted earlier, there are inherent difficulties when attempting to propagate any iterate map in the forward direction, i.e. $s[n] = f(s[n-1])$. Upon closer inspection, it will be very easy to see why, and how the nature of this difficulty can be used to work around it.

Suppose the value for $s[40]$ were known and one needed to estimate the initial condition $s[0]$. Recall from (3) that each $s[n-1]$ has two preimages associated with it. That means that there are 2^{40}, 1.1×10^{12}, possible preimages for $s[n]$. Clearly, the noninvertibility of the map creates a logistical impossibility of keeping track of the preimages without some bookkeeping scheme. However, recall that the itinerary of the signal allows us to mark each preimage $s[n-1]$ based on $s[n]$. This string of $1's$ and $0's$ creates a binary representation of the preimages.

The function operating upon $s[n-1]$ in (10) can be rewritten in terms of other functions as

$$f = h \circ T \circ h^{-1} \tag{18}$$

where h is defined in (16), T is the tent map (2), h^{-1} in (17), and $\circ$ defines a function order operator. That is, $h \circ T \circ h^{-1}$ is equivalent to $h(T(h^{-1}(s[n-1])))$.

If $s[n]$ is represented in the binary format, the effect of the tent map is to shift left for the argument of T in the interval $0 \leq arg < 0.5$ and shift left and complement if the argument is in the interval $0.5 \leq arg < 1$ [5]. This is readily seen if we consider the tent map with initial condition $s[0] = \frac{8}{25}$, which has a binary representation of 0.01010001111010111. Of course $\frac{8}{25} < 0.5$ so $s[1] = \frac{16}{25}$, which has a binary representation of 0.1010001111010111, as expected, since this is just the left-shifted version of $\frac{8}{25}$. For verification of the complement effect, since $\frac{16}{25} > 0.5$, the next iteration of the tent map produces $s[2] = -\frac{32}{25} + 2$, which is $\frac{18}{25}$. This has a binary representation of 0.10111000010100011, the left-shifted and complemented value of $s[1]$.

If the itinerary p_n is now determined from $h^{-1}(s[n])$ instead of $s[n]$, for $s[n]$ derived from the logistic map, this is clearly equivalent to mapping the logistic map onto the tent map. The itinerary is now that of the tent map with the initial condition of the logisitic map (fig. 6). All that remains to estimate $s[0]$ is to recursively determine

$$b_{n+1} = b_n \oplus p_n \tag{19}$$

where $\oplus$ denotes the *exclusive or* (xor) and b_{n+1} is determined by

$$h^{-1}(s[n]) = \begin{cases} 0.b_{n+1}b_{n+2}..., & if\ b_n = 0 \\ 0.\bar{b}_{n+1}\bar{b}_{n+2}..., & if\ b_n = 1 \end{cases} \tag{20}$$

To start the recursion, $b_1 = p_0$ [4]. Using (17) to map back to the logistic map, the halving method for estimating the initial condition is given by

$$\hat{s}[0] = \sin^2\left(\frac{\pi}{2}\sum_{n=1}^{N}\hat{b}_n 2^{-n}\right)[4]. \tag{21}$$

Essentially, the halving method reduces the interval in which the intial conditon must reside by half at each n iteration. The method can continue ad infinitum, in theory, or at least to $N-1$, or until the accuracy criteria are met. Caution must be used with this method, however, when using it to determine an initial condition of a chaotic map. Recall the central theme of a chaotic system: sensitive dependence on initial conditions. Now consider a modest time series length of $N = 10$. The detectable error is, at most, $\frac{1}{2^N}$ or approximately 0.001 [4]. This amount of error is catastrophic in initial condition estimation, as was shown in fig. 4.

Of course, if enough samples exist, there are no difficulties with getting an accurate estimation according to (21), since the halving method continues to divide the interval in half as $\frac{1}{2^N}$. A time series length of 100 samples, for instance, produces a maximum error of 10^{-31}! To illustrate this, consider the estimation of an initial condition from a time series where 10 samples are available in one instance, and 100 samples are available in the other. The initial condition is known to be 0.31. In the case of $N = 10$, the estimation from (21) produces $\hat{s}[0] = 0.30865828381746$, where as $N = 100$ produces

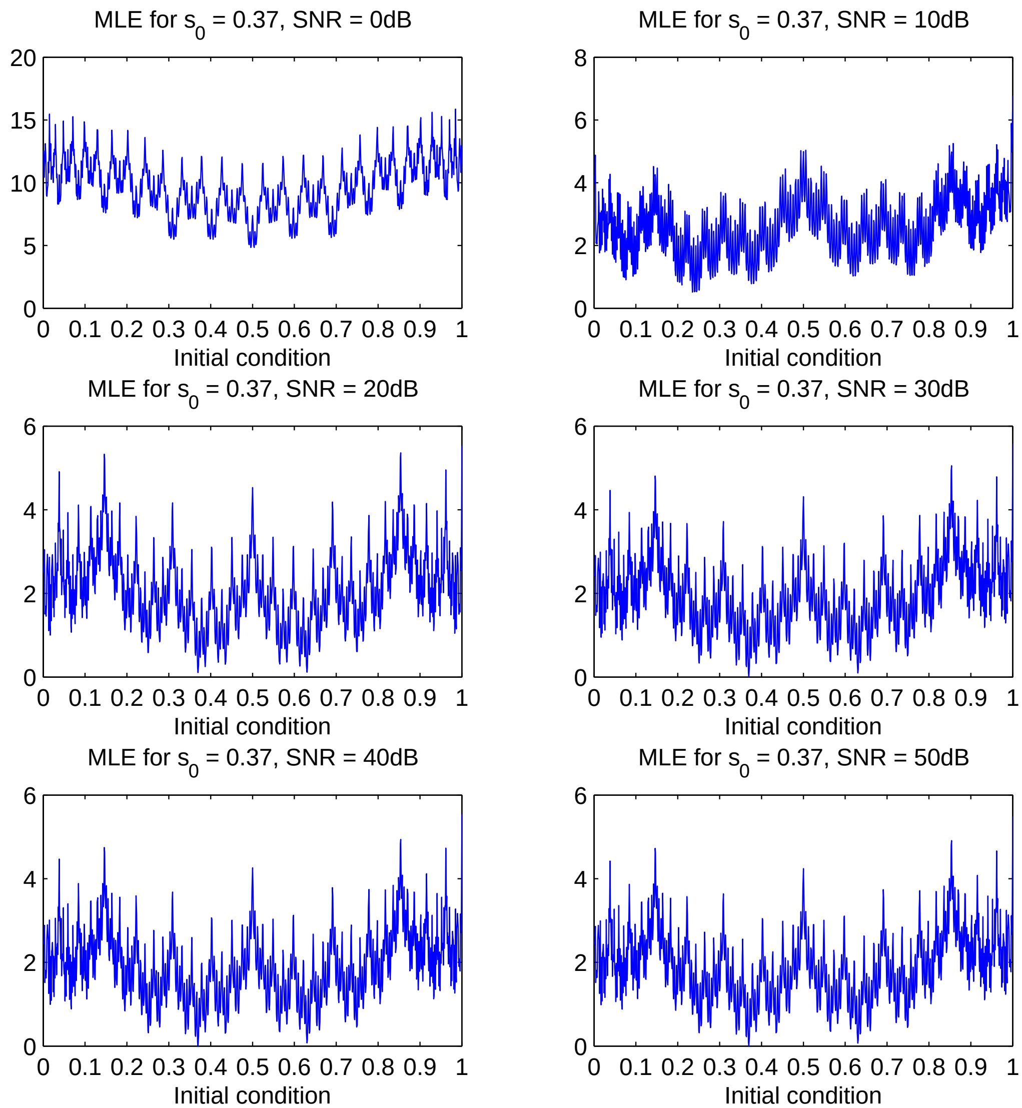

Figure 7: Performance of the MLE. Note that at high SNR, the minimum of the function estimates the correct initial condition ($s[0] = 0.37$), but also identifies the value $1 - s[0]$, since these two initial conditions produce the same time series. At low SNR, the MLE fails to identify the initial condition (top,left and right).

$\hat{s}[0] = 0.31000000000000$. For $N > 100$, the error is less than computational error so further lengths of N are superfluous.

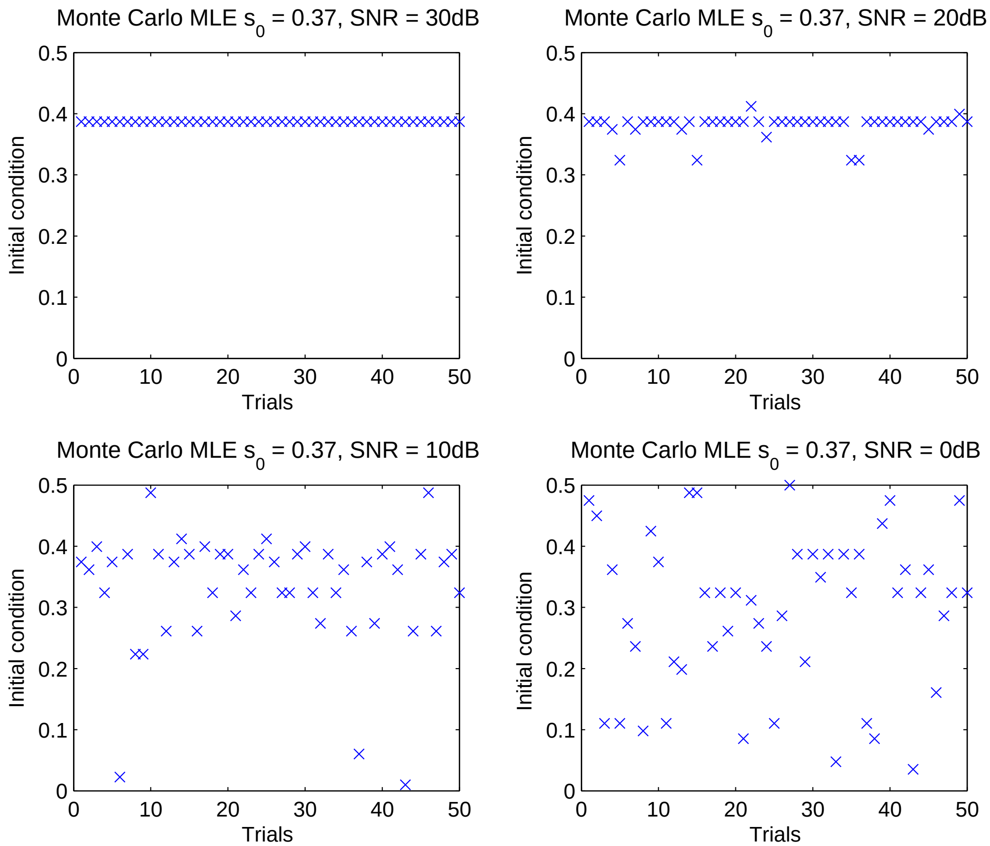

Figure 8: An SNR lower than 30dB causes the MLE to produce inconsistent estimations.

6 Performance of Estimators and the Cramer-Rao Lower Bound

In [5], the author aludes to the fact that the MLE is fractal in nature. This is readily seen in fig. 7. Notice that the overall MLE is composed of smaller versions of itself. This is the hallmark of self-similar, fractal characteristics. Berliner [6] notes that concentrated, spikey liklihoods generally lead to accurate estimates - for well behaved problems. It is well established, however, that chaotic systems are not well behaved. In this series, a grid search is used with the MLE to determine the initial conditon with a data record length of $N = 10$. In the case of fig. 7, the initial conditon $s[0]$ was arbitrarily chosen to be 0.37. For high signal-to-noise ratios (SNR) and a grid search size proportional to the accuracy of the initial condition, the MLE performs well, as expected. But even at an SNR of 20dB, the MLE breaks down and is not consistent. This is due to the fact that introducing a small amount of error in the initial condition, the evolution of the time series changes dramatically and it is impossible to determine the initial condition with any regularity (fig. 8). Of course, the MLE over the interval $0 \rightarrow 1$ is symmetric, since it was noted earlier that two different initial conditions will produce the same signal. This is easily remedied by the itinerary, since the itinerary contains the mapping that indicates which half of the interval the initial condition originated.

As stated in (8) the CRLB for the initial condition of the logistic map is known to be

$$I^{-1}(s[0]) = \frac{\sigma^2 s[1]}{\sum_{n=0}^{N-1} 4^n s[n+1]}. \tag{22}$$

This is derived from (7) by setting $s[n] = \sin^2(\pi 2^n \alpha)$ and $\alpha = \frac{1}{2\pi}\arccos(1 - 2s[0])$ [5]. To determine the performance of the MLE and the halving method, the power in dB of the mean square error (MSE) is plotted versus the SNR in (fig. 9). As expected, at high SNR, both the halving method and the MLE using the grid search method perform well. Recall that the MLE for the logistic map achieves the CRLB as SNR$\rightarrow \infty$. The plots in fig. 9 were generated using a Monte Carlo simulation of 100 trials. The small fluctuations in the values about the CRLB are due to statistical variations introduced by the simulations, since the bias, at high SNR, goes to zero (fig. 10).

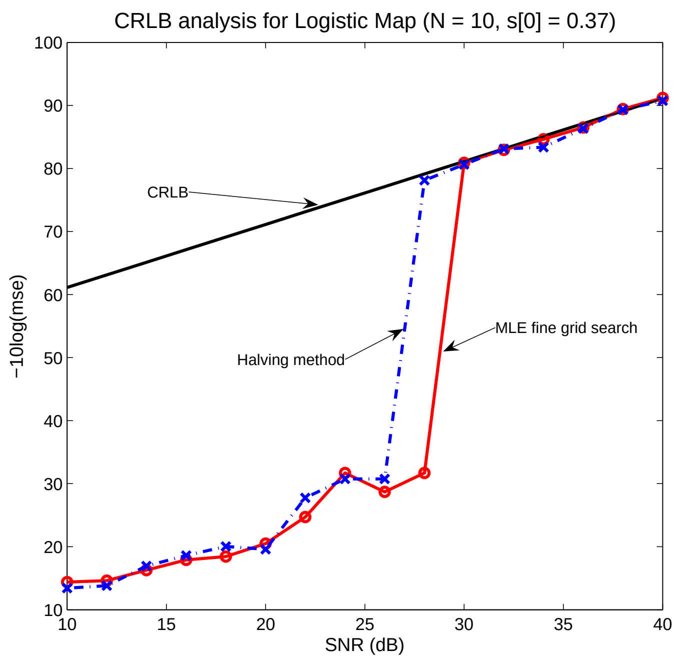

Figure 9: Comparison of the performance of the estimation methods.

6.1 CRLB Plotting Techniques

That the performance of an estimator is related to the length of the data record is well understood. In fact, it is the exponential relationship with the length that forms the bound. However, it has been noted that chaotic systems have a minimal relationship to data record lengths. Because of this, it is important to address the size of the estimation error, since large errors will not be able to attain the bound. In the case of the halving method, for a data record length of $N = 10$, the estimator can only achieve an error of approximately 0.001. (This is due to the fact that the halving method is bounded by $\frac{1}{2^N}$, as indicated in (21).) To compensate for this, [4] proposed a method to perform a coarse search with a modification to (21). A coarse estimate is first obtained by

$$\hat{s}[0] = \sin^2\left(\frac{\pi}{2}\left(\sum_{n=1}^{N}\hat{b}_n 2^{-n} + 2^{(N+1)}\right)\right) \tag{23}$$

The effect of this added term is to center the estimate on the interval $\left(\sum_{n=1}^{N}\hat{b}_n 2^{-n}, \sum_{n=1}^{N}\hat{b}_n 2^{-n} + \frac{1}{2^N}\right)$ before performing a grid search using 100 points in the interval [4]. The search is repeated to narrow the interval and reduce the error in the estimate.

The halving method is a suboptimal, but computationally efficient algorithm [5][7]. Its performance is similar to that of the MLE grid search method but involves less iteration and computational complexity. There appear to be no distinct advantages to using the MLE other than the fact that it is straightforward to compute, albeit with a longer computation time.

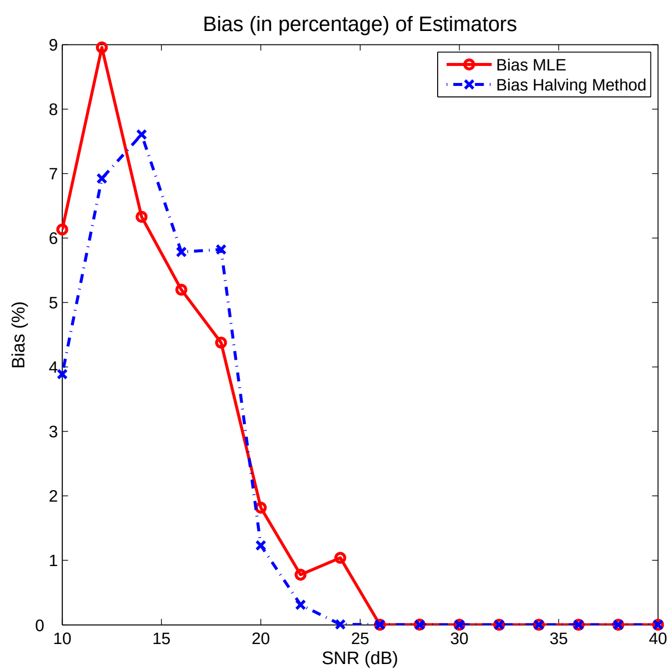

Figure 10: Bias of the estimation methods.

References

[1] J. Sprott, *Chaos and Time-Series Analysis.* Oxford, UK: Oxford University Press, 2004.

[2] H. O. Peitgen, H. Jurgens, and D. Saupe, *Chaos and Fractals: New Frontiers in Science.* Berlin, Vienna, New York: Springer-Verlag, 1992.

[3] S. Kay, *Fundamentals of statistical signal processing: estimation theory.* Upper Saddle River, NJ: Prentice Hall, Inc., 1993.

[4] S. Kay and V. Nagesha, "Methods for Chaotic Signal Estimation," *IEEE Transactions on Signal Processing*, vol. 43, pp. 2013-2016, Aug. 1995.

[5] S. Kay, "Asymptotic Maximum Likelihood Estimator Performance for Chaotic Signals in Noise," *IEEE Transactions on Signal Processing*, vol. 43, pp. 1009-1012, Apr. 1995.

[6] L. Mark Berliner, "Likelihood and Bayesian Prediction of Chaotic Systems," *Journal of the American Statistical Association*, vol. 86, no. 416, pp. 938-952, Dec. 1991.

[7] L. Cong, W. Xiaofu, S. Songgeng, "A General Efficient Method for Chaotic Signal Estimation," *IEEE Transactions on Signal Processing*, vol. 47, pp. 1424-1428, May 1999.

A Appendix - MATLAB™code

```
%%%%%%%%%%%%%%%%%FINAL PROJECT ELE 661%%%%%%%%%%%%%%%%%%
clear all
close all
A = 4;


x = 0.0001:.001:1;
N = length(x);
F_x = A*x.*(1-x);            %%logistic map

hold on
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%COBBWEB
xi = 0.37000000;             %%initial condition
%xi = 1 - xi;
x0 = xi;
x1 = A*xi*(1-xi);
x11 = x1;
xn = [x0];
xi = x0;
for i = 2:length(x)
    x1 = A*xi*(1-xi);
    xn = [xn x1];
    plot([xi xi],[xi x1],'r-',[xi x1],[x1 x1],'r-')
    %plot([xi xi],[xi x1],'-.g',[xi x1],[x1 x1],'-.g')
    xi = x1;
end
plot(x,x,'b','LineWidth',2)
plot(x,F_x,'k','LineWidth',2)
plot([x0 x0],[0 x11],'c-',[x0 x11],[x11 x11],'c-','LineWidth',2)
```

```
hold off


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%MLE
snr = 10.^-[1:0.2:4];
K = 10;
for kk = 1:length(snr)
        for M = 1:50
            x_init = linspace(0.001,1,1000);
            sn = xn(1:K) + sqrt(snr(kk))*randn(1,K);

            %coarse search
            for k = 1:length(x_init)
                MLE(k) = sum((sn - logistic_it(A,x_init(k),K-1)).^2);
            end
            [b,minf] = min(MLE);
            hol(kk,M) = x_init(minf);

            %fine search
            x_init_fine = linspace(x_init(minf)-0.001,x_init(minf)+0.001,1000);
            for ll = 1:length(x_init_fine)
                MLE_fine(ll) = sum((sn - logistic_it(A,x_init_fine(ll),K-1)).^2);
            end

            [bb,minff] = min(MLE_fine);
            holf(kk,M) = x_init_fine(minff);
            error(M) = (x_init_fine(minff) - x0)^2;
        end
    mse(kk) = (1/M)*sum(error);
    bias(kk) = abs(mean(holf(kk,:)) - x0);
    %subplot(3,3,kk),plot(x_init,MLE),hold on
end
% figure,subplot(3,2,1),plot(hol(1,:),'x'),axis([0 M 0 0.5])
% subplot(3,2,2),plot(hol(2,:),'x'),axis([0 M 0 0.5])
% subplot(3,2,3),plot(hol(3,:),'x'),axis([0 M 0 0.5])
% subplot(3,2,4),plot(hol(4,:),'x'),axis([0 M 0 0.5])
% subplot(3,2,5),plot(hol(5,:),'x'),axis([0 M 0 0.5])
% subplot(3,2,6),plot(hol(6,:),'x'),axis([0 M 0 0.5])

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%crlb
snp1(1) = [x0];
s1 = x0;
for k = 1:K
    sn = 4*s1*(1-s1);
    snp1(k+1) = sn;
    s1 = sn;
end
%n = 0:length(snp1)-1;
% crlb = snr.*snp1(2)/(sum(4.^(0:K-1).*snp1(2:end))); %% CRLB for logistic map
```

```
% plot(10*log10(1./snr),-10*log10(crlb),'k'),grid
% hold on
% plot(10*log10(1./snr),-10*log10(mse),'-or')
% xlabel('SNR (dB)'),ylabel('-10log(mse)')
% title('CRLB analysis for Logistic Map (N = 10, s[0] = 0.37)')
%figure,plot(10*log10(1./snr),bias)

iterates = logisticplot(x0,N);

%%%%%%%%%%%%%%%%%%%%%%%%%ITINERARY
itinerary = [];
if iterates(1)<0.5
    itinerary = 0;
else
    itinerary = 1;
end
for i=2:N
    xi=iterates(i);
    if xi<0.5
        itinerary = [itinerary,0];
    else
        itinerary = [itinerary,1];
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%HOMEOMORPHISMS
h_x = (sin(pi/2*x)).^2;
h_inv_x = 1/pi*acos(1-2*x);
%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%B_n+1

for k = 1:length(snr)
        for M = 1:50
            s = iterates(1:K) + sqrt(snr(k))*randn(1,K);

            h_inv_s = (1/pi)*acos(1 - 2*s);
            for ii = 1:length(s)
                if h_inv_s(ii)<0.5
                    pn(ii) = 0;
                else
                    pn(ii) = 1;
                end
            end

            b = [];
            b(1) = pn(1);

```

```
            for n = 2:length(pn)
                b(n) = xor(b(n-1),pn(n));
            end
            b(end) = [];
            n = 1:length(b);
            %coarse search
            s_hat = (sin((pi/2)*sum(b.*2.^(-n))))^2;

            %first fine search
            s_fine = linspace(s_hat-2^-K,s_hat+2^-K,100);
            for ll = 1:length(s_fine)
                MLE_s_fine(ll) = sum((s - logistic_it(4,s_fine(ll),K-1)).^2);
            end
            [bb,minff] = min(MLE_s_fine);
            holf(k,M) = s_fine(minff);

            %second fine search
            ss_fine = linspace(s_fine(minff)-2^-K,s_fine(minff)+2^-K,100);
            for ll = 1:length(ss_fine)
                MLE_ss_fine(ll) = sum((s - logistic_it(4,ss_fine(ll),K-1)).^2);
            end
            [bb,minfff] = min(MLE_ss_fine);
            holff(k,M) = ss_fine(minfff);

            error2(M) = (ss_fine(minfff) - x0)^2;
        end
    mse2(k) = (1/M)*sum(error2);
    bias2(k) = abs(mean(holff(k,:)) - x0);
end

crlb = snr.*snp1(2)/(sum(4.^(0:K-1).*snp1(2:end))); %% CRLB for logistic map
plot(10*log10(1./snr),-10*log10(crlb),'k'),grid
hold on
plot(10*log10(1./snr),-10*log10(mse),'-or',10*log10(1./snr),-10*log10(mse2),'-xk')
xlabel('SNR (dB)'),ylabel('-10log(mse)')
title('CRLB analysis for Logistic Map (N = 10, s[0] = 0.37)')
figure,plot(-10*log10(snr),bias,'-or',-10*log10(snr),bias2,'-xk')
```