Functional Bootstrap: a hardware constrained implementation of on-line bootstrap.

B. Apolloni°, D. Malchiodi°°,J. G. Taylor° °°

°Dipartimento di Scienze dell'Informazione, Università di Milano, Italy
apolloni@dsi.unimi.it

° °Dipartimento di Matematica "F. Enriquez", Università di Milano, Italy
malchiodi@mat.unimi.it

°°°Department of Mathematics, King's College, London, UK
j.taylor@mth.kcl.ac.uk

KEY WORDS: functional bootstrap, on-line bootstrap, neural networks, learning algorithms, hardware simulator

ABSTRACT: We cope with the key step of bootstrap methods of generating a possibly infinite sequence of random data preserving properties of the distribution law, starting from a primary sample actually drawn from this distribution.

We deal with two hardware resource constraints: i. absence of a long term memory, requiring an on line estimation of the bootstrap generator parameters, and ii. limited amount of mass memory, binding the number of statistics that can be collected at run-time.

We use a probabilistic Random Access Memory (pRAM) neural network as a suitable hardware with the mentioned constraints, and we split the the bootstrap sampling into the generation of many bernoullian variables. Each variable, since represents the random value of a single bit conditioned by the values assumed by others, identify its statistics with the content of the addresses the pRAM memory.

On this hardware, on-line estimation has been obtained by a *learning by gossips* model which properly manages the run-time values of correlated estimating processes. An entropic rule has been used for decimating the conditional distributions to a number storable into the pRAM memory.

Since the law of the bootstrap sample is now determined by the inner structure of the trained hardware, we speak of the functional bootstrap.

Physical limitations, open technical problems, extensibility and effectiveness of the method are discussed and exhibited through numerical examples.

1. Introduction.

Given a function g(s) of a sample s of a random variable **X**, there might be at least two reasons to prefer to infer its behaviour from a direct simulation rather than deduce it analytically from the distribution law L of **X**: i. we do not know L, ii. the analytical computation is much too hard.

As well known, bootstrap simulation consists in obtaining n replicas s* of s, drawing each s* with replacement from s, and studying the variable g(s*) (Efron and Tibshirani, 1993).

Having fixed the length of **X** in bits as ν, where X_i is the Bernoulli ($\{0,1\}$ valued) random variable representing its i-th bit, $\forall\ (x_1,x_2...,\ x_\nu) \in \{0,1\}^\nu$, the density function $f_{\mathbf{X}}$ of **X** is given by:

$f_{\mathbf{X}}(\mathbf{x}) = P(X_1=x_1;\ X_2=x_2;...;\ X_\nu=x_\nu) =$

$P(X_1=x_1)\ P(X_2=x_2|X_1=x_1)\ P(X_3=x_3|X_1=x_1,X_2=x_2)...P(X_\nu=x_\nu|X_1=x_1;...;X_{\nu-1}=x_{\nu-1})=$

$= \alpha(1)^{x_1}\ (1-\alpha(1))^{1-x_1}\ \alpha_{x_1}(2)^{x_2}\ (1-\alpha_{x_1}(2))^{1-x_2}\ \alpha_{x_1,x_2}(i)^{x_3}(1-\alpha_{x_1,x_2}(i))^{1-x_3}$

$\alpha_{x_1,x_2,...,x_{\nu-1}}(\nu)^{x_\nu}\ (1-\alpha_{x_1,x_2,...,x_{\nu-1}}(\nu))^{1-x_\nu}$

having introduced the set of 2^ν-1 parameters

$\{\alpha_{x_1,x_2,...,x_{i-1}}(i) = P(X_i=1|X_1=x_1;...;\ X_{i-1}=x_{i-1})\}$, with $\alpha(1) = P(X_1=1)$.

The Efron key operation of drawing s* from s coincides with sampling the elements X* of s* from the empirical distribution function $\hat{F}(\mathbf{x}) = \frac{1}{\#s}\sum_{\mathbf{x}_i \in s} I_{(-\infty,x]}(\mathbf{x}_i)$, where $I_{(-\infty,x]}(\mathbf{x}_i)$ is the indicator function of the set $(-\infty,x]$; thus, by definition, $I_{(-\infty,x]}(\mathbf{x}_i) = 1$ if and only if $\mathbf{x}_i$ belongs to $(-\infty,x]$ .This is equivalent to drawing the single bits of **X*** from the set of the 2^ν-1 bernoullian variables defined above, where each $\alpha_{x_1,x_2,...,x_{i-1}}(i)$ is substituted by the estimate

$$\hat{\alpha}_{x_1,x_2,...,x_{i-1}}(i) = \frac{1}{\#\{\mathbf{x}\in s \text{ such that } X_1=x_1,...,X_{i-1}=x_{i-1}\}} \sum_{\mathbf{x}\in s \text{ such that } X_1=x_1,...,X_{i-1}=x_{i-1}} I_{\{1\}}(x_i)$$

The set of quantities $\hat{\alpha}_{x_1,x_2,...,x_{i-1}}(i)$ form a probabilistic look-up table, being in essence the contents of the addresses of the probabilistic Random Access Memory (pRAM) machine. This machine is a stochastic neural network, generating strings of random bits, which provides a hardware realization of the bootstrap procedure described above. Such a realization is

(a) adaptive, so the memory content can be modified in terms of past history by a built in reinforcement learning (Barto, Sutton and Anderson, 1983) procedure,

(b) lightweight , since is interfaceability with different operating systems,

(c) fast, since the machinery is specially devised to learn and reproduce random variables, so providing attractive features for bootstrap estimators in a variety of situations, such as hostile environments or in rapidly varying ones.

The first characteristic of adaptability, noted earlier, is crucial to the whole approach, since the probabilities $\alpha_{x_1,x_2,...,x_{i-1}}(i)$ are unknown apriori, but have to be learnt by the machine, where learning here denotes the two operations of selecting a limited number of relevant probabilities and estimating their values.

An actual hardware implementation of this machine in VLSI technology has been realized at King's College London by Clarkson, Gorse, Taylor and Ng (1992). Thus, aiming at devising a realistic and efficient framework for our bootstrap implementation we will refer specifically to this machine to generate a set of limitations we will face up to in this paper. These constraints are:

i. absence of long therm memory

ii. limited amount of mass memory

and will be discussed later in this section. Before doing so we present the appropriate pRAM architecture to approach the problems above.

Figure 1 contains the hierarchical pRAM architecture by which every (strictly positive) ν-variate φ_α can be exactly simulated by a *sequentially* activated *linear chain* of ν pRAM units, in which each unit receives inputs from *all* and *only* the units preceding it in the sequential order. The i-th node contains the memory locations storing the parameters $\alpha_{x_1,x_2,...,x_{i-1}}(i)$, a built-in (pseudo) random number generator, which upon each call generates a random number U_i, and a built-in comparator, which, upon input $x_1, ..., x_{i-1}$ gives as output the integer part of $U_i + \alpha_{x_1,x_2,...,x_{i-1}}(i)$.

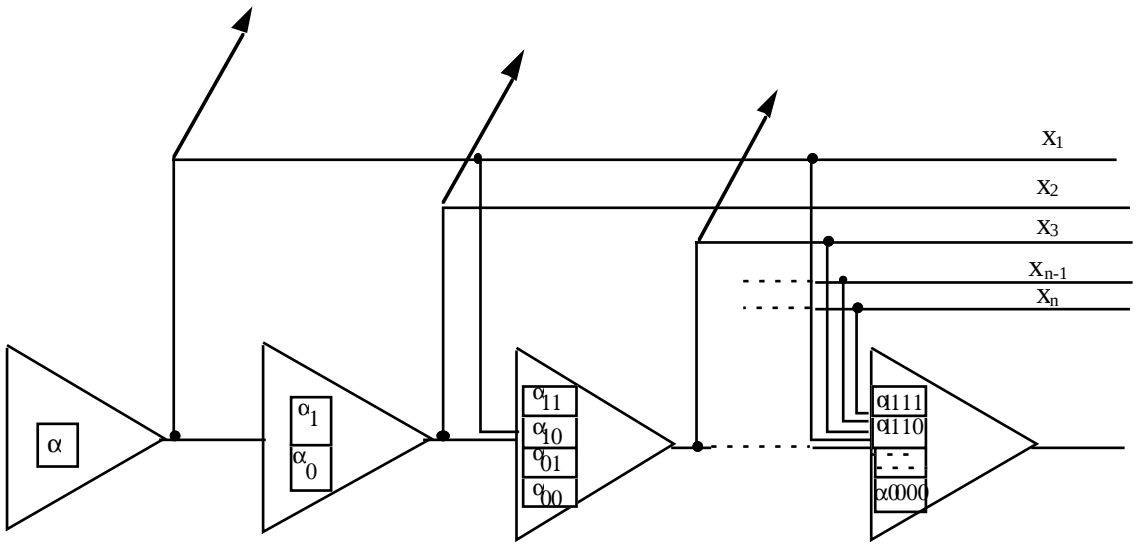

Figure 1.A fully connected hierarchical pRAM

The operation of such a chain can be described (in terms of the independent random variables $U_1,..., U_\nu$ with uniform distribution in (0,1) , generated by the built-in random number generators, and in terms of the action of the built-in comparators) according to the following scheme:

First unit:

No input;

1 stored parameter $\alpha(1)$;

Output $Y_1 = \text{Int}\,(U_1 + \alpha(1))$; where Int(x) indicates the integer part of x.

Second unit:

1 input (from the output of the first unit);

2 stored parameter $\alpha_{x_1}(2)$ (corresponding to $x_1=0$ and to $x_1=1$)

Output $Y_2 = \text{Integer}(\, U_2 + \alpha_{Y_1}(2)\,)$ (namely, the output Y_1 of the first unit decides which memory location of the second unit to access);

….

i-th unit:

i -1 inputs (from the outputs of the first i -1 units);

2^{i-1} stored parameters $\alpha_{x_1,x_2,...,x_{i-1}}(i)$ (corresponding to $x_h=0,1$, $h=1,...,i-1$)

Output $Y_i = \text{Integer}(\, U_i + \alpha_{Y_1,Y_2,...,Y_{i-1}}(i)\,)$

….

If the parameters α are estimated from a reasonably sized sample or for sparse distributions, plenty of estimators are never primed by any observation, so that we can shortcut some units and connections in our architecture.

Thus, in neural network terminology, the bootstrap translates into *learning* the family of distribution laws — their connection map and the Bernoulli parameters — and *generalizing* the observed sample.

Looking at usual, though not exhaustive, hardware constraints that can be met in many operational frameworks, here we face the following distinguishing features of the pRAM bootstrap implementation and devise a learning algorithm specially tailored to cope with our bootstrap generalization task.

1. The training facility of the machine updates the estimates of α at run time t on the basis of the current value $\mathbf{x}(t)$ without any memory of the past values of $\mathbf{x}$, the value of their number included. Thus the uniformly minimum variance unbiased estimator (Wilks, 1962) $\widehat{\alpha}_{x_1,x_2,...,x_{i-1}}(i)$ cannot be computed.

This means that the parameter $A_{x_1,x_2,...,x_{i-1}}(i)$ stored in the i-th unit in mean is more far than is $\widehat{\alpha}_{x_1,x_2,...,x_{i-1}}(i)$ from the target value $P(X_i=1|X_1=x_1;...; X_{i-1}=x_{i-1})$.

To reduce the damage coming from this deficiency, we propose to induce slight negative and symmetrical correlations between the elements of the sample $\{(Y_i \mid x_1,...,x_{i-1})_j, j=1...n\}$ generated by unit i, so that the bootstrap frequency $\tilde{S}_n=\frac{1}{n}\sum_{j=1}^{n}(Y_i|Y_1=x_1,...,Y_{i-1}=x_{i-1})_j$ – the unique statistic relevant to bootstrap procedures for bernoullian variables – has an asymptotic symmetrical distribution law centred on $P(X_i=1|X_1=x_1;...; X_{i-1}=x_{i-1})$ with a variance higher than, but possibly very close to that of $\hat{\alpha}_{x_1,x_2,...,x_{i-1}}(i)$. Note that this correlation remains inside the single bit generation and does not interfere with the probabilistical structure between the bits.

Actually, as we can be interested in the bootstrap analysis after receiving a limited number of **X** values and since $\tilde{S}_n$ may be a biased estimator in this case, in addition to the asymptotic variance we will study the decrease rate of its mean square error (MSE) as the *functional bootstrap quality profile.*

The separation between the single Bernoulli variables also holds during the search for the estimators. Indeed the incremental learning required by the memoryless constraint of our model consists in following a good trajectory in the parameter space from the point **A**, representing the set of the initial estimates, toward the point α corresponding to the actual distribution function F to be simulated.

Now, for infinitesimal changes d**A** of **A**, the relative entropy (Cover and Thomas, 1991) between the two models is half the Riemannian distance

$$I(\varphi_{A+dA},\varphi_A) = \frac{1}{2}\sum c_{(i\,;x_1,...,\,x_{i-1}),\,(j\,;x_1,...,\,x_{j-1})}\,dA_{x_1,...,\,x_{i-1}}(i)\,dA_{x_1,...,\,x_{j-1}}(j)$$

where c is the Fisher matrix (Rao, 1945; Amari, Kurata and Nagaoka, 1992):

$$c_{(i\,;x_1,...,\,x_{i-1}),\,(j;\,y_1,...,\,y_{j-1})} = \left(\frac{\partial^2 I(\varphi_\alpha,\,\varphi_A)}{\partial A_{x_1,...,\,x_{i-1}}(i)\;\partial A_{x_1,...,\,x_{j-1}}(j)}\right)_{\alpha\,=\,A}$$

A nice feature of our adopted parametrization of F is that the matrix c is diagonal (see Apolloni, de Falco and Taylor (1996) for further details). This implies that under entropic tracking strategies, the search for the best infintesimal step of our learning procedure can be done coordinate by coordinate in the parameter space, i.e. on each $A_{x_1,x_2,...,x_{i-1}}(i)$ separately.

Bootstrap methods directly use the simulated values g(s*) for studying the behaviour of g(s) in place of simulating g as a function of $\hat{F}(X)$, or equivalently of the set of parameters $\hat{\alpha}$, since it is easier. The same can be done with the functional bootstrap parameters, where those general properties deduced for the canonical bootstrap from the convergency of the frequency of the bootstrap variables to the frequency of the sample

variables remain true also for functional bootstrap, since a slight variant of the large number law (Feller,1960) applies also to $\widetilde{S}_n$.

To induce the correlation mentioned earlier, on the same conditioning string $(y_1,y_2\ldots, y_{i-1})$ each Y_i of our bootstrap sample is produced by a different generator. Let us underline this feature by putting, from now on, the subindex in brackets. This strategy is dual to the boosting approach recently much studied in the field of Computational Learning (Schapire, 1990). In both cases we could consider the single $Y_{[i]}$ as the response of an expert and we have to merge these responses to get the final decision. In our case the aim of the training is to update the capabilities of the experts, in the boosting approach we assume the expert capability fixed and change the values by which to weight their responses. In both cases, since the whole of the sample information is not available, the simple estimate of the bernoullian distribution parameter is also a relevant task by itself (Freund,1996).

2. The actually available pRAM chip is made up of 256 processors, the nodes of our linear chain, each one with fan-in 6 (pRAM-256 data sheet, 1994). This means that we can estimate only 2^6=64 conditional probabilities on each node. As mentioned before, the majority of the conditional probabilities are never observed, so that we can remove the corresponding connections and storing more cheaply the corresponding information as will explained later on. Nevertheless we can often be requested to sacrifice further non null probabilities in order to meet the fan-in constraint. A reduced version of this problem has been solved in a companion paper using an entropic criterion to optimize in this respect the pRAM layout (Apolloni, de Falco and Taylor, 1996). In this paper we recall the operational results and discuss their implementation in the — elementary but still representative — example of bootstrap analysis of a conditional entropy as a function of three random bits linked by the AND relationship.

The rest of the paper is organized as follows. Section 2 discusses the problem of obtaining a good on-line estimate of $\alpha_{x_1,x_2,\ldots,x_{i-1}}(i)$, in terms of asymptotic variance and MSE convergency rate, possibly using a run-time generated bootstrap sample, together with the original sample. Section 3 reviews this problem with respect to the functional bootstrap quality profile mentioned earlier. Section 4 summarizes the previous results in terms of properties of the bootstrap generator, with special attention being paid to actual implementability by the pRAM chip in feasible time, and gives the mentioned numerical example. Section 5 is devoted to discussion and open problems.

2. Learning a Bernoulli distribution law

We are faced with the following basic problem: given X(1), ..., X(m),.... a sequence of independent, identically distributed random variables, distributed according to the

Bernoulli law with unknown parameter $\theta=P(X=1)$, to draw a bootstrap sample $Y(1),...,Y(n)$ suitable for a low variance estimate of $F(X)$.
At each discrete time t, we are supposed to have:

- $X(t)$, the observation at time t;
- $A_k(t-1)$, a family of estimators of θ based on the observations up to time t-1, with $A_k(0)$ an *initial* estimator of θ, possibly synthesizing a previous sequence $X(-m),..,X(0)$;
- $Y_{[k]}(t)=Int\ (U_k+ A_k(t-1)\ )$, the output signal at time t of the model based on observations up to time t - 1. Here, each U_k is uniformly distributed in (0,1) and independent of the other U_i, of the behaviour of the X up to time t and of the all A_i up to time t-1.

Learning the bernoullian distribution just consists in updating $A_k(t-1)$ into $A_k(t)$.
We have examined various alternatives depending on:

i. the number of managed estimators: only one A(t) or many $A_k(t)$ variously interconnected.
ii. the variables affecting the updating: only X(t) or also one or more $Y_{[k]}(t)$.

The zero option considers only one A(t) constructed from A(t-1) and X(t).
The general form

$A(t) = (1-\varepsilon)\ A(t-1) + \varphi\ (X(t))$,

(since $\varphi(X(t))$ is a real function of a Boolean variable, with $0<\varepsilon<1$ in order to have an exponential decay of the initial "prejudice" A(0)) translates into

$A(t) = (1-\varepsilon)\ A(t-1) + \varepsilon_0 + \varepsilon_1\ X(t)$.

Requiring that the sequence A(t) of estimators of θ be asymptotically unbiased, namely that $\lim_{t\to\infty} E(A(t)) = \theta$, we have $\varepsilon_0 = 0$, and $\varepsilon = \varepsilon_1$. So the basic updating rule is:

$A(t) = (1-\varepsilon)\ A(t-1) + \varepsilon X(t)$. (1)

Elementary mathematical statistics provides the *best* choice in the above updating rule:
$\varepsilon = \varepsilon\ (t) = 1/t$. The ensuing iteration rule

$A(t) = (1-1/t\ )\ A(t-1) + 1/t\ X(t)$,

leads in fact, in a manner insensitive to A(0), to the estimator

$$A(t) = \frac{1}{t}\sum_{i=1}^{t} X(i),$$

namely to the uniformly minimum variance unbiased estimator of θ based on the sample $X(1),\ ....,\ X(t)$.

This abstract optimal choice has, however, two shortfalls in actual implementation by pRAM hardware:

a. It would require to keep track of "*local* time" (the number of times the given memory location has been updated);

b. It would require a *local* (time-dependent) reward parameter $\rho=1/t$ in the penalty-reward learning rule implemented by the pRAM hardware described in (Clarkson, Gorse, Taylor and Ng, 1992).

Neither feature can be implemented in a fast and simple way on the existing hardware.

On the contrary, the pRAM wiring allows the following incremental (reinforcement) learning rule:

$$\Delta\alpha = \rho\left((X - \alpha)\, r + \lambda\, (X^c - \alpha)\, p\right),$$

where $\Delta\alpha$ is the increment of the pRAM parameter α, X^c is the complement to 1 of X and ρ, r and λ are *constant* real parameters.

For $\rho=\varepsilon \in (0,1)$, $\lambda=1$, r=1 and p=0 we obtain exactly (1).

The asymptotic unbiasedness mentioned above comes from the fact that:

$$E[A(t)] = (1-\varepsilon)\ E[A(t-1)] + \varepsilon\, \theta, \qquad (2)$$

so that

$$\lim_{t\to\infty} E[A(t)] = \theta. \qquad (3)$$

Moreover

$$\mathrm{var}[A(t)] = (1 - \varepsilon)^2\, \mathrm{var}[A(t-1)] + \varepsilon^2\, \theta\left(1 - \theta\right)$$

so that, by a simple fixed point argument, the asymptotic mean square error can be computed to be:

$$\lim_{t\to\infty} \mathrm{var}[A(t)] = \frac{\varepsilon^2\, \theta\left(1 - \theta\right)}{1 - (1 - \varepsilon)^2} = \frac{\varepsilon\, \theta\left(1 - \theta\right)}{2 - \varepsilon} < \frac{\varepsilon}{4}. \qquad (4)$$

Thus, having maintained ε fixed bars the statistical consistency (Wilks,1962) of the estimator.

At the other extreme we consider a full option model constituted of a family of n statistics $A_k(t)$, whose updating rule is the following:

$$A_k(t) = (1-\varepsilon)A_k(t-1) + \varepsilon_0 X(t) + \varepsilon_1 Y_{[k]}(t) + \varepsilon_2 \sum_{j\neq k} Y_{[j]}(t)\ ;\ \varepsilon=\varepsilon_0+\varepsilon_1+(n-1)\varepsilon_2 \qquad (5)$$

and the obvious estimator:

$$\tilde{A}=\frac{1}{n}\sum_{k=1}^{n} A_k \qquad (6)$$

Two criteria are useful in order to compare these or other intermediate models:
i. asymptotic values of expectation and variance of the estimators.
ii. rate of decrease of the mean square error $E[(A-\theta)^2]$ (MSE[A]).

It is easy to realize that $E[\widetilde{A}(t)]$ has fixed point θ, provided that :

$$\varepsilon-\varepsilon_0-\varepsilon_1-(n-1)\varepsilon_2=0 \text{ and } |1-\varepsilon_0-n\varepsilon_2|<1, \text{ with } \varepsilon_0>0 \text{ and } \varepsilon_1,\varepsilon_2\geq 0 \qquad (7)$$

and equivalent conditions hold for any estimator obtained by *symmetrically* pruning some Y's.
It is easy to see that both the expected values E(t) of A(t) and $\widetilde{A}(t)$ have the same rate of convergency to θ for ε in A(t) equal to ε_0 in $\widetilde{A}(t)$, being determined by the relation:
$E(t) = (1- \varepsilon_0)E(t-1)+\varepsilon_0\theta$.
The addition of further random variables in the updating of the estimates, however, makes the decrease of the MSE slower. In fact:

$$E[(A(t)-\theta)^2] = (1-\varepsilon)^2E[(A(t-1)-\theta)^2] + \varepsilon^2\theta(1-\theta) \qquad (8)$$

and (see Appendix)

$$E[(\widetilde{A}(t)-\theta)^2] = (1-\varepsilon_0)^2E[(\widetilde{A}(t-1)-\theta)^2] + \varepsilon_0^2\theta(1-\theta)+ \qquad (9)$$
$$+[\varepsilon_1+(n-1)\varepsilon_2]^2\{E[\overline{Y}((t)-\theta)^2]-E[(\widetilde{A}(t-1)-\theta)^2]\}$$

where $\overline{Y}(t) = \frac{1}{n}\sum_{k=1}^{n} Y_{[k]}(t)$
giving the asymptotic value

$$E\left[\left(\widetilde{A}(\infty)-\theta\right)^2\right]= \frac{1}{n}\,\frac{\{n[1-(1-\varepsilon_0-n\varepsilon_2)^2]+(n-1)(\varepsilon_1-\varepsilon_2)^2\}E[(A_k(\infty)-\theta)^2]-(n-1)(\varepsilon_1-\varepsilon_2)^2\theta(1-\theta)}{1-(1-\varepsilon_0-n\varepsilon_2)^2} \qquad (10)$$

where, for any $k\in\{1,...,n\}$,

$$E\left[\left(A_k(\infty)-\theta\right)^2\right]= \theta(1-\theta)\left\{1- \frac{2\varepsilon_0(1-\varepsilon_0)\,[1-(1-\varepsilon_0-n\varepsilon_2)^2]}{\{1-[1-\varepsilon_0-(n-1)\varepsilon_2]^2+\varepsilon_1^2\}\{1-(1-\varepsilon_0)^2+\varepsilon_2[2(1-\varepsilon_0)-n\varepsilon_2]\}-2(n-1)\varepsilon_2^2\,[1-\varepsilon_0-(n-1)\varepsilon_2-\varepsilon_1][-\varepsilon_2)^2][2(1-\varepsilon_0)-n\varepsilon_2]}\right\}$$

Thus, $\widetilde{A}$ suffers greater asymptotic variance than A.
A non perfect balancing of the $Y_{[k]}$ contributions, as happens for the single A_k for instance, produces ever greater MSE values. Negative values of parameters ε_i, though

non disruptive of the convergency *per se*, might require a suitable clipping of the updated statistics to preserve the bernoullian law of the Y variables.
In conclusion, having as target an efficient estimate of θ, no better strategy emerges than the simple rule (1).

3. Generating bootstrap samples

In spite of the harmless conclusion of the previous section, if we look for optimizing the functional bootstrap quality profile claimed in the Introduction, a more efficient method for using the sample information is possible.
We analyze two strategies:
1. starting from A, we generate a bootstrap sample $Y_1,..,Y_n$, sampling from a Bernoulli variable of parameter A.
2. we take n different estimates A_k, according to rule (5), and produce analogously one $Y_{[k]}$ from each estimate.
We apply the same two criteria (asymptotic values and convergence speed) of section 2 on the bootstrap frequencies $S_n=\frac{1}{n}\sum_{k=1}^{n} Y_k$ and $\widetilde{S}_n=\frac{1}{n}\sum_{k=1}^{n} Y_{[k]}$ for comparing these strategies.

The involved computations are long rather than difficult. So we prefer, as already done for (9) and (10), to avoid giving all the details in the text but give just a streamlined version in the Appendix.

Both $E[S_n(t)]$ and $E[\widetilde{S}_n(t)]$ have the fixed point θ.
Concerning MSE, for the first frequency we have the dynamical system:

$$E[A(t)]=(1-\varepsilon)E[A(t-1)]+\varepsilon\theta$$

$$Var[A(t)]=(1-\varepsilon)^2 Var[A(t-1)]+\varepsilon^2\theta(1-\theta) \qquad (11)$$

$$MSE[S_n(t)]=(1-\varepsilon)^2 MSE[S_n(t-1)] + \frac{1}{n}\varepsilon(1-\varepsilon)(1-2\theta)\, E[S_n(t-1)] + \frac{\varepsilon\theta}{n}[1-\varepsilon\theta+(n-1)\varepsilon(1-\theta)]$$

giving the asymptotic value

$$E\left[\left(S_n(\infty)-\theta\right)^2\right]=\frac{1}{n}\theta(1-\theta)+\frac{n-1}{n}\frac{\varepsilon}{2-\varepsilon}\theta(1-\theta) \qquad (12)$$

which is made up of the expected value of the variance of $S_n(\infty)$ given $A(\infty)$ plus the variance of $A(\infty)$.
The second strategy aims at reducing the first term of the above sum, where A is substituted by $\widetilde{A}$, as follows.

Starting from the elementary fact that

$$E\left[\left(\left(\frac{1}{n}\sum_{k=1}^{n} Y_{[k]}\right)-\widetilde{A}\right)^2\right]=\frac{1}{n^2}E\left(\sum_{k=1}^{n}(Y_{[k]}-A_k)\right)^2=\frac{1}{n^2}\sum_{k=1}^{n}E(Y_{[k]}-A_k)^2+\sum_{\substack{i,j=1\\ i\neq j}}^{n}E(Y_{[i]}-A_i)(Y_{[j]}-A_j)$$

we choose the coefficient ε_2 in order to make the second summation in the last term of the equality chain negative. This is obtained by inducing negative correlations between the various statistics A_k through a negative value of ε_2.

Figure 2 shows the typical dependency of this summation on ε_2 and (13) gives the asymptotic value of MSE[$\widetilde{S}_n$]:

$$E\left[\left(\widetilde{S}_n(\infty)-\theta\right)^2\right]= \tag{13}$$

$$=\frac{1}{n}\,\frac{(n-1)[1-(1-\varepsilon_0-n\varepsilon_2)^2+(\varepsilon_1-\varepsilon_2)^2]E\left[\left(A_k(\infty)-\theta\right)^2\right]+[1-(1-\varepsilon_0-n\varepsilon_2)^2-(n-1)(\varepsilon_1-\varepsilon_2)^2]\theta(1-\theta)}{1-(1-\varepsilon_0-n\varepsilon_2)^2}$$

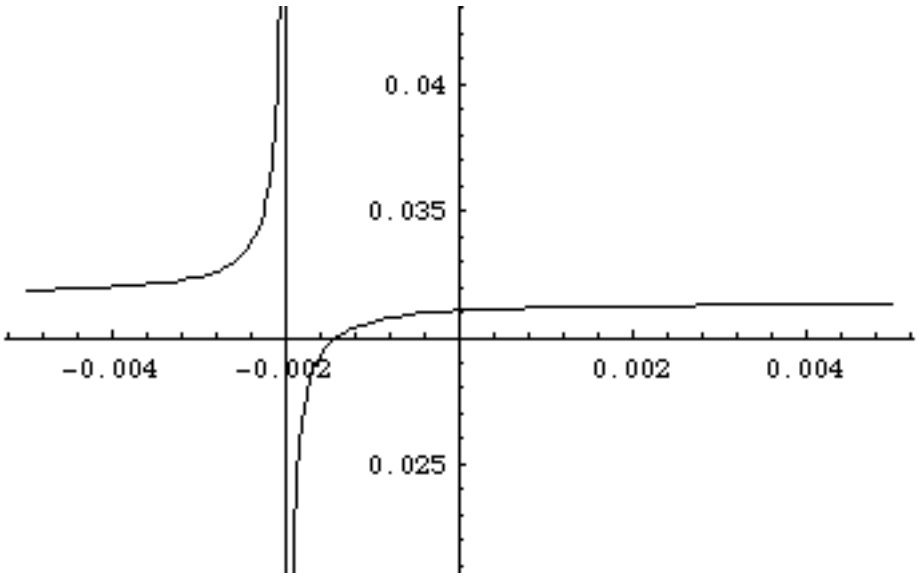

Figure 2. Covariance contribution to $E[(\widetilde{S}_n-\widetilde{A})^2]$ versus ε_2.
n= 5,θ =0.7, ε_0=0,01, ε_1=0,005.

The graph shows a tight hole towards negative values of the covariance contribution. On the left, conditions of non convergence are attained by the single estimate A_k at $\varepsilon_2 = -\frac{\varepsilon_0}{n}$, and $\varepsilon_2 \geq -\frac{\varepsilon_0}{n}$ applies from (7), so that $\varepsilon_2 > -\frac{\varepsilon_0}{n}$ is mandatory. Actually equality (13) holds for non negative ε_i. Now for $\varepsilon_2<0$ A_k might not yet result in a convex combination of numbers in (0,1) so that the clipping of the A_k mentioned earlier might be required to continue the learning process suitably. This expedient breaks the effect of the vertical

asymptote in $\frac{\varepsilon_0}{n}$ and induces some minor changes in the dynamics of $MSE[\widetilde{S}_n]$ that will be discussed later.

At the moment we disregard these changes and consider the analytical expression of $MSE[\widetilde{S}_n(\infty)]$ exactly as written in (13). Its behaviour with ε_2 strictly follows the one of the covariance contribution (see figure 3)

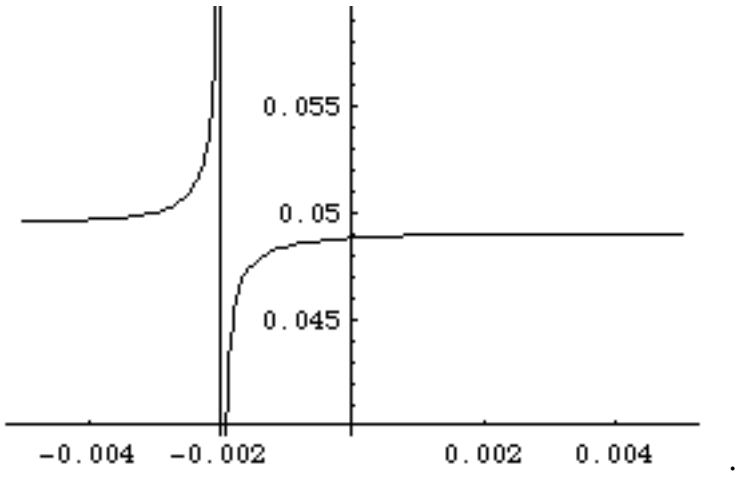

Figure 3. Asymptotic value of $MSE[\widetilde{S}_n]$ versus ε_2.
Same parameters like in figure 2

Of course, in the *range of feasibility* of ε_2 (see discussion at the end of this section) this parameter never goes below the variance of $\overline{X}$ and also of A, but might be a bit lower than $MSE[S_n(\infty)]$, where this difference substantially vanishes with increasing n. Figure 4 shows the relative gain with n for the parameters of the previous figures and $\varepsilon_2 = -0.95\frac{\varepsilon_0}{n}$.

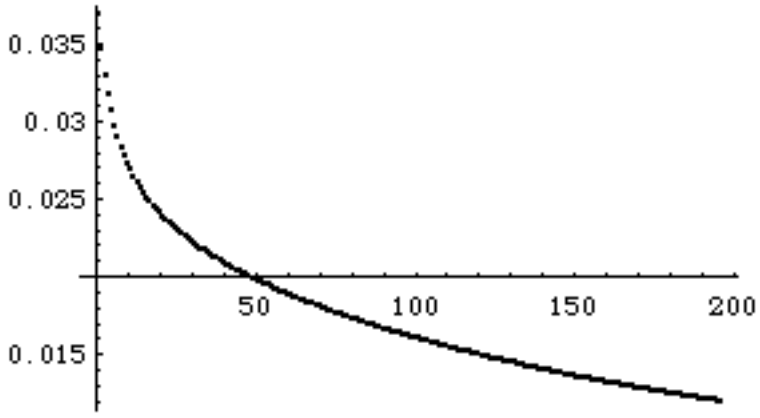

Figure 4. Graph of asymptotic $\frac{MSE[S_n] - MSE[\widetilde{S}_n]}{MSE[S_n]}$ in dependence of the bootstrap sample size.

$\varepsilon_2 = -0.95\frac{\varepsilon_0}{n}$, the other parameters like in figure 2

More interesting insights come from considering the dependency of MSE[$\widetilde{S}_n$] dynamics on the size t of the original sample:

$$MSE[\widetilde{S}_n(t)]=(1-\varepsilon_0)^2 MSE[\widetilde{S}_n(t-1)]+2(n-1)(1-\varepsilon)\varepsilon_2\{Var[\widetilde{A}(t-2)]-Var[\widetilde{S}_n(t-1)]\}+$$
$$+[2(1-\varepsilon_0)-n\varepsilon_2]\varepsilon_2 E[\widetilde{S}_n(t-1)[1-\widetilde{S}_n(t-1)]]+\frac{1}{n}(1-\varepsilon_0)\{E[\widetilde{S}_n(t-1)]-2\theta E[\widetilde{S}_n(t-1)]+\theta\}+\varepsilon_0^2\theta(1-\theta) \quad (14)$$

Comparing the trajectories of the MSE of $\widetilde{S}_n$ and S_n(see figure 5), we note that, in the specific parametrization, the first trajectory is always below the second, but this gap reduces above a given size of the original sample.

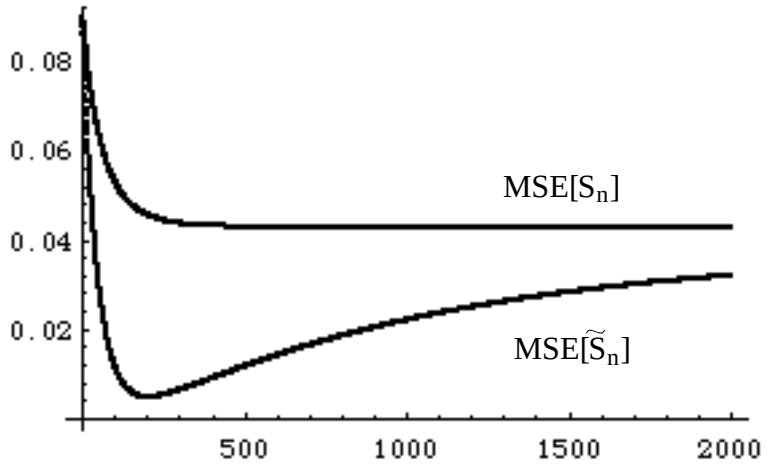

Figure 5. Graph of the MSE of S_n and $\widetilde{S}_n$ with the size of the original sample.
same parameters of figure 4

This corresponds to an instance of the *overfitting* phenomenon which occurs in inductive learning, for example dealing with neural networks (Hecht-Nielsen, 1989), but also with some symbolic learning algorithms, such as ADAboost (Freund and Schapire, 1995), and so on. Forcing the learning system to reproduce the training set too strongly reduces the system generalization capability for performing well after training. In our case the generalization quality is measured by the MSE of the estimate $\widetilde{S}_n$ of θ from the variables produced by the pRAMs after training. Concerning the second production strategy, too large an X sample, which induces an excessive updating of the A_k estimates from it, reduces the quality of the bootstrap sample.

Figure 6 supplies a clear explanation of this phenomenon in our case and suggests limiting the length of the training session. The negative covariance between the A_k increases as these estimates converge to nearby values.

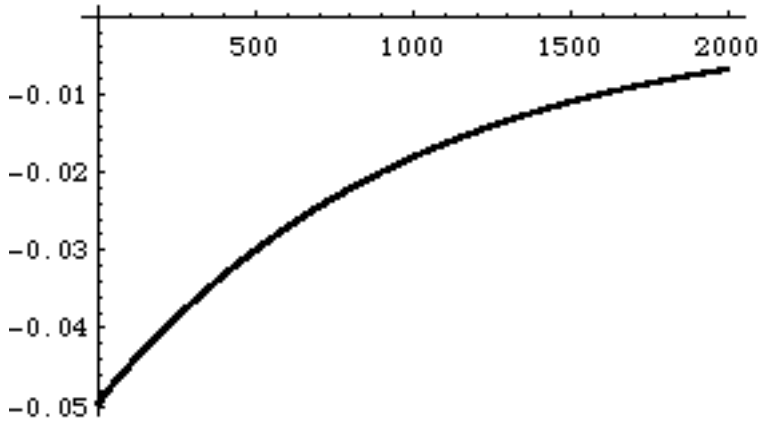

Figure 6. Covariance contribution to $E[(\widetilde{S}_n-\widetilde{A})^2]$ versus the size of the original sample.

Note that, as seen from figure 7, this overfitting phenomenon nullifies the benefit of the negative value of ε_2 on the MSE[$\widetilde{A}$] which asymptotically, as mentioned before, is driven, by the additive noise due to the $Y_{[i]}$, toward values higher than those corresponding to A.

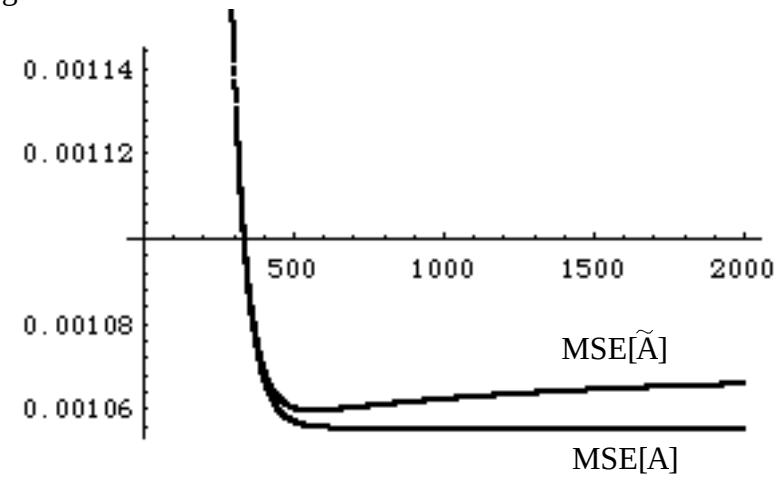

Figure 7.Graph of the MSE of A and $\widetilde{A}$ versus the size of the original sample.

Obviously, all these trajectories depend on the initial prejudices $A_k(0)$ about θ. In the cases shown in the figures we started from the fully biased values $A_k(0)\in\{0,1\}$ with around half of the parameters set to 0 and half to 1 (3 and 2 parameters). This choice allowed us to start with zero variance of both A_k's and $Y_{[k]}$'s. The equipartition of the assignments between 0 and 1, on one hand, accounts for our ignorance on θ, and on the other hand induces the highest variance of $\widetilde{S}_n$, a suitable starting point for a learning process.

The previous discussion left open two key problems: i. what is the influence of the A_k clipping, ii. what is the rule for an optimal joint selection of the ε_i and n. We are not able

to give definite answers to these questions, since the formal treatment is hard. However we can give some informal comments.

To give an idea of what happens when ε_2 is very close to $-\frac{\varepsilon_0}{n}$ from above, we drew the graph in figure 8 maintaining the value of MSE equal to its limiting value – three values within the five values of $Y_{[i]}$ fixed to 1 so that $\widetilde{S}_n = 0.6$ against $\theta=0.7$ and MSE=0.01 – each time the updating rule violated this threshold. The downward curvature is flattened, but the original shape of figure 5 is recovered as soon as the MSE dynamics brings this parameter back over the threshold. Thus, the least value of ε_2 for avoiding the A_k's overflowing is a random variable: the adoption of too small an ε_2 might produce undue expectation of the efficiency of our bootstrap sample; however we expect that the benefit of the induced negative covariances is also maintained in this case.

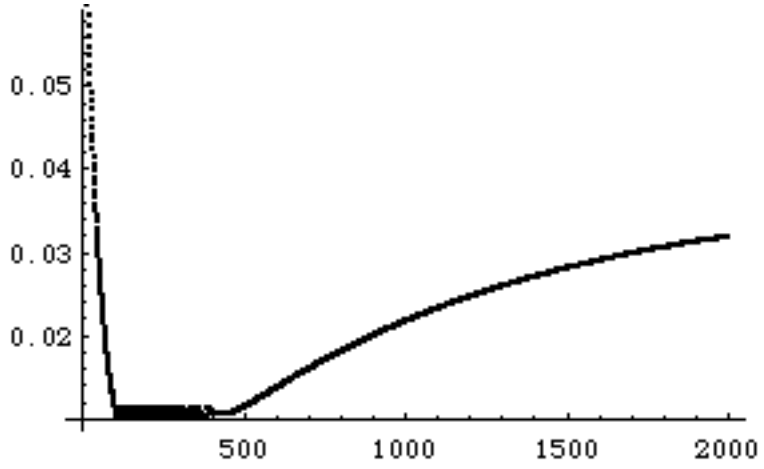

Figure 8. "True" shape of MSE of $\widetilde{S}_n$ for low values of ε_2.
Each time the consistency of the parameter is violated, its value is frozen at its boundary.

To give a feeling of the mutual influence of the three ε_i and the clipping intervention we drew the graph in figure 9 as a generalization of figure 8. Having ε_1 as curve parameter, for each ε_0, moving from $-\frac{\varepsilon_0}{n}$, we marked the first ε_2 for which no point of the MSE profile versus the original sample size t undergoes the limit value $\frac{\theta(1-\theta)}{t}$, i.e. the variance of $\overline{X}$ – here assumed as a more general threshold as in figure 8. The graph shows a negative linear dependency of ε_2 on ε_0, as was expected, and almost insensitivity to ε_1 in the neighbourhood of ε_0.

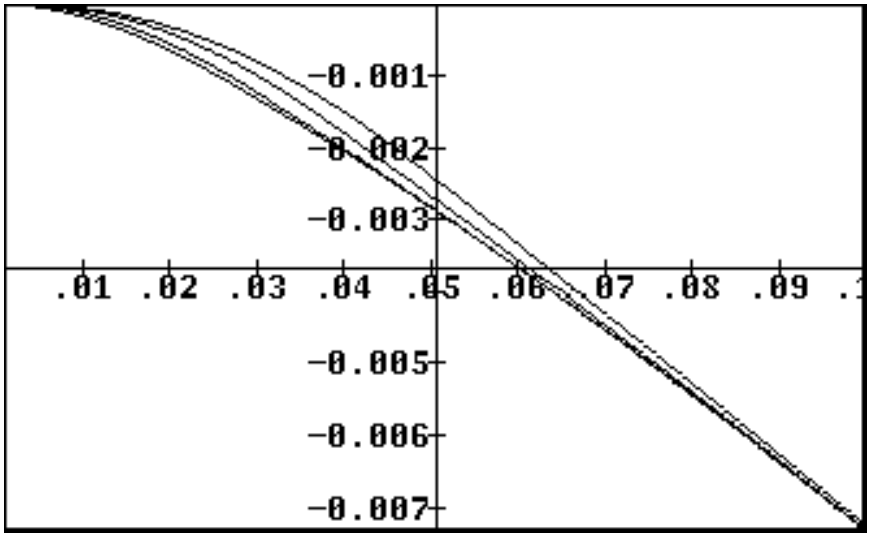

Figure 9. Dependency of *feasible* ε_2 on ε_0 and ε_1.
Vertical axis ε_2, horizontal axis ε_0, curve parameter ε_1
ε_1= 0.001,0.060,0, 0.120, 0.180 (from lower to upper curves), θ= 0.7, n=10.

The meaning of these curves can be inferred from figure 10. In particular, in figure 10.a we see that the suggested values of ε_2 reduce the downward curvature of the MSE profiles – it totally disappears for even larger ε_2 – where ε_1 has the pre-eminent role of laying down the MSE profile on the threshold limit and accelerating the descent. However, ε_1 can not increase too much, for instance up to three times ε_0, otherwise it either primes clipping again or requires higher ε_2, with an increasing minimum MSE.

Note that a more accurate choice of ε_2 in function of ε_1 will push the four downward peaks to the same value 0.021 (the actual value of $\frac{\theta(1-\theta)}{t}$ for the adopted parameters). This occurs in figure 10.b for other values of ε_2 and downward peak, with no changes, however, in the discussed general behaviour of MSE profiles. More in general, fine optimization of the learning parameters has to be carried out by a trial and error approach. This is not a heavy job, since the learning process for extreme values of the parameters is hard to study, but easy to monitor.
As a general direction, Fig. 10.b suggests working with low values of both ε_2 and ε_1 in order to exploit the intrinsic stability of the corresponding process. Indeed, in light of the considerations on figure 8, we expect that the emergence of clipping will bring back the curves (b) and (c) to the feasible region over the 0.021 threshold, while the others will continue rebounding over the threshold.

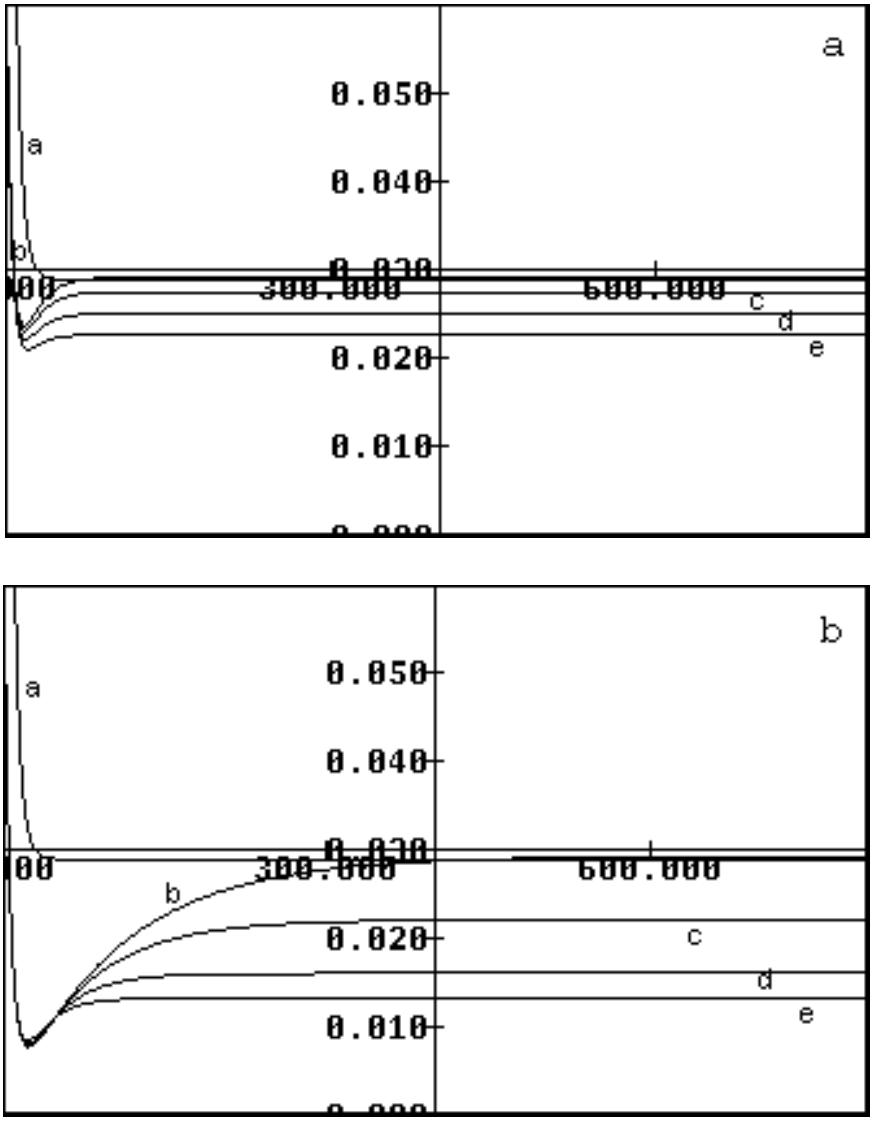

Figure 10 MSE[$\widetilde{S}_n$] profiles around *feasibility* points.
10.a MSE profiles in the feasible points, ε_0=0.08, ε_2=–0.0054
10.b same profiles with ε_2=–0.0076 (=-0.95ε_0/n) priming stability.
(a) reference MSE(S_n) profile, (b) ε_1 =0.001, (c) ε_1 =0.06, (d) ε_1 =0.12, (e) ε_1 =0.18
Horizontal axis: size of the original sample. θ= 0.7 n=10.

4. The actual bootstrap generator

Summarizing, we must capture information content from a sample of m discrete ν bits random variables $\mathbf{X}(1),...,\mathbf{X}(m)$ to produce a suitable set of n random variables $\mathbf{Y}_1,...,\mathbf{Y}_n$. The goal is to have a low MSE estimator of the cumulative distribution F of **X**, under the constraint that we have forgotten the original sample and the statistics on it have been computed on line at each presentation of the single **X**(i), where also the index i is unknown.

The goal is efficiently divided into computing, under the same constraint, low MSE estimators of the parameters of the 2^{ν}-1 bernoullian variables produced by a set of pRAM units to simulate **X**.

To compute the parameter of a single Bernoulli variable we propose a procedure which can be recalled through the following evocative *learning by gossips* model. We use n statistics which act like gossips hearing some news about a given fact θ. Each gossip i gives ear to the current news $X(t)$ about θ, and compares it with its own idea $Y_{[i]}(t)$ and those of the other gossips $Y_{[k]}(t)$ ($k \neq i$). In order to updating its inner conviction $A_i(t)$ about the facts, the model gives a positive weight to its previous experience $A_i(t-1)$ about the fact, current new and its own idea, while distrusting the other gossips whose ideas it gives negative weight.

Figure 11.a shows a magnified view of the ensemble evolution of the 50 gossips convictions A_k that are supposed initially equally distributed in the interval [0,1]. Figure 11.b shows, in addition, an actual bootstrap estimation of θ on the basis of other 50 replicas A_k, now starting with 25 replicas initialized to 0 and 25 initialized to 1, as suggested before. The two families converge toward the mean value θ =.7. S_n and $\widetilde{S}_n$ get close to this value, but $\widetilde{S}_n$, the dark bars, is less spread than S_n, the light bars, about in the first half thousand steps.

Figure 11.c reporting the MSE of the two estimators quantifies this greater initial efficiency of $\widetilde{S}_n$. Note that with this high number of replicas the asymptotic values are undistinguishable, but a little downward curvature persists for $\widetilde{S}_n$. Comparing figures b and c we observe that the MSE($\widetilde{S}_n$) gets very close to its asymptotic value before A_k reach their stable fluctuation around θ. This timing results from the induced negative covariances which compensate deviations of the replicas from θ, until replicas get nearby and covariances almost vanish. The covariance intervention explains also the greater efficiency of initially bipartite versus diffuse replicas, in spite of the lower convergency of the formers.

A last remark on the picture concerns the negative threshold for ε_2. Here we set ε_2 to $-0.99\varepsilon_0/n$, but, thanks to the number of gossips, no A_k clipping occurred.

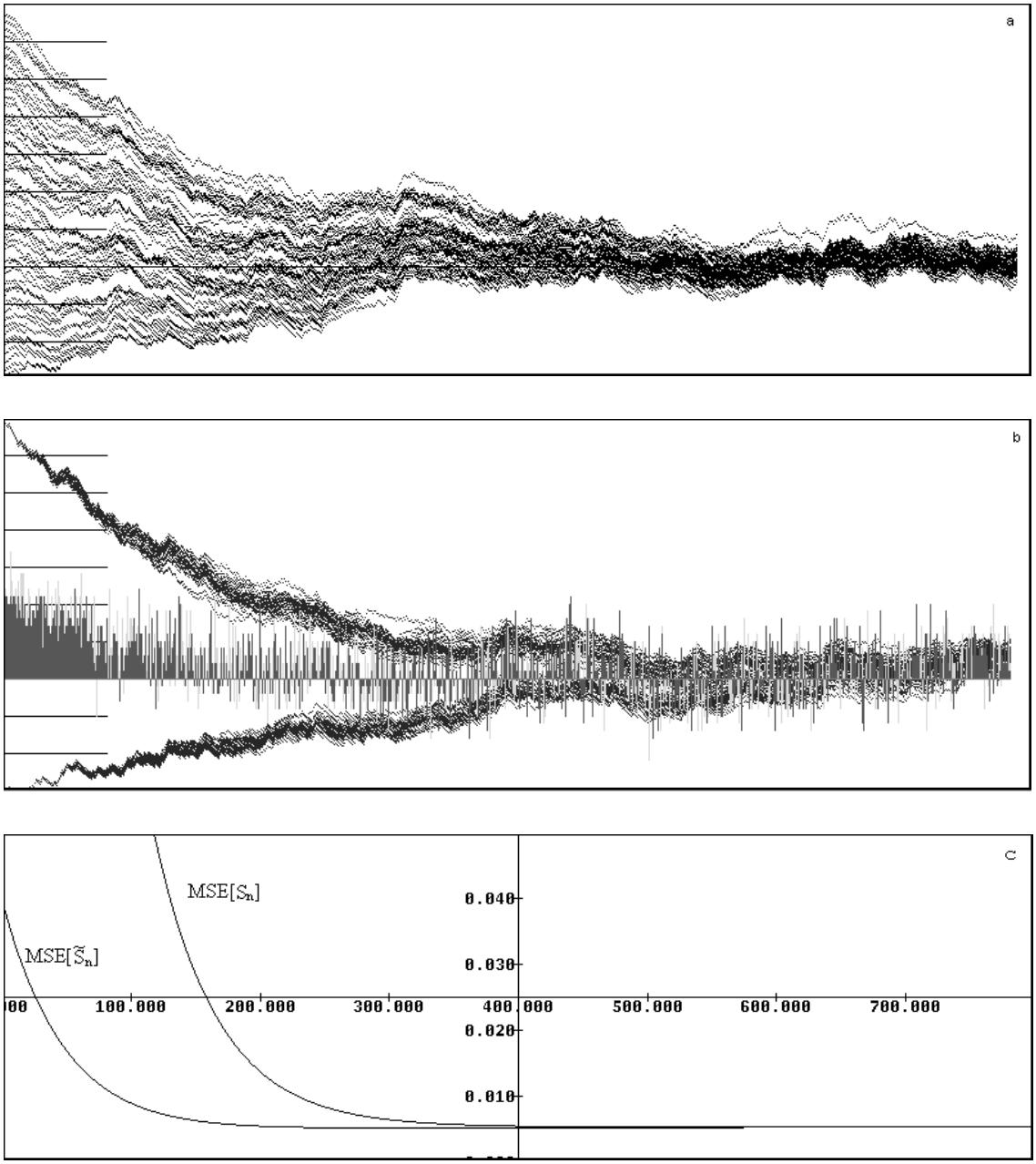

Figure 11. Convergence toward the unknown parameter.
11.a convergence of the gossips from homogeneous prejudices
11.b convergence of the gossips from bipartite extreme prejudices and of the bootstrap statistics
light bars = S_n, dark bars = $\tilde{S}_n$, continuous trajectories = A_k
11.c MSE profiles
Horizontal axis: size of the original sample

Note that the bits $Y_{[k]}$ coming from the replicas of a given pRAM node do not engage links between pRAM units. They accumulate in the reward-penalty signal sent to the unit by the external bus. So we can regard the *gossip* device as a wafer of pRAM chips, vertically interconnected, unit by unit to exchange reward-penalty terms.

On the horizontal plane, the different units should be connected in the hierarchical structure of figure 1. As mentioned before, to meet the fan-in constraint (each unit can receive inputs from only 6 other units) we must drastically prune the model connections.
In case of reduced parametrization, it is easy to show (Apolloni, de Falco and Taylor, 1996) that, whatever the set of the remaining available connections, the model $\beta^{(k)}$, with

$$\beta^{(k)}_{x_{k_1(i)},\ldots,\, x_{k_N(i)}}(i) = \alpha_{x_{k_1(i)},\ldots,\, x_{k_N(i)}}(i) \ ,$$

bounded fan-in N, $(k_1(i),\ k_2(i),\ldots..,k_N(i)) = k(i)$ the connection pointer of node i indicating the nodes which provide their outputs as inputs to node i and $k = (k(1),\ldots,k(\nu))$ the connection matrix, is the sole model which minimizes the relative entropy $I(\varphi_\alpha,\varphi_{\beta^{(k)}})$.

The task of pruning an already trained pRAM is sufficiently simple.
First of all we remove all the memory locations with parameter α equal either 0 or 1, and store elsewhere the corresponding defined bit. As the number of live locations remaining after the whole pruning is polynomial, the full table resulting from this first decimation can be compressed efficiently (i.e. in polynomial space and time).
The further decimation is less obvious. It corresponds to solve the hard and much studied topic of the search for optimal layout in a neural network (for three examples connected to our approach see Akaike (1984), Apolloni and Ronchini (1994) and Murata, Yoshizawa and Amari (1995)). In a companion paper (Apolloni, de Falco and Taylor, 1996) we discuss a local method still based on the constrained minimization of $I(\varphi_\alpha,\varphi_{\beta^{(k)}})$. From an operational point of view the method consists in comparing the efficiency of two candidate connection pointers k(i) and k'(i) of a given node i, the pointers of the other nodes being equal, through the difference

$$I_k - I_{k'} \equiv I\left(\varphi_\alpha,\ \varphi_{k,\,\beta^{(k)}}\right) - I\left(\varphi_\alpha,\ \varphi_{k',\,\beta^{(k')}}\right) \qquad (15)$$

If this difference is greater than zero we prefer pointer k'(i). The general sense of this rule is that the best pointers are those which make the output of each node depend most on its input.
The comparison is feasible since the computations involved are local in the pRAMs themselves. In fact,

$$I_k - I_{k'} = E_\alpha\left[H\left(Y_i \middle| k\right) - H\left(Y_i \middle| k'\right)\right] = I(Y_i;X_{k'_1(i)},\ldots X_{k'_N(i)}) - I(Y_i;X_{k_1(i)},\ldots X_{k_N(i)})$$

where $H(Y_i|k)$ is the entropy of the Bernoulli law simulated at node i, given the addresses that the observation $\mathbf{X}$ opens through the connection matrix k, and I is the mutual information between Y_i and the other connected bits. In other words, the above

rule reads as follows: we preserve connections which convey greatest mutual information (Cover and Thomas (1991)).

The actual lay-out problem is harder, since we need to determine at run time relevant bits and their suitable connection pointers. This is left here as an open technical problem, whose solution can be found within the techniques of on-line coding (Storer, 1988), familiar to Transmission engineers, and of on-line learning (Littlestone, 1991), developed in the last years by the Computational Learning community.

To give a numerical conclusion, in figure 12 we consider the paradigmatic instance of the bootstrap production of a three bits vector $\mathbf{Y}$ with Y_1 and Y_3 uniformly and independently distributed and Y_2=AND(Y_1, Y_3). These bits are obtained in output from a pRAM with three units chained in the same sequential order. We assume fan-in 1 on each unit; thus we have to disconnect node 3 from *one* of the nodes 1 or 2 (see figure 12.a). The two strategies of Section 3 read:

1)generate sample $\mathbf{Y}_1$,..., $\mathbf{Y}_n$, from a single set of estimated parameters A $=\{A^1,A^{2/1},A^{2/0}, A^{3/1},A^{3/0}\}$, being $A^{i/j}$ the estimate of $\alpha_{x_k}(i)$ for x_k=j, according to (1) for a fixed conditioning unit k.

2)generate sample $\mathbf{Y}_{[1]}$,..., $\mathbf{Y}_{[n]}$ from the wider set of parameters $A=\{A_1,..., A_n\}$, where $A_i=\{A_i^1,A_i^{2/1},A_i^{2/0},A_i^{3/1},A_i^{3/0}\}$ and $A_i^{j/h}$ is estimated according to (5).

The second strategy performs better in two respects:

i) run time selection of the connection pointer k(3). Figure 12.b shows a typical run of the learning process where the estimate $\widehat{H}(Y_3|k(3))$ better discriminates between the two candidates pointers k(3)=1 and k(3)=2 when the different gossip estimators A_i are involved in place of the single A. This is a side effect of the non linearity of the estimated parameter in A that overcomes the drawback depicted in Figure 7 of having MSE[$\widetilde{A}$]>MSE[A].

ii) bootstrap analysis. Focusing on the best connection pointer, figures 12.c and 12.d show that $\widehat{\widehat{H}}(Y_3|2)$ as a function of the bootstrap estimate $\widehat{\widehat{\alpha}}_{x_2}(3)$ of $\alpha_{x_2}(3)$ goes close to the parameter $H(Y_3|2)$ when $\widehat{\widehat{\alpha}}$ is a function of $\widetilde{S}_n$ than when $\widehat{\widehat{\alpha}}$ is a function of S_n. Furthermore, the picture shows that after sufficient training, our gossips generate samples which perform better than Efron bootstrap samples of the same size. To avoid giving a tricky meaning to this sentence, however, we must mention that in these experiments the training sample was larger than the bootstrap sample. The performance profile can be sketched through the following table.

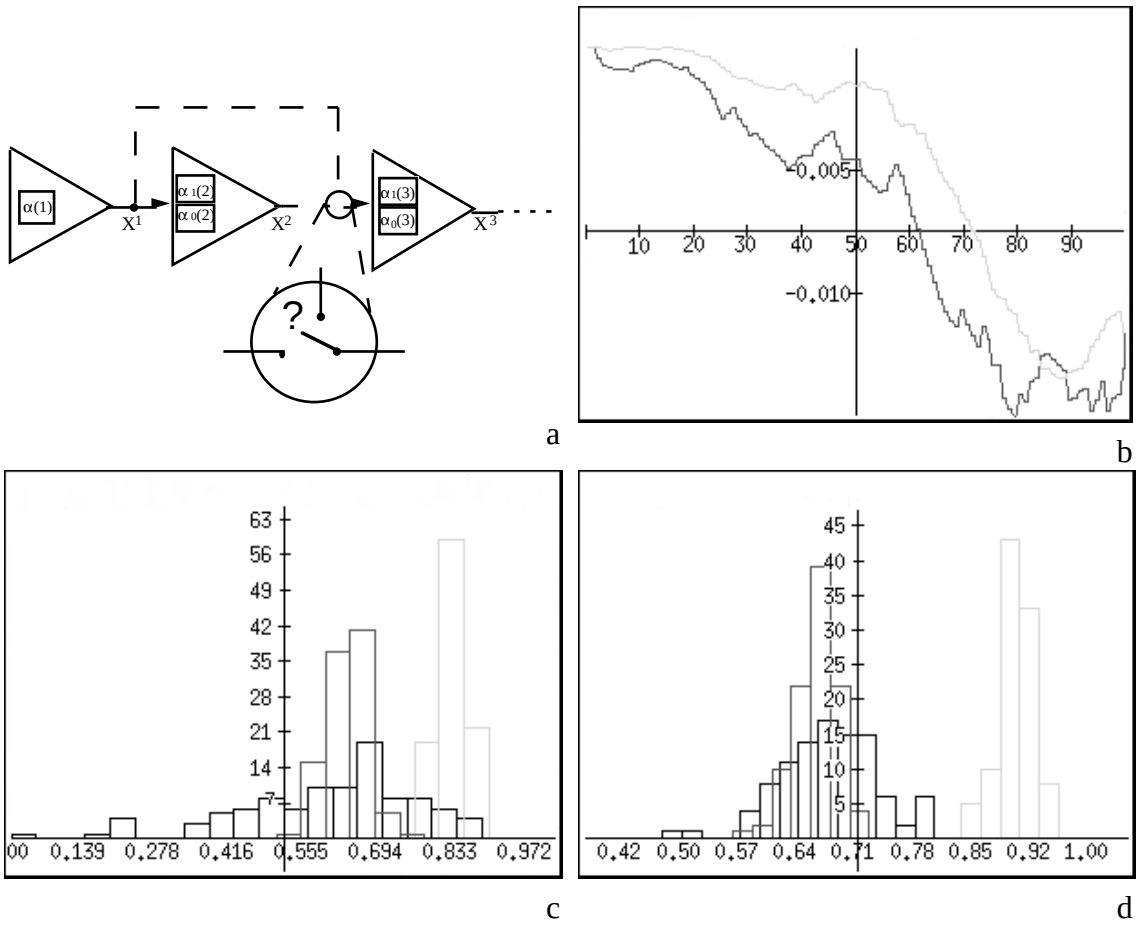

Figure 12. Learning to randomly generate an unknown boolean function under architectural constraints. Unknown function: X_2=AND(X_1, X_3).

12.a The architectural options. The third unit can be connected either to the first or to the second unit.

12.b Graph of

$$-\frac{1}{n}\sum_{i=1}^{n}\left\{\left[\hat{\alpha}_{x_2}(3)\log\left(\hat{\alpha}_{x_2}(3)\right)+(1-\hat{\alpha}_{x_2}(3)\log\left(1-\hat{\alpha}_{x_2}(3)\right)\right]-\left[\hat{\alpha}_{x_1}(3))\log\left(\hat{\alpha}_{x_1}(3)\right)+(1-\hat{\alpha}_{x_1}(3))\log\left(1-\hat{\alpha}_{x_1}(3)\right)\right]\right\}_i$$

as an estimate of $E_{\alpha}\left[H\left(Y_3\middle|2\right)-H\left(Y_3\middle|1\right)\right]$ with the size of the original sample.

Gray line refers to $\widehat{\alpha}$ = A as in (1), black line refers to $\widehat{\alpha}=\widetilde{A}$ as in (6).
The sums are computed on the basis of the last n accesses to the pRAM memory in the learning process. n =20.

12.c Histogram of $-\frac{1}{n}\sum_{i=1}^{n}\left[\hat{\hat{\alpha}}_{x_2}(3)\log\left(\hat{\hat{\alpha}}_{x_2}(3)\right)+(1-\hat{\hat{\alpha}}_{x_2}(3))\log\left(1-\hat{\hat{\alpha}}_{x_2}(3)\right)\right]_i$ as a double plug-in estimate of $H(Y_3|2)$

Black line refers to the Efron bootstrap estimate $\widehat{\widehat{\alpha}}$ of a, dark gray to $\widehat{\widehat{\alpha}}=\widetilde{S}_n$,

light gray line to $\widehat{\widehat{\alpha}}=S_n$. n =20; number of bootstrap replicas = 1000.

12.d The same as in 12.c, but with n=100.

sample size		Functional bootstrap		Efron Bootstrap
		with gossips	without gossips	
20	var	0.00248	0.00158	0.0221
	MSE	0.00497	0.0160	0.0282
	MSE'	0.00257	0.0273	0.0235
100	var	0.000725	0.000408	0.00325
	MSE	0.00129	0.0412	0.00357
	MSE'	0.000927	0.0451	0.00331

Table 1. Performance profiles of $\hat{\hat{H}}(Y_3|2)$ estimation.

$H(Y_3|2)= 0.689$;

$E[\hat{\hat{H}}(Y_3|2)] = 0.648$ for sample size 20, 0.679 for sample size 100

Sample size for training the functional bootstrap = 300. Number of bootstrap replicas=1000.

$MSE = E[(\hat{\hat{H}}(Y_3|2)-H(Y_3|2))^2]$; $MSE' = E[(\hat{\hat{H}}(Y_3|2)-E[\hat{\hat{H}}(Y_3|2)])^2]$

The on-line estimates of α's without gossips are still far from the true parameters after 300 learning steps. We realized that they need around 500 steps more to be suitably employed for simulating **X**. This is the cause of the large bias of $\hat{\hat{H}}(Y_3|2)$ based on these estimates. $\hat{\hat{H}}(Y_3|2)$ based on Efron bootstrap samples may perform even worse for the smaller sample size, since it does not enjoys the large (300 items) training sample. At n=100, on-line estimate without gossips is off-side and our method beats Efron's. However, we verified that a fairer comparison (300 observations of **X** both) favors the Efron method, as it was to be expected.

Lastly, we note that the benefit of gossips' samples in the bootstrap analysis emerges after larger training samples than in the connection pointer selection. This is due to the fact that in the first case $\hat{\hat{H}}[Y_3|2]$ is conditioned by the bootstrap variable Y_2 and not by the observed variable X_2 as in $H[Y_3|2]$

5. Conclusions

With the widespread use of computers in developing scientific disciplines, mathematical statistics methodologies are turning from the analytical study of simple functions of small sample data to numerical analysis of complex functions of, possibly huge, samples. The complexity of these functions is rewarded by a deeper exploitation of the information content of the sample data. Namely, we can devise statistics which are specialized with respect to the kind of applications and constraints one is dealing with. In particular we

explore a metaphor where Nature sent us from time to time some examples of a phenomenon that we experienced modifying each time some registers of our limited memory and forgetting the single data, and presently we require a large sample allowing us to infer some facts about the phenomenon.

This calls for new statistical perspectives to study and appreciate the quality of the computed statistics and a careful analysis of the feasibility of the resulting computations in terms of both supporting hardware and computing time.

This paper moves one step in this direction by proposing a bootstrap method for estimating the distribution law of a discrete variable in the above metaphor. The novelty rises from:

i. devising a functional bootstrap sampler which reproduces replicas of the original sample through a function implemented on a pRAM hardware reflecting the above memory limitations;

ii. using artificially correlated statistics to minimize the drawbacks driving from these limitations.

The sampling function is learnable in a feasible time and denotes interesting properties. The use of correlated statistics throws some new light on learning processes.

References

Akaike, H. (1985), "A new look at the statistical model identification", *IEEE Transactions on Automatic Control*, **19**, 716-723.

Amari, S., Kurata K. and Nagaoka H.(1992) "Information geometry of Boltzmann machines", *IEEE Transactions on neural networks*, **3**, 260-271.

Apolloni, B., de Falco D. and Taylor J.G (1996),"pRAM layout optimization", *Neural Networks*, to appear.

Apolloni, B.and Ronchini G.(1994), "Dynamic sizing of multilayer perceptrons" *Biological Cybernetic*s, **71**, 49-63.

Barto, A.G., Sutton R.S. and Anderson C.W. (1983), "Neuronlike Adaptive Elements that Can Solve Difficult Learning Control Problems", *IEEE Transactions on Systems, Man and Cybernetics*, **13**, 834-846

Clarkson, T.G., Gorse D. , Taylor J.G.and Ng C. K.(1992), "Learning probabilistic RAM nets using VLSI structures", *IEEE Transactions on computers*, **41**, 1552-1561.

Cover,T. and Thomas J. (1991), *Elements of information Theory*, John Wiley, N.Y.

Efron, B. and Tibshirani R.(1993), *An introduction to the Bootstrap*, Chapmann and Hall, Freeman, New York.

Feller,W. (1960), *An introduction to Probability Theory and its applications*, John Wiley, N.Y.

Freund, Y. (1996), "Predicting a Binary sequence almost as well as the optimal biased coin", proc. ACM-COLT 96, 89-98.

Freund,Y. and Schapire R.E. (1995),"A decision-theoretic generalization of on-line learning and an application to boosting", Proc. II European Conference on Computational Learning Theory, Barcellona, March 95

Hecht-Nielsen, R. (1989) "Theory of backpropagation neural networks", proc. IJCNN,**1**,593-606.

Littlestone, N.(1991)" On line learning of linear functions", CRL-UCSC-91-29

Murata, N., Yoshizawa S. and Amari S.(1995), "Network information criterion - Determining the number of hidden units for an artificial neural network model" *IEEE Transactions on Neural Networks*, **5**, 865-872.

"pRAM-256 data sheet", Department of Electronic and Electrical Engineering, King's College, London (1994).

Rao, C.R.(1945), "Information and accuracy attainable in the estimation of statistical parameters", *Bull. Calcutta Math. Soc.*, **54**, 81-91.

Schapire,R.E. (1990), "The strength of weak learnability", *Machine Learning*, **5**,197-227.

Storer, J.A. (1988) *Data compression methods and theory*, Computer Science Press.

Wilks, S.S.(1962) *Mathematical statistics*, John Wiley, N.Y.

APPENDIX

Hereafter we give some directions to compute formulas (9), (10),(11) and (13).

- Formula (9)

•Starting statements:

$A_k(t) = (1-\varepsilon)A_k(t-1) + \varepsilon_0 X(t) + \varepsilon_1 Y_k(t) + \varepsilon_2 \sum_{j\neq k} Y_j(t)$; $\varepsilon=\varepsilon_0+\varepsilon_1+(n-1)\varepsilon_2$;

$\widetilde{A}(t) = \frac{1}{n}\sum_{k=1}^{n} A_k(t)$; $Y_{[k]}(t)$= Integer$[A_k(t-1) + U_k]$; U_k uniform in [0,1]; $\overline{Y}(t) = \frac{1}{n}\sum_{k=1}^{n} Y_{[k]}(t)$.

•Recurrency relations:

1. $\widetilde{A}(t) = (1-\varepsilon)\widetilde{A}(t-1) + \varepsilon_0 X(t) + [\varepsilon_1 + (n-1)\varepsilon_2]\overline{Y}(t)$

   $E[\widetilde{A}(t)] = (1-\varepsilon)E[\widetilde{A}(t-1)] + \varepsilon_0\theta$

2. $Cov[\widetilde{A}(t-1),\overline{Y}(t)] = E[\widetilde{A}(t-1)\overline{Y}(t)] - E[\widetilde{A}(t-1)]E[\overline{Y}(t)] = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(E[A_i(t-1)Y_{[j]}(t)] - E[A_i(t-1)]^2\right)$

$= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(E[A_i(t-1)A_j(t-1)] - E[A_i(t-1)]^2\right) = \frac{1}{n^2}\sum_{i=1}^{n} Var[A_i(t-1)] + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j\neq i,\ j=1}^{n} Cov[A_i(t-1),A_j(t-1)]$

$= Var[\widetilde{A}(t-1)]$

3. $Var[\widetilde{A}(t)] = (1-\varepsilon_0)^2 Var[\widetilde{A}(t-1)] + \varepsilon_0^2\theta(1-\theta) + [\varepsilon_1+(n-1)\varepsilon_2]^2 Var[\overline{Y}(t)] +$
$+ 2(1-\varepsilon)\,[\varepsilon_1+(n-1)\varepsilon_2]^2 Cov[\widetilde{A}(t-1), \overline{Y}(t)]$
$= (1-\varepsilon_0)^2 Var[\widetilde{A}(t-1)] + \varepsilon_0^2\theta(1-\theta) + [\varepsilon_1+(n-1)\varepsilon_2]^2 \{Var[\overline{Y}(t)] - Var[\widetilde{A}(t-1)]\}$

•Final equation:

$E[(\widetilde{A}(t)-\theta)^2] = E[(\widetilde{A}(t)-E[\widetilde{A}(t)] + E[\widetilde{A}(t)]-\theta)^2]$
$=(1-\varepsilon_0)^2 E[(\widetilde{A}(t-1)-\theta)^2] + \varepsilon_0^2\theta(1-\theta) + [\varepsilon_1+(n-1)\varepsilon_2]^2 \{E[(\overline{Y}(t)-\theta)^2] - E[(\widetilde{A}(t-1)-\theta)^2]\}$

- Formula (10)

•Starting statements:

1. $E[(A_k(\infty)-\theta)^2] = E[(A_k(\infty))^2] - \theta^2$

•Recurrency relation:

1. $E[A_i(t)A_j(t)] = E\left[\left((1-\varepsilon)A_i(t-1) + \varepsilon_0 X(t) + \varepsilon_1 Y_{[i]}(t) + \varepsilon_2 \sum_{k\neq i,k=1}^{n} Y_{[k]}(t)\right)\right.$
$\left.\left((1-\varepsilon)A_j(t-1) + \varepsilon_0 X(t) + \varepsilon_1 Y_{[j]}(t) + \varepsilon_2 \sum_{k\neq j,k=1}^{n} Y_{[k]}(t)\right)\right.$

•Fixed point equations:

1. $E[(A_i(\infty))^2] - E[A_i(\infty)A_j(\infty)] = (1-\varepsilon_0-n\varepsilon_2)\,\{E[(A_i(\infty))^2] - E[A_i(\infty)A_j(\infty)]\} - (\varepsilon_1-\varepsilon_2)^2\{E[(A_i(\infty))^2] - \theta\}$

•Final equation:

$E\left[\left(A_k(\infty)-\theta\right)^2\right] = \theta(1-\theta)\left\{1 - \frac{2\varepsilon_0(1-\varepsilon_0)\,[1-(1-\varepsilon_0-n\varepsilon_2)^2]}{\{1-[1-\varepsilon_0-(n-1)\varepsilon_2]^2+\varepsilon_1^2\}\{1-(1-\varepsilon_0)^2+\varepsilon_2[2(1-\varepsilon_0)-n\varepsilon_2]\} - 2(n-1)\varepsilon_2^2\,[1-\varepsilon_0-(n-1)\varepsilon_2-\varepsilon_1][-\varepsilon_2)^2][2(1-\varepsilon_0)-n\varepsilon_2]}\right\}$

•Starting statements:

1. $Var[\widetilde{A}(t)] = \frac{1}{n^2}\left(\sum_{k=1}^{n} Var[A_k(t)] + \sum_{i=1}^{n}\sum_{j\neq i,j=1}^{n} Cov[A_i(t)A_j(t)]\right)$

•Fixed point equations:

1. $Cov[A_i(\infty)A_j(\infty)] = \frac{1}{(1-\varepsilon_0-n\varepsilon_2)^2}\left\{[(1-\varepsilon_0-n\varepsilon_2)^2 + (\varepsilon_1-\varepsilon_2)^2]E[(A_k(\infty))^2 - (\varepsilon_1-\varepsilon_2)^2\,\theta(1-\theta)\right\}$

•Final equation:

$E\left[\left(\widetilde{A}(\infty)-\theta\right)^2\right] = \left\{\frac{1}{n^2}\,nVar[A_i(\infty) + n(n-1)Cov[A_i(\infty)A_j(\infty)]\right\}$
$= \frac{1}{n}\,\frac{\{n[1-(1-\varepsilon_0-n\varepsilon_2)^2+(n-1)(\varepsilon_1-\varepsilon_2)^2\}E[(A_k(\infty)-\theta)^2] - (n-1)(\varepsilon_1-\varepsilon_2)^2\theta(1-\theta)}{1-(1-\varepsilon_0-n\varepsilon_2)^2}$

- Formula (11)

•Starting statements:

$A(t) = (1-\varepsilon)A(t-1) + \varepsilon X(t)$; $Y(t) = \text{Integer}[A(t-1)+U]$; $S_n(t)=\frac{1}{n}\sum_{k=1}^{n} Y_k(t)$, U uniform in [0,1].

•Recurrency relations:

1. $\text{Cov}[Y_i(t),Y_j(t)] = E[Y_i(t),Y_j(t)]-E[Y_i(t)]E[Y_j(t)] = E[A(t-1)^2]-E[A(t-1)]^2=\text{Var}[A(t-1)]$
2. $\text{Var}[Y_i(t)] = E[Y_i(t)^2] - E[Y_i(t)]^2 = E[A(t-1)]\{1-E[A(t)]\}$
3. $\text{Var}[A(t)]= (1-\varepsilon)^2\text{Var}[A(t-1)] + \varepsilon^2\theta(1-\theta)$
4. $\text{Var}[S_n(t)]=\frac{1}{n}E[A(t-1)]\{1-E[A(t-1)]\} + \frac{n-1}{n}\text{Var}[A(t-1)]$

•Final equation:

$\text{MSE}[S_n(t)] = \text{Var}[S_n(t)] + (E[S_n(t)]-\theta)^2= \frac{1}{n}E[A(t-1)](1-E[A(t-1)])+\frac{n-1}{n}\text{Var}[A(t-1)]+E\left[\left(A(t-1)-\theta\right)^2\right]$

$=(1-\varepsilon)^2\text{MSE}[S_n(t-1)] + \frac{1}{n}\varepsilon(1-\varepsilon)(1-2\theta)\, E[S_n(t-1)] + \frac{\varepsilon\theta}{n}[1-\varepsilon\theta+(n-1)\varepsilon(1-\theta)].$

- Formula (13)

•Starting statements:

$\widetilde{S}_n=\frac{1}{n}\sum_{k=1}^{n} Y_{[k]}.$

•Recurrency relations:

1. $\text{Var}[\widetilde{S}_n(t)] = \frac{1}{n^2}\left\{\sum_{i=1}^{n} \text{Var}[Y_{[i]}(t)]+\sum_{i=1}^{n}\sum_{j\neq i,j=1}^{n} \text{Cov}[Y_{[i]}(t)Y_{[j]}(t)]\right\}$

$= \frac{1}{n^2}\left\{\sum_{i=1}^{n} E[A_i(t-1)](1-E[A_i(t-1)])+\sum_{i=1}^{n}\sum_{j\neq i,j=1}^{n} \text{Cov}[A_i(t-1)A_j(t-1)]\right\}$

•Limit statements:

1. $E[(\widetilde{S}_n(\infty)-\theta)^2] = \lim_{t\to\infty} \text{Var}[\widetilde{S}_n(t)]$

•Final equation:

$$E\left[\left(\widetilde{S}_n(\infty)-\theta\right)^2\right]=\frac{1}{n}\,\frac{(n-1)[1-(1-\varepsilon_0-n\varepsilon_2)^2+(\varepsilon_1-\varepsilon_2)^2]E\left[\left(A_k(\infty)-\theta\right)^2\right] + [1-(1-\varepsilon_0-n\varepsilon_2)^2-(n-1)(\varepsilon_1-\varepsilon_2)^2]\theta(1-\theta)}{1-(1-\varepsilon_0-n\varepsilon_2)^2}$$

Formula (14)

•Recurrency relations:

1. $E[\widetilde{S}_n(t)] = \frac{1}{n}\sum_{k=1}^{n} E[A_k(t-1)] = (1-\varepsilon)E\left[\frac{1}{n}\sum_{k=1}^{n} A_k(t-2)\right] + \varepsilon_0\theta + \varepsilon_1 E[\widetilde{S}_n(t1)] + \varepsilon_2\frac{1}{n}\sum_{k=1}^{n}\sum_{h\neq k,h=1}^{n} E[A_h(t-2)]$

$= (1-\varepsilon)E[\widetilde{S}_n(t-1)] + \varepsilon_0\theta + \varepsilon_1 E[\widetilde{S}_n(t-1)] + (n-1)\varepsilon_2 E[\widetilde{S}_n(t-1)] = (1-\varepsilon_0)E[\widetilde{S}_n(t-1)] + \varepsilon_0\theta$

2. $Var[\widetilde{S}_n(t)] = E[\widetilde{S}_n(t)^2] - E[\widetilde{S}_n(t)]^2$

$= \frac{1}{n}E[\widetilde{S}_n(t)] + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j\neq i,j=1}^{n} E[A_i(t)A_j(t)] - (1-\varepsilon_0)^2 E[\widetilde{S}_n(t-1)]^2 - \varepsilon_0^2\theta^2 - 2\varepsilon_0(1-\varepsilon_0)\theta E[\widetilde{S}_n(t-1)]$

$= (1-\varepsilon_0)^2 Var[\widetilde{S}_n(t-1)] + 2(n-1)(1-\varepsilon)\varepsilon_2\{Var[\widetilde{A}(t-2)] - Var[\widetilde{S}_n(t-1)]\}$

$+ [2(1-\varepsilon_0) - n\varepsilon_2]\varepsilon_2 E[\widetilde{S}_n(t-1)(1-\widetilde{S}_n(t-1))] + \frac{1}{n}\varepsilon_0(1-\varepsilon_0)\{E[\widetilde{S}_n(t-1)] - 2\theta E[\widetilde{S}_n(t-1)] + \theta\} + \varepsilon_0^2\theta 1-\theta)$

•Final equation:

$MSE[\widetilde{S}_n(t)] = Var[\widetilde{S}_n(t)] + (E[\widetilde{S}_n(t)-\theta)^2 = Var[\widetilde{S}_n(t)] + (1-\varepsilon_0)^2(E[\widetilde{S}_n(t-1)-\theta)^2$

$= (1-\varepsilon_0)^2 MSE[\widetilde{S}_n(t-1)] + 2(n-1)(1-\varepsilon)\varepsilon_2\{Var[\widetilde{A}(t-2)] - Var[\widetilde{S}_n(t-1)]\} +$

$+ [2(1-\varepsilon_0) - n\varepsilon_2]\varepsilon_2 E[\widetilde{S}_n(t-1)[1-\widetilde{S}_n(t-1)]] + \frac{1}{n}(1-\varepsilon_0)\{E[\widetilde{S}_n(t-1)] - 2\theta E[\widetilde{S}_n(t-1)] + \theta\} + \varepsilon_0^2\theta(1-\theta)$