Simultaneous Estimation of Several Persentiles

Kimmo E. E. Raatikainen
University of Helsinki
Department of Computer Science
Tukholmankatu 2
SF-00250 Helsinki, Finland

Key words: discrete event simulation, percentile estimation

ABSTRACT

In many simulation studies the performance of the modeled system is characterized only by mean values of the performance indices. However, the percentiles are more informative in summarizing the performance. Several simultaneous percentiles, presented for example in the form of the boxplot, provide the analyst with better understanding of the system's behaviour.

The simultaneous estimation of several percentiles is a cumbersome task. The extended P^2-algorithm proposed in this study significantly reduces the computation of the estimation. The algorithm simultaneously estimates several percentiles without storing and sorting the observations. Empirical validation studies indicate that the algorithm can be used in simulation studies.

INTRODUCTION

Many simulation studies rely only on the mean values of the performance indices of the modeled system. However, if several percentiles can be estimated simultaneously, the analyst can get a much deeper insight to the system.

The estimation of several percentiles simultaneously is traditionally a computationally expensive task. The computational effort of the percentile estimation diminishes, if the distribution is characterized by a histogram. However, useful histograms require prior knowledge of the distribution and its percentiles.

The extended P^2-algorithm significantly reduces the computational effort of the percentile estimation. The algorithm simultaneously estimates several percentiles without storing and sorting the observations. The extended version, presented in this study, is a straightforward generalization of the original P^2-algorithm (Jain and Chlamtac 1985), which estimates a single percentile.

The reported experiments indicate that the extended P^2-algorithm provides estimates practically as good as the order statistics. The algorithm is easy to implement. There will be no difficulties to insert the implementation to simulation packages, which release the output sequences to user-defined analysis.

THE ALGORITHM

This section describes the ideas of the extended P^2-algorithm. The Appendix contains the algorithm in all details.

Instead of storing and sorting the observations, the P^2-algorithm updates the 'marker information' after each observation. Each marker has a height, a position and a desired position, and is associated with the estimation of a specific percentile.

Suppose that a marker estimates the percentile at point p, and a sample of n observations $(x_1, \ldots, x_n)$ is available. Let $x_{(j)}$ denote the jth observation in the ordered sample. The desired position of the marker is $1 + (n-1)p$. The algorithm tries to keep the actual position of the marker, m, close to $[1+(n-1)p]$ (the nearest integer to $1+(n-1)p$). The height approximates the value of $x_{(m)}$, and is thus an estimate of the percentile at point p.

Suppose that percentiles at points $p_1, \ldots, p_m$ are to be estimated. The algorithm consists of maintaining $m+2$ principal markers and $m+1$ middle markers. The principal markers estimate the minimum, the required percentiles, and the maximum. Each middle marker estimates the percentile at the point midway between two adjacent principal markers. Middle markers stabilize the adjustments of the marker heights.

The markers are updated, if necessary, after each observation. After n observations, the actual and desired positions of the markers at the minumum and at the maximum are 1 and n, respectively. Their heights are the minimum and the maximum values in the sample.

The updating of the actual and desired positions of the other markers is simple. For each marker, whose height is greater than the observation, the actual position is increased by one. In addition, if the actual position of a marker is now off to the left or to the right of its desired position by more than one, then the height and the actual position are adjusted. The marker height is adjusted using either a piecewise-parabolic or a piecewise-linear formula. The parabolic formula is the primary one. The linear formula is used only in situations where the parabolic adjustment would cause decreasing order of the marker heights.

The parabolic formula assumes that the curve passing through any three adjacent markers is of the form $q_i = a\,n_i^2 + b\,n_i + c$, where q_i is the height and n_i is the actual position of the marker. The movement of the marker's actual position to the left or to the right is always one position.

The parabolic adjustments are

$$q_i := q_i + \frac{d}{n_{i+1} - n_{i-1}}((n_i - n_{i-1} + d)\frac{q_{i+1} - q_i}{n_{i+1} - n_i}$$
$$+(n_{i+1} - n_1 - d)\frac{q_i - q_{i-1}}{n_i - n_{i-1}}) \text{ and } n_i := n_i + d \ .$$

where $d = 1$, if the movement is to the right; and $d = -1$, if the movement is to the left.

The linear adjustments are

$$q_i := q_i + d\frac{q_{i+d} - q_i}{n_{i+d} - n_i} \ , \quad \text{and} \quad n_i := n_i + d \ .$$

The initialization of the algorithm requires $2m+3$ observations. These observations are sorted and used as the initial heights of the markers, $q_i = x_{(i)}$. The actual positions initialize to $n_i = i$. This initialization is the simplest one. More sophisticated initializations are possible, but not used in this study. For example, the first n' observations are generated and sorted. The actual positions initialize to $n_i = [d_i]$, where d_i is the desired position of the marker i. Then it must be checked that the actual positions are strictly increasing. If not, some adjustments must be made. The heights then initialize to $q_i = x_{(n_i)}$.

EXPERIMENTS

The objective of the experiments was twofold. Results of the original algorithm (Jain and Chlamtac 1985) used only independent samples. The assumption of independent samples is extremely unrealistic in a simulation study. Hence, the behaviour of the algorithm, when the sample contains autocorrelation, was of interest. The second objective was to study the accuracy of the extended algorithm.

The sojourn times in an M/M/1-queue with three traffic intensities (0.5, 0.8, and 0.9) were simulated. With each intensity 101 independent sequences of length 32768 were generated. The percentiles at points 0.1; 0.25; 0.5; 0.75; and 0.9 were estimated by the extended P^2-algorithm and the order statistic. The estimates were evaluated using sequence lengths of 512, 2048, 8192, and 32768. The shorter sequences were used as parts of the longer ones.

The simulations were carried out using VAX-8600 running VMS 4.1. The random numbers were generated by the VAX-FORTRAN library function RAN. The initial seed of each sequence was taken from the wall-clock time. The queue was initialized using the steady-state distribution. The results are in Figure 1 and Table 1. In Figure 1, the ratios of the estimated and the exact percentiles are boxplotted. In the used form of boxplot, the distribution is summarized by five percentiles and some extremal values. In Table 1 are standardized square roots of mean square error. The standardization is done by dividing the square root of the mean square error by the exact value of the percentile.

In Figure 1, the upper labels, **P** and **O**, indicate the estimation method; **P** is for the extended P^2-algorithm, and **O** for the order statistics. The lower labels - 0.10; 0.25; 0.50; 0.75; and 0.90 - denote the points at which the percentiles are estimated. The horizontal scale of the plots is $\log_2$. The ends of the boxes denote the percentiles at points 0.25 and 0.75. The horizontal line inside the box denotes the median. The 'whiskers' reach the percentiles at points 0.05 and 0.95. The extreme values (five smallest and largest) are marked with horizontal lines. A star on the frame indicates that at least one extremal value is outside the plotted range.

The conclusions are self evident. The extended P^2-algorithm provides as good estimates as the order statistics. Neither the Mann-Whitney test nor the Kolmogorov-Smirnov two-smaple test can find any statistically significant difference between the methods. The confidences to equal shape (Kolmogorov-Smirnov) and to equal location (Mann-Whitney) of the distributions are in Table 2. Although the differences are not statistically significant, in four cases the null-hypothesis of equal shape or location is rejected at significance level 0.1. The Kolmogorov-Smirnov test reclaims that the shapes of the distributions are different with p=0.9, rho=0.5, N=512, and 32768. Using the

Table 1. Standardized square roots of mean square errors.

		Sequence length							
		512		2048		8192		32768	
rho	point	P	O	P	O	P	O	P	O
0.5	0.10	0.182	0.178	0.087	0.084	0.044	0.045	0.019	0.020
	0.25	0.156	0.148	0.065	0.065	0.039	0.039	0.016	0.016
	0.50	0.131	0.130	0.055	0.057	0.033	0.033	0.014	0.015
	0.75	0.135	0.127	0.061	0.059	0.034	0.033	0.014	0.014
	0.90	0.178	0.146	0.093	0.073	0.044	0.037	0.018	0.017
0.8	0.10	0.409	0.391	0.167	0.160	0.077	0.076	0.043	0.042
	0.25	0.359	0.356	0.154	0.154	0.074	0.073	0.042	0.042
	0.50	0.311	0.327	0.142	0.147	0.082	0.082	0.045	0.045
	0.75	0.351	0.339	0.180	0.170	0.092	0.089	0.051	0.051
	0.90	0.360	0.338	0.209	0.194	0.122	0.110	0.064	0.059
0.9	0.10	2.378	2.330	0.400	0.386	0.172	0.170	0.079	0.079
	0.25	1.280	1.255	0.432	0.445	0.172	0.172	0.083	0.083
	0.50	0.753	0.753	0.440	0.442	0.186	0.187	0.091	0.091
	0.75	0.496	0.502	0.385	0.374	0.190	0.184	0.100	0.100
	0.90	0.406	0.409	0.370	0.363	0.218	0.211	0.143	0.122

Table 2. Confidences to equal shape (KS) and to equal location (MW).

		Sequence length							
		512		2048		8192		32768	
rho	point	KS	MW	KS	MW	KS	MW	KS	MW
0.5	0.10	0.823	0.581	0.971	0.973	1.000	0.827	0.971	0.151
	0.25	0.823	0.648	0.914	0.781	1.000	0.990	0.914	0.966
	0.50	0.914	0.773	0.914	0.931	0.995	1.000	0.995	0.977
	0.75	0.291	0.162	0.378	0.366	0.914	0.720	0.971	0.923
	0.90	**0.083**	0.136	0.161	0.130	0.594	0.289	**0.057**	**0.037**
0.8	0.10	0.914	0.612	0.823	0.874	0.995	0.857	1.000	0.996
	0.25	0.823	0.704	0.971	0.971	1.000	0.948	1.000	0.956
	0.50	0.480	0.676	0.995	0.889	1.000	0.994	1.000	1.000
	0.75	0.480	0.497	0.823	0.602	0.971	0.577	0.971	0.729
	0.90	0.712	0.481	0.378	0.166	0.117	0.105	0.161	**0.079**
0.9	0.10	0.712	0.762	0.995	0.937	0.995	0.966	0.995	0.952
	0.25	0.914	0.638	0.995	0.910	0.995	0.914	1.000	0.950
	0.50	0.995	0.755	1.000	0.838	1.000	0.895	1.000	0.840
	0.75	0.971	0.769	0.995	0.863	0.712	0.582	0.914	0.638
	0.90	0.971	0.733	0.712	0.498	0.291	0.226	0.117	**0.046**

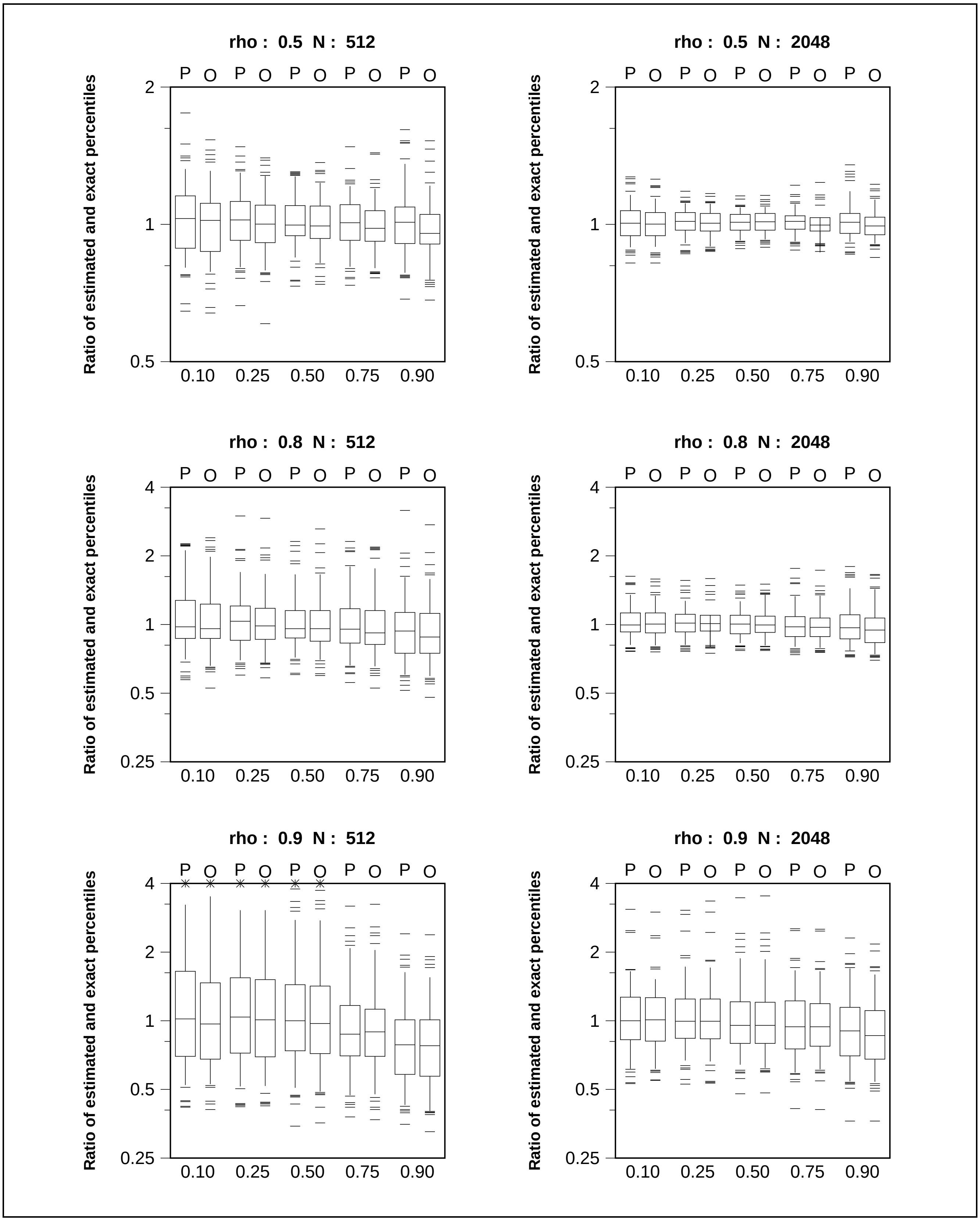

Figure 1. Boxplots of the ratios of the estimated and exact percentiles *(continued on next page)*.

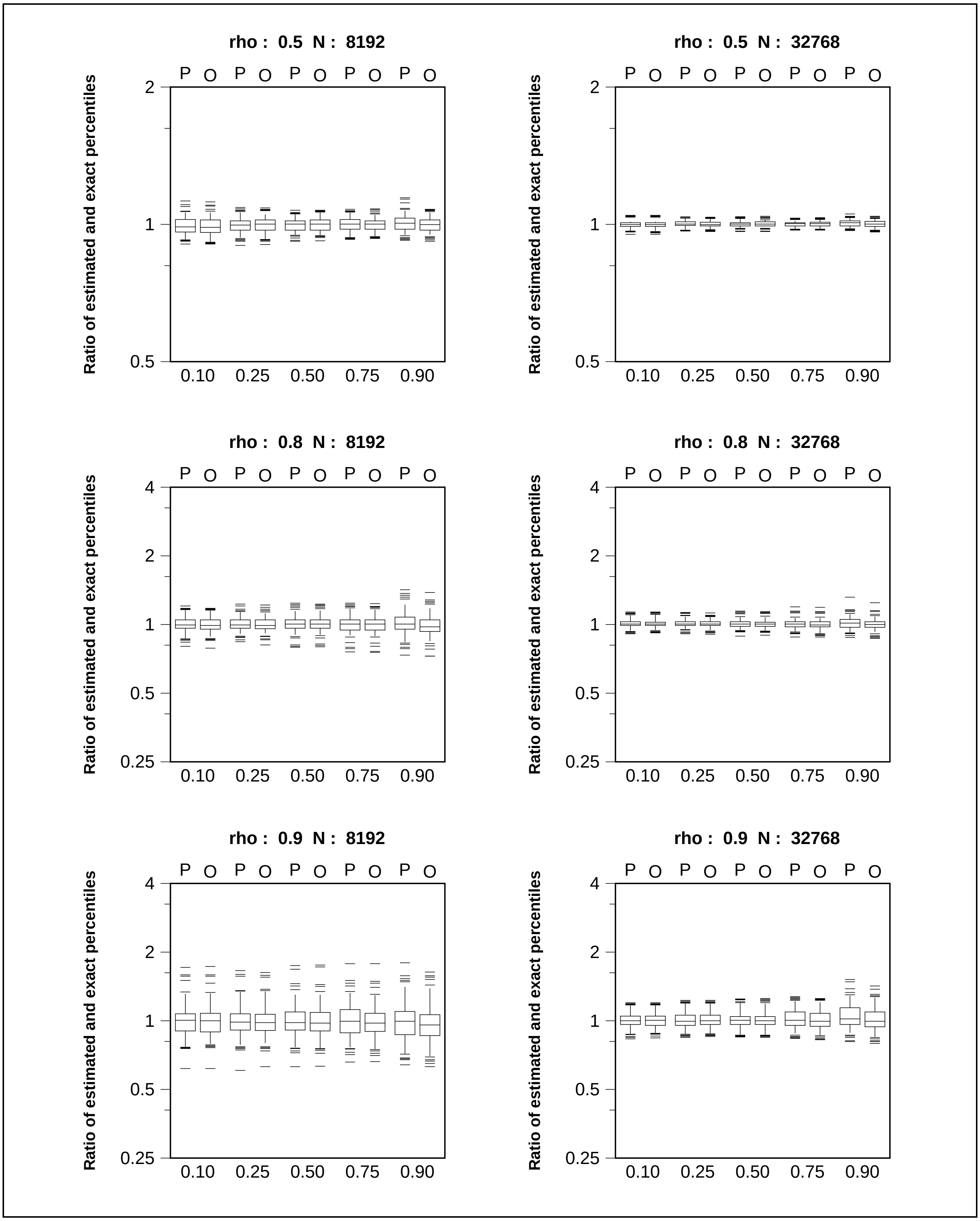

Figure 1. *(continued.)*

Mann-Whitney test the hypothesis of equal location is rejected with N=32768, p=0.9, rho=0.5, 0.8, and 0.9. The empirical distribution functions of these four cases are in Figure 2. They show that the extended P^2-algorithm produces slightly greater estimates. However, the differences are small and of no practical importance. In two cases, the medians of the P^2-estimates are closer to the true values and, also in two cases, they are further away from the true values.

The boxplots indicate that the P^2-algorithm produces estimates having slightly greater spread, when the percentile at point 0.9 is estimated. But the estimates are practically identical. Table 3, where the correlation coefficients between the estimates are given, confirms the practical identicality of the estimates.

The overall summary of the experiments is that the extended P^2-algorithm can be used to estimate the percentiles in simulation studies, where the simulated sequence is autocorrelated. The accuracy of the estimates is acceptable, when compared to the order statistics.

REFERENCES

Jain, R. and Chlamtac, I. 1985. "The P^2 Algorithm for Dynamic Calculation of Quantiles and Histograms Without Storing Observations." *Communications of the ACM* **28**, *no. 10* (Oct.): 1076–1085.

Table 3. Correlations between the estimates.

		Sequence length			
		512	2048	8192	32768
0.5	0.10	0.897	0.967	0.982	0.988
	0.25	0.957	0.974	0.991	0.994
	0.50	0.945	0.976	0.991	0.994
	0.75	0.852	0.922	0.981	0.990
	0.90	0.804	0.860	0.895	0.907
0.8	0.10	0.957	0.973	0.986	0.995
	0.25	0.985	0.990	0.995	0.999
	0.50	0.975	0.983	0.996	0.999
	0.75	0.957	0.976	0.981	0.991
	0.90	0.980	0.947	0.937	0.938
0.9	0.10	0.996	0.994	0.997	0.998
	0.25	0.994	0.996	0.998	0.999
	0.50	0.996	0.997	0.998	0.999
	0.75	0.992	0.987	0.978	0.987
	0.90	0.997	0.983	0.952	0.921

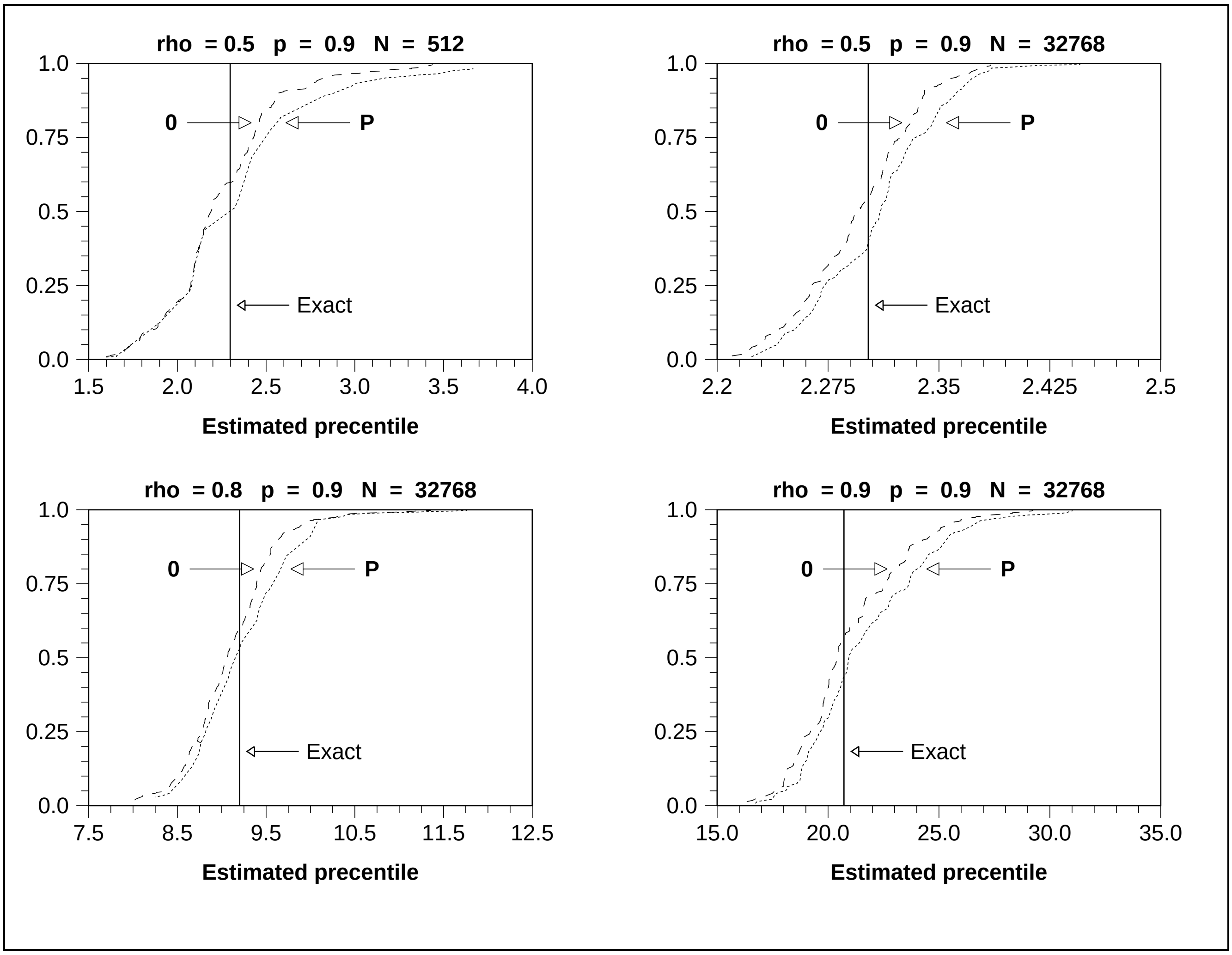

Figure 2. Emprical distribution functions.

APPENDIX

Extended P^2-Algorithm to estimate the percentiles of $x_1, x_2, \ldots$ at points $p_1, \ldots, p_m : 0 < p_1 < p_2 < \ldots < p_m < 1$

A. Initialization: Sort the first $2m+3$ observations and set

$q_i := x_{(i)}$,	$i = 1, \ldots, 2m+3$	*heights*
$n_i := i$,	$i = 1, \ldots, 2m+3$	*actual positions*
$f_1 := 0, \quad f_{2m+3} := 1$,		*increments of*
$f_{2i+1} := p_i$,	$i = 1, \ldots, m$	*desired*
$f_{2i} := (f_{2i-1} + f_{2i+1})/2$,	$i = 1, \ldots, m+1$	*positions*
$d_i := 1 + 2(m+1)f_i$,	$i = 1, \ldots, 2m+3$	*desired positions*

B. For each subsequent observation x perform the following steps

1. Find cell k such that $q_k \leq x < q_{k+1}$
   if $x < q_1$ **then** $k := 1$ and $q_1 := x$
   if $x \geq q_{2m+3}$ **then** $k := 2m+2$ and $q_{2m+3} := x$
2. Increase actual positions of markers $k+1, \ldots, 2m+3$
   $n_i := n_i + 1, \quad i = k+1, \ldots, 2m+3$
3. Increase desired positions of all markers
   $d_i := d_i + f_i, \quad i = 1, \ldots, 2m+3$
4. Adjust heights and actual positions of markers $2, \ldots, 2m+2$, if necessary

```
for i = 2 to 2m + 2 do
    begin d := d_i − n_i              offset to desired position
          dp := n_{i+1} − n_i         offset to next position
          dm := n_{i−1} − n_i         offset to previous
                                      position
          qp := (q_{i+1} − q_i)/dp
          qm := (q_{i−1} − q_i)/dm
          if (d ≥ 1 and dp > 1) then
          begin qt := q_i + ((1 − dm) * qp
                      +(dp − 1) * qm)/(dp − dm)
                if q_{i−1} < qt < q_{i+1} then q_i := qt
                else q_i := q_i + qp
                n_i := n_i + 1
          elseif (d ≤ −1 and dm < −1) then
          begin qt := q_i − ((1 + dp) * qm
                      −(dm + 1) * qp)/(dp − dm)
                if q_{i−1} < qt < q_{i+1} then q_i := qt
                else q_i := q_i − qm
                n_i := n_i − 1
enddo
```

C. Estimates: $\hat{q}_i = q_{1+2i}, i = 1, \ldots, m$