Challenges and Recent Advances in Distributed Predictive Coding

Ankur Saxena and Kenneth Rose
Dept. of Electrical and Computer Engineering
University of California
Santa Barbara, CA 93106
Email:{ankur, rose}@ece.ucsb.edu

(Invited Paper)

Abstract— **Distributed coding of correlated sources with memory poses a number of considerable challenges that threaten practical applications, particularly (but not only) in the context of sensor networks. This problem is strongly motivated by the obvious observation that most common sources exhibit temporal correlations that may be at least as important as spatial or inter-source correlations. This paper presents an analysis of the underlying tradeoffs, paradigms for coding systems, and approaches for distributed predictive coder design optimization. Motivated by practical limitations on both complexity and delay (especially for dense sensor networks) the focus here is on predictive coding. From the source coding perspective, the most basic tradeoff (and difficulty) is due to conflicts that arise between distributed coding and prediction, wherein 'standard' distributed quantization of the prediction errors, if coupled with imposition of zero decoder drift, would drastically compromise the predictor performance and hence the ability to exploit temporal correlations. Another challenge arises from instabilities in the design of closed loop predictors, whose impact has been observed in the past, but is greatly exacerbated in the case of distributed coding. The main contribution focuses on the tradeoffs encountered within a more general paradigm where decoder drift is allowable or unavoidable, and must be effectively accounted for and controlled. We briefly review our earlier results on which we build to derive an overall design optimization method that avoids the pitfalls of naive distributed predictive quantization and produces an optimized low complexity and low delay coding system.**

I. Introduction

The theoretical foundation of the field of distributed source coding (DSC) was laid in the early seventies with the seminal work of Slepian and Wolf [1], which was followed shortly afterwards by Wyner and Ziv in [2]. A considerable revival of interest, with focus on practical code design, has been witnessed since the late nineties, with the work of Pradhan and Ramchandran [3] as a notable precursor. The field eventually saw the emergence of various distributed coding techniques, mostly with an eye towards sensor networks (see e.g.,[4], [5]). The basic setting in DSC involves multiple correlated sources (e.g. data collected by spatially distributed sensors) transmitting information to a fusion center without any inter-communication amongst themselves (see Fig. 1). The main objective in DSC is to exploit inter-source (e.g. spatial) correlations despite the fact that each sensor source is encoded without access to other sources. The only information available to a source encoder about other sources involves their joint statistics (e.g., extracted from training set data).

The main motivation for the work presented here springs from the fact that most correlated sources in the real-world are sources with memory, i.e., they exhibit temporal correlations. In particular, sensor networks will often produce data whose time correlations are at least as important as their spatial (inter-source) correlations. Examples range from simple sensors monitoring slowly varying physical quantities such as temperature, to the extreme of video cameras collecting highly correlated frame sequences.

Realizing the prevalence of sources with memory and the importance of exploiting both temporal and inter-source correlation, we target the problem in the representative setting of distributed *predictive* coding (DPC) systems. Given the historical focus on inter-source correlations, most existing DSC work naturally addressed memoryless sources where one need not worry about temporal correlations. The implicit assumption may have been that predictive coding per se is a largely solved problem, and that extending DSC results to incorporate prediction would require a straightforward integration phase. (An alternative argument may appeal to handling long blocks of source data, e.g., by vector quantization, to exploit time correlations but the cost in delay and complexity may be considerable.) We shall, however, see that the generalization from DSC to DPC is highly non-trivial due to conflicting objectives of distributed coding versus efficient prediction in DPC. In other words, optimal distributed coding (in terms of current reconstruction quality) may severely compromise the prediction loop at each source encoder. We therefore propose to investigate new DPC system paradigms and methods to optimize their design.

There exists a channel coding "camp" of DSC research (see e.g.,[6], [7]), where long delays may be employed to achieve the desired performance, (e.g. using turbo/LDPC like codes, see [8], [9]). The other DSC research direction builds directly on source coding methodologies. Algorithms for distributed vector quantizer design have been proposed in [10], [11], [12] with major or exclusive focus on memoryless sources. The source coding perspective will be most relevant to us here.

1-4244-1564-0/07/$25.00 ©2007 IEEE

The temporal correlations within a source can be accounted by blocking sources into large vectors, but such a scheme will have high complexity and will be extremely sensitive to initialization and poor local optima [13], [14], [15]. Motivated by these observations, a notable approach to predictive coding of correlated sources have been proposed in [14] where a uniform quantization grid was imposed on the product space (across sources) of prediction errors, on which the main support of the joint distribution was identified and a DSC code devised. The emphasis in that paper's results was on the design of optimal predictor filters in such distributed setting and on how they deviate from the case of non-distributed predictive coding. Also in [16], an algorithm was given for predictive coding of correlated sources where different components (encoder and decoders) were designed. However in both the previous settings, neither the optimality of the algorithms was proven nor the system can be guaranteed to be drift-free for all values of inter-source/temporal correlations. We have proposed an optimal algorithm with 'zero-drift' for distributed predictive coding in [17]. Our 'zero-drift' algorithm subsumes as special extreme cases (a) separate predictive coding of sources and (b)memoryless distributed coding. The main contribution of this paper, is a 'controlled drift' approach that subsumes the zero-drift approach as a special case that emerges whenever the impact of potential drift overwhelms the benefits in improved prediction.

Another design difficulty whose origins are in standard predictive quantizer design [18] is exacerbated in the distributed setting. On the one hand, open-loop design is simple and stable but the quantizer is mismatched with the true prediction error statistics (as the system eventually operates in closed loop.) On the other hand, if a distributed quantizer is designed in closed-loop, the effects of quantizer modifications are unpredictable as quantization errors are fed back through the prediction loop and can build up. Hence the procedure is unstable and may not converge. The effect is greatly exacerbated in the case of DPC. This will be explained in more details in Sec. II-E To circumvent these difficulties we use the technique of asymptotic closed loop (ACL) {[19], [20]} which we rederive for DPC system design. Within the ACL framework, the design is effectively in open-loop within iterations (eliminating issues of error buildup through the prediction loop), while ensuring that asymptotically, the prediction error statistics converge to closed loop statistics. In other words, the prediction loop is essentially closed asymptotically.

The organization of the rest of the paper is as follows. In Section II, we state the problem formally, introduce notation, specify the components of the DPC system in zero-drift scheme and the need of ACL approach for DPC design. Section III motivates and specifies the components in controlled-drift based DPC design. Section III-B gives a brief overview of ACL for signle source design and presents the iterative 'controlled-drift' algorithm for DPC design. Simulation results are summarized in Section IV.

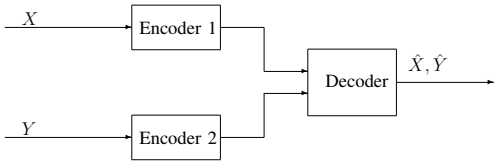

Fig. 1. Distributed coding of two correlated sources

II. Distributed Predictive Coding

A. Preliminaries

Consider the simplest distributed source coding scenario of Fig. 1 where for brevity, but without loss of generality, we restrict the presentation to two sources. Here X and Y are two continuous amplitude, correlated (scalar or vector) sources with memory. The two source encoders (Fig. 1) compress and transmit source information at rates R_1 and R_2 bits per sample, respectively to the central unit (joint decoder). The objective is to minimize the following expected distortion:

$$D_{net} = E\{\alpha d(X, \hat{X}) + (1-\alpha) d(Y, \hat{Y})\}, \qquad (1)$$

where $d(\cdot,\cdot)$ is an appropriately defined distortion measure, $\hat{X}$ and $\hat{Y}$ are the reconstruction values for X and Y, respectively and $\alpha \in [0,1]$ is a weighing factor to govern the relative importance of the sources at the decoder.

We employ (say, linear) prediction to exploit temporal redundancies within the sources X and Y respectively. The prediction errors e_x (for X) and e_y (for Y) are likely correlated. Therefore, instead of the standard predictive quantizer, a distributed quantizer needs to be designed to exploit inter-source correlations. Since additional information (from correlated source) about a source can be utilized, the encoder and decoder reconstruction of the prediction errors can be different. We begin by explaining the 'zero-drift' approach [17] wherein the decoder has access to exactly the same prediction error reconstruction and then describe the 'controlled drift' approach where the constraint of zero-drift is relaxed.

B. Zero Drift Approach: Distributed Predictive Encoder

The distributed predictive encoder (in zero-drift approach) for source X is depicted in Fig. 2. High resolution quantizer Q_x maps the prediction error $e_x = X - \tilde{X}_{enc}$ to an index k representing Voronoi region C_k^x. Next, a *lossy* mapping which we refer to as Wyner-Ziv (WZ) mapping is employed (the name loosely accounts for the fact that the scenario involves lossy coding with side information whose asymptotic performance bound was given in [2]). The WZ mapping block takes in k and outputs transmission index $i = v(k)$ representing region $R_i^x = \bigcup_{k;v(k)=i} C_k^x$, as well as a corresponding prediction error reconstruction value $\hat{e}_{x,enc}$ for the encoder prediction loop. An example of WZ mapping for a scalar source with $\mathcal{K} = 7$ and $\mathcal{I} = 3$, is given in Fig. 2. The reconstructed residual $\hat{e}_{x,enc}$ is added to $\tilde{X}_{enc}$ to obtain $\hat{X}_{enc}$, the sample reconstruction value for the encoder prediction loop. A linear predictor P_x is used to predict the next sample of X from

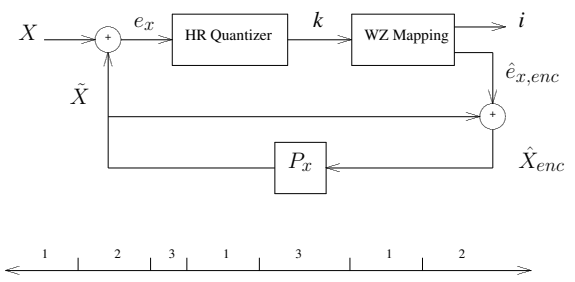

Fig. 2. Block diagram of a DPC encoder and a scalar example of WZ mapping from prototypes (Voronoi regions) to indices.

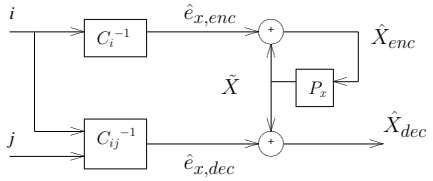

Fig. 3. The DPC decoder (Zero-Drift) reconstructing source X

$\hat{X}_{enc}$. For the second source Y, we similarly define the quantizer Q_y, regions C_l^y, R_j^y and prototypes e_l^y. Here, the L Voronoi regions are mapped to J indices via WZ mapping $w(l) = j$.

The joint decoder receives an index pair (i, j) to generate reconstruction values $\hat{e}_x$ and $\hat{e}_y$, and calculates $\hat{X}$ and $\hat{Y}$. We next explain the functioning of the distributed predictive coder.

C. Distributed Predictive Decoder

The decoder module in charge of reproducing X (see Fig. 3) receives indices i and j from sources X and Y respectively. The index i is used to reconstruct $\hat{e}_{x,enc}$ and the operations at encoder prediction loop are mimicked (in order to avoid drift) to generate $\hat{X}_{enc}$ and $\tilde{X}_{enc}$ using the predictor P_x. Given index pair (i, j), the decoder retrieves $\hat{e}_{x,dec}$ from the *decoder codebook*, and adds it to $\tilde{X}_{loop}$ to obtain the decoder reconstruction $\hat{X}_{dec}$. Here C_i^{-1} and C_{ij}^{-1} represent "inverse quantizers", i.e., the corresponding table look-up operation applied to the respective codebooks.

D. Observations and Intuitive Considerations

It is important to note that the WZ mapping compromises the quality of the sample reconstruction in the prediction loop in order to exploit inter-source correlation and improve the decoder reconstruction. In particular, region $R_i^x = \bigcup_{k;v(k)=i} C_k^x$ is a union of likely distant Voronoi cells C_k^x in the hope that the information from source Y will allow the decoder to separate them (see the example mapping in Fig. 2). A fundamental tradeoff emerges, as in order to exploit inter-source correlations between e_x and e_y to better reconstruct the current sample at the decoder, we compromised the performance of the prediction loop and hence the quality of future reconstruction.

It should also be noted that $\tilde{X}_{enc}$ is a (coarse) reconstruction of X which only serves the prediction loop, and is generally different from $\hat{X}_{dec}$, the decoder reconstruction of X. Also

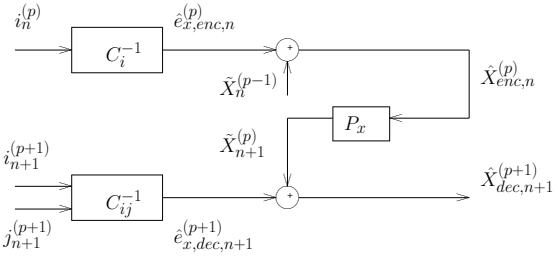

Fig. 4. DPC decoder in open loop during the design phase

note that the "encoder codebook" C_i^{-1} which is used in the prediction loop at both the encoder and the decoder is, in general, different from the "decoder codebook" C_{ij}^{-1} (used only at the decoder).

E. Closed Loop vs ACL Design

We note that the quantized error sample $\hat{e}_{x,enc}$ at time n impacts $\tilde{X}_{enc}$ and $\hat{X}_{dec}$ from time $n+1$ onwards due to the presence of prediction loop. On the other hand, $\hat{e}_{x,dec}$ at time n only impacts $\hat{X}_{dec}$ at time n, as is explicitly depicted in Fig. 3. Hence, if one tries to directly design a *distributed quantizer* for the quantities being quantized, namely, the pair of prediction errors e_x, e_y, while ignoring effects on the prediction loop, i.e., minimize the following distortion:

$$E[\alpha d(e_x, \hat{e}_{x,dec}) + (1-\alpha) d(e_y, \hat{e}_{y,dec})] \tag{2}$$

(see e.g., DSC in [12]), the ultimate distortion in (1) will not be minimized.

It is evident that there are conflicting design objectives for the distributed quantizer in terms of current reconstruction versus prediction performance. Let us now consider the need for an asymptotic closed-loop (ACL) approach that allows the design iteration to be performed in open-loop, but with essential closing of the loop asymptotically. Consider the "open-loop" decoder of Fig. 4. Here a particular sample of $\hat{e}_{x,enc}$ will affect only the next sample (in time) of $\hat{X}_{dec}$ and not all the samples following it. The super-script p in Fig. 4 denotes the ACL iteration (more details and clarifications in the next section). The ACL design iteration is actually performed in open-loop by keeping the $\tilde{X}_{enc}$ sequence (for all n) fixed throughout the iteration. A new $\tilde{X}_{enc}$ sequence is then available for the next iteration of ACL. At this point we merely point out that the design is performed in open-loop but the prediction loop is effectively closed asymptotically as the operation mimics closed loop.

By adopting ACL for a DPC system, we can optimize the various codebooks and WZ mappings to minimize the distortion of (1). A zero-drift approach for DPC design utilizing ACL has recently been presented in [17]. Herein we relax the zero-drift constraint in order to achieve better exploitation of inter-source correlations. We next outline the controlled drift approach, the underlying ACL, and show how it can be applied to DPC design.

III. Controlled Drift Approach

A. Motivation & Description

To maintain zero-drift in the system, the encoder codebook is restricted to index i as input. However, the source encoder has complete knowledge of the prediction error itself or effectively index k (which is the output of high resolution quantizer used primarily to discretize the source), while the decoder has additional knowledge about the prediction error from the correlated source Y, in the form of index j. This implies that there exist some additional information that could be exploited, if an appropriate means were devised. We now use different codebooks for the prediction loop at the decoder versus encoder. The encoder codebook can have k as input, and the decoder loop codebook has inputs i and j. This flexibility enables better utilization of inter-source correlation at the cost of some drift in the system. However, appropriate design of encoder and loop codebooks (as well as predictor) will optimize the precise overall performance while accounting for and managing the drift. Note that the controlled-drift approach actually subsumes the zero-drift scheme where the encoder and loop codebooks are effectively the same and depend only on i. The encoder and decoder for controlled drift approach are depicted in Fig. 5 and Fig. 6.

Components to Optimize Distributed predictive coding design optimizes the predictors P_x, P_y, high rate quantizers, WZ mappings, encoder codebooks, loop codebooks (in controlled drift approach) and decoder codebooks for all sources. We will restrict the scope here to the design of the various codebooks and WZ mappings, and briefly touch on predictor optimization later on. (For simplicity, we first assume a fixed predictor and high rate quantizers that are designed using Lloyd's algorithm [21] given the open-loop prediction error.)

We next give a brief overview of ACL, cast the distributed predictive coding problem within the ACL framework and give the update rules (necessary conditions for optimality) for the WZ mappings and the three different codebooks: Encoder, Loop and Decoder; for both the sources. the encoder, loop and decoder codebooks. For conceptual simplicity, we analyze the DPC system assuming first order linear prediction. An iterative algorithm for zero-drift coding can be found in our previous work [17].

B. Asymptotic Closed Loop Design

1) Predictive Quantizer Design: Let us consider standard (non-distributed) predictive coding. A predictive quantizer can be designed using an open-loop (OL) or closed-loop (CL) approach [18]. In OL the training set of prediction errors for quantizer design is independent of the quantizer and a greedy design algorithm (e.g., Lloyd's) is stable and converges to a local minimum. However, the prediction loop must be closed to operate the designed system, resulting in prediction-error statistics that differ from those observed during design. Hence, the system performance is suboptimal. In CL, the system iterates a closed loop run to generate new training data, followed by redesign of the quantizer, until (hopefully) convergence. However, since the training set changes with each iteration, each redesigned quantizer is applied to error statistics it had not been designed for. Moreover, the change in statistics is generally unpredictable as, due to the prediction loop that feeds back errors, there can be distortion build up as the sequence is processed causing non-stationary statistics and actual divergence (in terms of the performance cost). In general, there is no guarantee that the algorithm will converge and the procedure may be unstable.

The asymptotic closed-loop (ACL) design approach [20], [19] overcomes these shortcomings of traditional predictive coder design. A subterfuge is employed wherein the design is effectively performed in open-loop, where each quantizer is designed for the statistics of the exact signal it then quantizes to produce a new sequence of reconstruction for the next iteration, thereby circumventing stability issues. Asymptotically, the loop is effectively closed in the sense that the design approaches closed-loop statistics despite open-loop operation within each iteration. For a detailed treatment of ACL and its applications, see [19], [20].

More specifically, for a given quantizer $Q^{(p-1)}$ obtained at iteration $p-1$, a new training set of prediction errors $T^{(p)} = \{e_n^{(p)}\}_{n=1}^N$ is generated as:

$$e_n^{(p)} = x_n - P[\hat{x}_{n-1}^{(p-1)}], \tag{3}$$

where the subscript n denotes time and P is the predictor. Using $T^{(p)}$, a new quantizer $Q^{(p)}$ is designed and a new set of reconstruction values for x is obtained by applying the new quantizer on $T^{(p)}$ itself as:

$$\hat{x}_n^{(p)} = P[\hat{x}_{n-1}^{(p-1)}] + Q^{(p)}[e_n^{(p)}]. \tag{4}$$

Note that the prediction is not from the reconstruction of the previous sample at the current iteration, but rather from the fixed reconstruction sequence of the previous iteration. Hence, unlike CL, the prediction errors to be quantized are fixed and do not change as we modify the quantizer. Since the quantizer is applied to the exact error training set for which it was designed, it is the best quantizer for the job and hence the cost will decrease. This will result in better prediction. A new error training set $T^{(p+1)}$ is then obtained and the procedure is performed until convergence. Since the entire design is performed in open-loop, it is stable. At convergence, the quantizer updates are vanishingly small $Q^{(p+1)} \approx Q^{(p)}$. Therefore, the reconstructed sequence is unchanged with iterations, i.e., $\hat{x}_n^{(p+1)} \approx \hat{x}_n^{(p)}$ implying $P[\hat{x}_{n-1}^{(p+1)}] \approx P[\hat{x}_{n-1}^{(p)}]$ which means that asymptotically we are effectively predicting from the previous sample reconstruction in the current iteration, i.e., the loop is effectively closed. So, even though the algorithm is always running in open loop, the design asymptotically approaches closed loop conditions.

2) ACL for Distributed Predictive Coder Design: Fig.6 depicts the decoder of source X during ACL design of a distributed predictive coding system. A similar decoder for Y is not shown. Note that the prediction loop is indeed open.

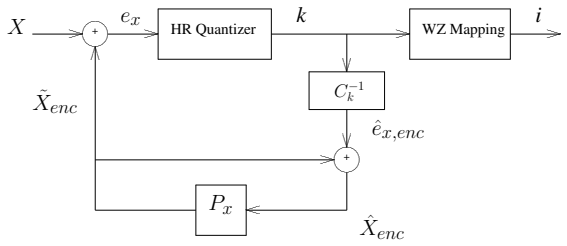

Fig. 5. Controlled-Drift DPC encoder

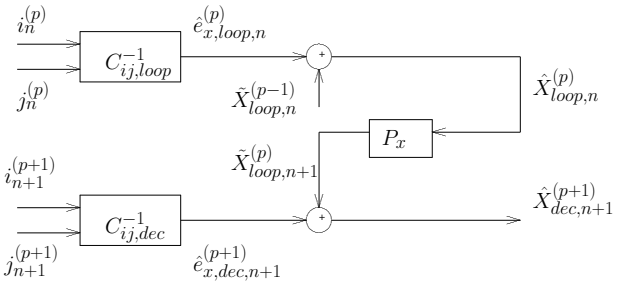

Fig. 6. Controlled-Drift DPC decoder

The distortion cost to be minimized is:

$$E[\alpha\ d(X, \hat{X}_{dec}^{(p+1)}) + (1-\alpha)\ d(Y, \hat{Y}_{dec}^{(p+1)})]. \quad (5)$$

Clearly, during iteration p, we seek to minimize the decoder reconstruction error at iteration $p+1$. While it is immediately seen from Fig. 4 and Fig. 6 that the impact of the optimization should be so measured, this is an important observation because it illustrates the greater importance of ACL in the *distributed coding* scenario. In the case of plain (non-distributed)predictive quantization, optimizing the quantizer for best reconstruction of the current sample is a reasonable (though not perfect) objective, since better reconstruction is expected to help prediction of the next sample. This is no longer the case in DPC because of the WZ module. Optimizing WZ for best decoder reconstruction of the current sample will considerably compromise the encoder reconstruction (which has no access to source Y) and thereby compromise the prediction loop. This is, in fact, an illustration of the underlying conflict between predictive and distributed coding. Note finally that, asymptotically, the fact that we optimize for reconstruction at iteration $p+1$ makes no difference due to the effectively closed loop.

C. Update Rules

We assume mean-squared error distortion for simplicity. While the notation in what follows is heavy due to the multiple indexing involved in DPC; in a nutshell, we alternate between optimization of the decoder codebook, encoder codebook, loop codebook and WZ mapping while fixing the other three. Let the distortion for the next sample in the next iteration be:

$$d_{n+1}^{(p+1)} = \alpha\ d[x_{n+1} - \tilde{x}_{loop,n+1}^{(p)}, \hat{e}_{x,dec}(i_{n+1}^{(p+1)}, j_{n+1}^{(p+1)})] + (1-\alpha)\ d[y_{n+1} - \tilde{y}_{loop,n+1}^{(p)}, \hat{e}_{y,dec}(i_{n+1}^{(p+1)}, j_{n+1}^{(p+1)})]. \quad (6)$$

The following determines the update rules in terms of the subset of distortion terms to be minimized (i.e., those that depend on the parameters being updated) while avoiding detailed notation.

1) **Decoder Codebook**: Entry (i,j), $i = 1:\mathcal{I}$ and $j = 1:\mathcal{J}$ is obtained by minimizing

$$\sum_{n:(e_{x,n+1}^{(p+1)}, e_{y,n+1}^{(p+1)}) \in R_i \times R_j} d_{n+1}^{(p+1)}. \quad (7)$$

2) **Loop Codebook**: Entry (i,j), $i = 1:\mathcal{I}$ and $j = 1:\mathcal{J}$ is given by minimizing:

$$\sum_{n:(e_{x,n}^{(p)}, e_{y,n}^{(p)}) \in R_i \times R_j} d_{n+1}^{(p+1)}. \quad (8)$$

3) **Encoder Codebook**: Entry k, $k = 1:\mathcal{K}$ is given by minimizing:

$$\hat{e}_{x,enc}(k) = \sum_{n:e_{x,n}^{(p)} \in C_k} d_{n+1}^{(p+1)}, \quad (9)$$

4) **WZ Mappings**: For $k = 1:\mathcal{K}$, assign region k to index $i = v(k)$ such that:

$$v(k) = \arg \min_{i \in \{1..I\}} \sum_{\substack{n:e_{x,n}^{(p)} \in C_k \text{ or} \\ e_{x,n+1}^{(p+1)} \in C_k}} d_{n+1}^{(p+1)}. \quad (10)$$

To reduce clutter, superscripts were omitted above where obvious, e.g., R_i rather than R_i^x.

IV. Simulation Results

The following Gauss-Markov source model is used for simulations: $X_n = \beta X_{n-1} + w_n$ and $Y_n = \gamma Y_{n-1} + u_n$, where w_n, u_n are i.i.d., zero-mean, unit variance, jointly Gaussian scalar sources with correlation coefficient ρ. A training set of size 5000 scalars is generated. The predictors P_x (and P_y) are first-order linear predictors designed using X (and Y). Simulation results are depicted in Fig. 7. In all simulations, the weighting coefficient of (1) is set to $\alpha = 0.5$ so that equal importance is given to both sources at the decoder. The number of prototypes is 60 for each source.

In the first experiment, $\beta = \gamma = 0.8$ and $\rho = 0.97$. Both sources are encoded at the same rate. The weighted distortion at the decoder is plotted versus the number of transmitted bits for each source. We compare: (a) non-distributed predictive coding (PC), i.e., each source is compressed using standard predictive coding; (b) memoryless distributed coding, i.e., no prediction is performed; (c) zero-drift based distributed predictive coding (DPC-ZD) and (d) controlled-drift based distributed predictive coding (DPC-CD). The two DPC schemes (with or without drift) clearly outperform the other two compression schemes and gains of ~ 1.8 dB are achieved (e.g., at

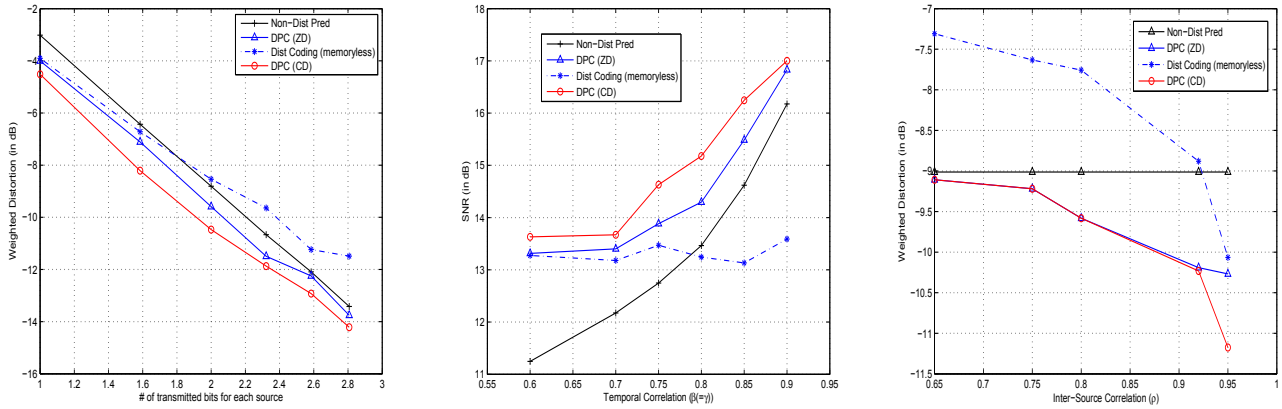

Fig. 7. Performance comparison of distributed predictive coding schemes, non-distributed predictive coding, and memoryless distributed coding. The figures show distortion or SNR vs. rate, temporal correlation, and inter-source correlation.

$R_1 = R_2 = 2$ bits/sample) between the DPC-CD scheme and tradition predictive coding or memoryless distributed coding. In the second experiment, $\rho = 0.97$ and the transmission rates for the sources are fixed at 2 bits/sample. Here, we plot $SNR = \frac{\alpha E[X^2]+(1-\alpha)E[Y^2])}{\alpha E[(X-\hat{X})^2]+(1-\alpha)E[(Y-\hat{Y})^2]}$ versus temporal correlation $\beta(=\gamma)$. In the third experiment, $\beta = \gamma = 0.6$ and $R_1 = R_2 = 2$ bits/sample. We plot the weighted distortion versus inter-source correlation ρ.

All experiments provide evidence that DPC schemes perform considerably better than individual predictive coding of sources or distritbuted (memoryless) coding.

ACKNOWLEDGMENTS

This work was supported in part by the NSF under grant IIS-0329267, the University of California MICRO program, Applied Signal Technology Inc., Dolby Laboratories Inc., and Qualcomm Inc.

REFERENCES

[1] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. on Information Theory*, vol. 19, no. 4, pp. 471–480, Jul 1973.

[2] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side-information at the decoder," *IEEE Trans. on Information Theory*, vol. 22, pp. 1–10, Jan 1976.

[3] S. S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): Design and construction." *IEEE Trans. on Information Theory*, vol. 49, no. 3, pp. 626–643, Mar 2003.

[4] S. Pradhan, J. Kusuma, and K. Ramchandran, "Distributed compression in a dense microsensor network," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 51–60, Mar 2002.

[5] Z. Xiong, A. Liveris, and S.Cheng, "Distributed source coding for sensor networks," *IEEE Signal Processing Magazine*, vol. 21, no. 5, pp. 80–94, Sep 2004.

[6] X. Wang and M. T. Orchard, "Design of trellis codes for source coding with side information at the decoder," in *IEEE Data Compression Conference*, Mar 2001, pp. 361–370.

[7] P. Mitran and J. Bajcsy, "Coding for the wyner-ziv problem with turbo-like codes," in *IEEE International Symposium on Information Theory*, Jul 2002, p. 91.

[8] J. Garcia-Frias and Y. Zhao, "Compression of correlated binary sources using turbo codes," *IEEE Communication Letters*, vol. 5, no. 10, pp. 417–419, Oct 2001.

[9] A. D. Liveris, Z. Xiong, and C. Eorghiades, "Compression of binary sources with side information at the decoderusing ldpc codes," *IEEE Communication Letters*, vol. 6, no. 10, pp. 440–442, Oct 2002.

[10] J. Cardinal and G. B. Assche, "Joint entropy-constrained multiterminal quantization," in *IEEE International Symposium on Information Theory*, Jun 2002, p. 63.

[11] M. Fleming, Q. Zhao, and M. Effros, "Network vector quantization," *IEEE Trans. on Information Theory*, vol. 50, no. 8, pp. 1584–1604, Aug 2004.

[12] D. Rebollo-Monedero, R. Zhang, and B. Girod, "Design of optimal quantizers for distributed source coding," in *IEEE Data Compression Conference*, Mar 2003, pp. 13–22.

[13] A. Saxena, J. Nayak, and K. Rose, "A Global Approach to Joint Quantizer Design for Distributed Coding of Correlated Sources." in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, May 2006, pp. 53–56.

[14] E. Tuncel, "Predictive coding of correlated sources," in *IEEE Information Theory Workshop*, Oct 2004, pp. 111–116.

[15] A. Saxena, J. Nayak, and K. Rose, "On Efficient Quantizer Design For Robust Distributed Source Coding," in *IEEE Data Compression Conference*, Mar 2006, pp. 63–72.

[16] P. Yahampath, "Predictive vector quanizer design for distributed source coding," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, April 2007, pp. 629–632.

[17] A. Saxena and K. Rose, "Distributed predictive coding for spatio-temporally correlated sources," in *IEEE International Symposium on Information Theory*, June 2007.

[18] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

[19] H. Khalil, K. Rose, and S. L. Regunathan, "The asymptotic closed-loop approach to predictive vector quantizer design with application in video coding," *IEEE Trans. on Image Processing*, vol. 10, no. 1, pp. 15–23, Jan 2001.

[20] H. Khalil and K. Rose, "Predictive vector quantizer design using deterministic annealing," *IEEE Trans. on Signal Processing*, vol. 51, no. 1, pp. 244–254, Jan 2003.

[21] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Trans. on Information Theory*, vol. 28, no. 2, pp. 129–137, Mar 1982.