FORWARD ADAPTIVE KLT CODING

Carlos E. Davila

Electrical Engineering Dept., Southern Methodist University
Dallas, Texas 75205, USA

ABSTRACT

A framework for implementing the forward adaptive Karhunen-Loève Transform (FAKLT) is described. Unlike backward adaptive methods, FAKLT computes transform coefficients using basis vectors derived from the most recent signal frame. As a result, it exhibits improved energy compaction compared to the backward adaptive KLT. The method encodes only the KLT coefficients and a small amount of side information, the KLT basis vectors (eigenvectors) are not encoded.

Index Terms— Data compression, Eigenvalues and eigenfunctions, Karhunen-Loeve transforms, Adaptive systems

1. BACKGROUND

Many signal compression algorithms are frame-based, their goal being to represent the $N \times 1$ source frame

$$x_n = \left[\begin{array}{cccc} x(nN-1) & x((nN-2) & \cdots & x(N(n-1)) \end{array} \right]^T$$

$n = 1, 2, \ldots$, with as few bits as possible. In transform coding, rather than quantizing and encoding x_n directly, it is multiplied by a transform matrix A, giving a set of transform coefficients: $y_n = A^T x_n$. The matrix A is usually an orthonormal matrix, i.e. $AA^T = I$. These coefficients are then quantized, giving $\hat{y}_n$ after which they are encoded and transmitted. To recover $\hat{x}_n$, the decoder must also have knowledge of the transform matrix. The estimated source frame is then given by $\hat{x}_n = A\hat{y}_n$. This is why the columns of the transform matrix A are called the *basis vectors* of the transform. The degree to which $\hat{y}_n$ is close to y_n of course will determine how close $\hat{x}_n$ will be to x_n.

The KLT is known to be the optimum transform for Gaussian sources, when using either a fixed rate or variable rate encoder [1]. Certain non-Gaussian sources have been found for which the KLT is sub-optimal [2], while a relatively rich class of distributions called Gaussian scale mixtures (GSM) and Gaussian vector-scale mixtures (GVSM) are known for which the KLT is optimum [3]. The GSM sources include as special cases the Laplacian, Cauchy, symmetrized gamma, and lognormal densities and the GVSM sources extend the range of distributions for which the KLT is optimum even further [3]. The KLT transform matrix consists of the eigenvectors of the autocorrelation matrix of the source frame x_n. By assumption x_n has autocorrelation matrix $R = E\left[x_n x_n^T\right]$, which is assumed to have rank N. This means that x_n can be represented as a linear combination of the eigenvectors of R given by $q_1, q_2, \ldots, q_N$, corresponding to eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_N > 0$, respectively. Let $Q = \left[\begin{array}{cccc} q_1 & q_2 & \cdots & q_N \end{array} \right]$ be an $N \times N$ matrix whose columns are the KLT basis vectors (eigenvectors of R). Then the transform matrix is given by $A = Q$. The transform coefficients are given by $y_n = Q^T x_n$. If the signal $x(n)$ is statistically stationary then the eigenvectors need only be computed and encoded once, which would not lead to much loss of compression, however in practice, the eigenstructure of most signals tends to vary considerably over time. Hence the eigenvectors of R need to be constantly re-encoded which is why the KLT is not often used. We will describe an algorithm for doing KLT coding using forward-adaptive updates. That is, the encoder transmits the transform coefficients, and the decoder computes the transform matrix using only the transform coefficients and a minimal amount of side information.

This paper is organized as follows: Section 2 gives a review of backward-adaptive KLT algorithms. Section 3 describes the encoder and decoder computations for the proposed forward-adaptive KLT algorithm. Section 4 discusses some additional quantization effects which the proposed algorithm exhibits that are not found in traditional transform coding. Finally, Section 5 documents some simulations while Section 6 gives some conclusions.

2. BACKWARD ADAPTIVE KLT

Goyal et al. have developed an algorithm for doing backward adaptive KLT transforms [1]. Several closely related algorithms have also been published [4, 5]. Backward adaptation makes it possible for the decoder to determine the KLT basis vectors given only knowledge of the KLT coefficients. In order to implement the backward adaptive KLT, the autocorrelation matrix is estimated using

$$\hat{R}_n = \gamma \hat{R}_{n-1} + \hat{x}_n \hat{x}_n^T \tag{1}$$

where $0 < \gamma < 1$ and $\hat{x}_n$ is the estimated source frame. Let

$$\hat{Q}_n = \left[\begin{array}{cccc} \hat{q}_{1,n} & \hat{q}_{2,n} & \cdots & \hat{q}_{N,n} \end{array} \right] \tag{2}$$

and $\hat{\Lambda}_n = \text{diag}(\hat{\lambda}_1(n), \hat{\lambda}_2(n), \ldots, \hat{\lambda}_N(n)), \hat{\lambda}_1(n) \geq \hat{\lambda}_2(n) \geq \cdots \hat{\lambda}_N(n)$ be the eigenvectors and eigenvalues, respectively, of $\hat{R}_n$. The autocorrelation matrix is estimated using $\hat{x}_n$ instead of x_n because both the encoder and decoder need to have identical estimates of the autocorrelation matrix in order for the decoder to be able to compute the same KLT basis vectors being used by the encoder. The encoder first computes the KLT coefficients as $y_n = \hat{Q}_{n-1}^T x_n$. The key to backward adaptation is to use the transform matrix at time $n-1$ rather than at time n. The coefficients can then be quantized $\hat{y}_n = \Delta(y_n)$ where Δ represents an arbitrary quantizer. The encoder must then update the autocorrelation matrix estimate using (1) with $\hat{x}_n = \hat{Q}_{n-1}\hat{y}_n$. The encoder then performs an eigenvalue decomposition of $\hat{R}_n$ to find $\hat{Q}_n$. Finally $\hat{y}_n$ is encoded and sent to the decoder. The decoder computes the source frame estimate as $\hat{x}_n = \hat{Q}_{n-1}\hat{y}_n$. The decoder can then update $\hat{R}_n$ and $\hat{Q}_n$ just as the encoder did. This approach to backward adaptive KLT, henceforth called BAKLT is summarized in Table 1. One drawback with BAKLT is that it uses the KLT matrix at time $n-1$, i.e. $\hat{Q}_{n-1}$ to

encoder	decoder
$y_n = \hat{Q}_{n-1}^T x_n$	receive and decode $\hat{y}_n$
$\hat{y}_n = \Delta(y_n)$	$\hat{x}_n = \hat{Q}_{n-1}\hat{y}_n$
$\hat{x}_n = \hat{Q}_{n-1}\hat{y}_n$	$\hat{R}_n = \gamma\hat{R}_{n-1} + \hat{x}_n\hat{x}_n^T$
$\hat{R}_n = \gamma\hat{R}_{n-1} + \hat{x}_n\hat{x}_n^T$	$\hat{R}_n = \hat{Q}_n\hat{\Lambda}_n\hat{Q}_n$
$\hat{R}_n = \hat{Q}_n\hat{\Lambda}_n\hat{Q}_n$	
encode and transmit $\hat{y}_n$	

Table 1. Computations for backward adaptive KLT (BAKLT).

compute that KLT coefficients at time n. This is problematic given that the KLT basis vectors at time $n-1$ do not account for any new information in x_n. Though several authors have shown that backward adaptive KLT will eventually converge to the optimum KLT, this is only true for stationary sources [1, 5]. If the source is nonstationary, BAKLT will always require an increased rate in order to arrive at the distortion level which the conventional KLT can achieve. This is due to the the fact that at time n, the KLT basis vectors have only been adapted up to time $n-1$. In nonstationary environments, the KLT will exhibit improved energy compaction in its transform coefficients relative to the BAKLT.

3. FORWARD ADAPTIVE KLT

Forward adaptive KLT (FAKLT) differs from BAKLT in several important respects. While BAKLT uses $\hat{Q}_{n-1}$ to form the KLT coefficients, FAKLT uses the eigenvectors of $R_n = \gamma\hat{R}_{n-1} + x_n x_n^T$ which we denote by Q_n, since the update uses the original, unquantized frame x_n. The encoder then computes the KLT coefficients $y_n = Q_n^T x_n$ which are then quantized to form $\hat{y}_n$. On the other hand, in order to replicate the computations being carried out by the decoder, the encoder must ultimately compute $\hat{x}_n = \hat{Q}_n\hat{y}_n$, from which it obtains $\hat{R}_n = \gamma\hat{R}_{n-1} + \hat{x}_n\hat{x}_n^T$ Both the encoder and decoder find $\hat{Q}_n$ from $\hat{y}_n$ by solving an additive inverse eigenvalue problem (AIEP). Consider the sample autocorrelation matrix update:

$$\hat{R}_n = \gamma\hat{R}_{n-1} + \hat{Q}_n\hat{y}_n\hat{y}_n^T\hat{Q}_n^T \tag{3}$$

Pre- and post-multiplying by $\hat{Q}_n^T$ and $\hat{Q}_n$, respectively, and rearranging gives $\gamma\hat{Q}_n^T\hat{R}_{n-1}\hat{Q}_n = \hat{\Lambda}_n - \hat{y}_n\hat{y}_n^T$. If we treat the diagonal matrix $\hat{\Lambda}_n$ as an unknown quantity, say C, the AIEP involves finding the unknown diagonal matrix C such that the eigenvalues of $C - \hat{y}_n\hat{y}_n^T$ match the eigenvalues of $\gamma\hat{Q}_n^T\hat{R}_{n-1}\hat{Q}_n$. Since these eigenvalues, $\gamma\hat{\Lambda}_{n-1}$, and the KLT coefficients $\hat{y}_n$ are known to the encoder and decoder at time n, the problem is tractable. The AIEP has been studied extensively and can be solved with Newton's method [6, 7]. Once $C = \hat{\Lambda}_n$ has been found, an eigenvalue decomposition leads to $\hat{W}_n\hat{\Lambda}_{n-1}\hat{W}_n^T = \hat{\Lambda}_n - \hat{y}_n\hat{y}_n^T$ where $\hat{W}_n = \hat{Q}_n^T\hat{Q}_{n-1}$. This makes it possible to compute the KLT basis vectors as $\hat{Q}_n = \hat{Q}_{n-1}\hat{W}_n^T$. The sign of the eigenvectors in $\hat{W}_n$ may not agree with the corresponding signs of $\hat{Q}_n$ and $\hat{Q}_{n-1}$ so by convention, the encoder chooses the sign of the eigenvectors in Q_n so that the diagonal entries of $W_n \equiv Q_n^T\hat{Q}_{n-1}$ are positive. For now, we will also assume that the diagonal entries of W_n have the maximum modulus for each column. The signs of of the columns of $\hat{W}_n$ can subsequently be chosen so that its diagonal entries are positive. We will assume that the eigenvalues $\hat{\Lambda}_{n-1}$ are distinct, i.e. $\hat{\lambda}_1(n-1) > \hat{\lambda}_2(n-1) > \cdots > \hat{\lambda}_N(n-1)$. To solve the AIEP, we apply Newton's method to the following nonlinear system of equations

$$f_n(c) = \begin{bmatrix} \gamma\hat{\lambda}_1(n-1) - \tilde{\lambda}_1(n) \\ \gamma\hat{\lambda}_2(n-1) - \tilde{\lambda}_2(n) \\ \vdots \\ \gamma\hat{\lambda}_N(n-1) - \tilde{\lambda}_N(n) \end{bmatrix} = 0 \tag{4}$$

where $\tilde{\lambda}_i(n), i = 1, \ldots, N$, are the eigenvalues of $C - \hat{y}_n\hat{y}_n^T$ with $C = diag(c)$, and $c \in \mathbb{R}^N$ is the unknown parameter vector. Let the eigenvectors of $C - \hat{y}_n\hat{y}_n^T$ be

$$\hat{W}_n = \begin{bmatrix} \hat{w}_{1,n} & \hat{w}_{2,n} & \cdots & \hat{w}_{N,n} \end{bmatrix} \tag{5}$$

Since $\tilde{\lambda}_i(n) = \hat{w}_{i,n}^T\left(C - \hat{y}_n\hat{y}_n^T\right)\hat{w}_{i,n}, i = 1, \ldots, N$,the ik^{th} entry of the Jacobian of $f_n(c)$ can be shown to be given by

$$J_{ij}(c) = \hat{w}_{i,n}(j)^2, \quad i,j = 1, \ldots, N \tag{6}$$

The parameter vector in Newton's method is updated by solving the following linear system $J(c_k)(c_{k+1} - c_k) = f_n(c_k)$ where c_k represents the kth iterate, and ideally should converge to the elements of $\hat{\Lambda}_n$. The details for this method can be found in [6] under Method I. Newton's method for the AIEP converges quadratically, provided the starting point is sufficiently close to the solution [6]. The iteration is carried out until the eigenvalues of $diag(c_k) - \hat{y}_n\hat{y}_n^T$ converge to $\gamma\hat{\Lambda}_{n-1}$. To insure that Newton's method converges, we use homotopy continuation [8, 9]. The quantity $\tilde{\lambda}_k(n), k = 1, \ldots, N$ in (4) is replaced by the eigenvalues of $C - t\hat{y}_n\hat{y}_n^T$ where $0 \leq t \leq 1$. For $t = 0$, the AIEP has a simple solution, the diagonal elements of $\gamma\hat{\Lambda}_{n-1}$. By slowly increasing t toward 1, Newton's method is able to track a solution, since the starting point is always close to the solution. The solution for $t = 1$, of course, corresponds to the desired solution for the AIEP. It can be readily seen that the AIEP in (4) with $\tilde{\lambda}_i(n) = w_{i,n}^T\left(C - t\hat{y}_n\hat{y}_n^T\right)w_{i,n}, i = 1, \ldots, N$ is a homotopy function, $h(c,t) : \mathbb{R}^{N+1} \to \mathbb{R}^N$. Moreover, it can be shown that $h(c,t) = 0$ represents a solution path in $\mathbb{R}^{N+1}$ that is unique and continuously differentiable [8]. We will let t increase monotonically, i.e. we will not allow intermittent decreases. The following theorems, stated without proof, give conditions under which this can happen.

Theorem 1. *Let $\bar{J}$ be the $N \times (N+1)$ Jacobian matrix of the homotopy function $h : \mathbb{R}^{N+1} \to \mathbb{R}^N$. If the sign of $\det \bar{J}_{N+1}$ remains fixed, and $\det \bar{J}_{N+1} \neq 0$ then t can be increased monotonically.*

Theorem 1 essentially fixes how large we can let t get. Let t_{max} be this maximum value. Given that t can only increase monotonically, we must establish conditions which enable t_{max} to attain the desired value of one, and thereby allow us to find a solution to $f_n(c) = 0$. The following theorem, which makes use of the matrix 2-norm, does this.

Theorem 2. *If $\|\gamma\hat{\Lambda}_{n-1}\|_2$ is sufficiently larger than $\|t\hat{y}_n\hat{y}_n^T\|_2$, then $t_{max} \geq 1$. Moreover, $\|\gamma\hat{\Lambda}_n\|_2$ can be made arbitrarily large by letting $\gamma \to 1$.*

The trade-off to choosing γ close to one is, of course, a loss of adaptivity. The following theorem addresses the number of solutions to the AIEP.

Theorem 3. *The AIEP has $N!$ solutions for $0 \leq t \leq t_{max}$.*

A similar result for more general inverse eigenvalue problems is given in [10]. Since there are $N!$ possible solutions to the AIEP, finding the correct solution becomes a seemingly impossible task for large values of N. However this task can be greatly simplified by noting that the autocorrelation matrix eigenvectors do not change much from one frame to the next. The encoder, which has access to Q_n can determine the extent of the changes in the eigenvectors from frame $n-1$ to frame n by computing $W_n = Q_n^T \hat{Q}_{n-1}$. The diagonal entries of this product should be large, typically between 0.7 and 1. Even though the change in the eigenvectors may be small, two or more adjacent eigenvectors may permute from one frame to the next. Under these conditions then clearly, the homotopy continuation method will converge to one of the $N! - 1$ incorrect solutions. To insure convergence to the correct solution, we re-order the KLT coefficients $\hat{y}_n$ so that they correspond to the ordering of the eigenvalues $\hat{\Lambda}_{n-1}$. Re-ordering the KLT coefficients implies a corresponding re-ordering of the columns of Q_n. Once the correct solution is obtained at $t = 1$, the eigenvalues of $\hat{R}_n$ can simply be sorted in decreasing order in preparation for the next frame.

4. EIGENVECTOR SENSITIVITY TO QUANTIZATION

Next, we look at the sensitivity of the eigenvectors computed with the homotopy continuation method in response to quantization errors. This is important because since $\hat{x}_n = \hat{Q}_n \hat{y}_n$, quantization errors in $\hat{x}_n$ will depend on quantization errors present in both $\hat{Q}_n$ and $\hat{y}_n$, rather than on just $\hat{y}_n$ as is the case when using more traditional forms of transform coding. We begin with a well-known result from eigenvector perturbation theory [11]

Theorem 4. *Let the matrix $A \in \mathbb{R}^{N \times N}$ have eigenvectors $q_1, q_2, \cdots, q_N$ and corresponding eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_N$. For some positive constant $\epsilon \ll 1$, the eigenvectors $\hat{q}_m$ of $A + \epsilon F$ can be approximated as*

$$\hat{q}_m \approx \left[q_m + \sum_{\substack{i=1 \\ i \neq m}}^{N} \frac{q_i^T F q_m}{\lambda_m - \lambda_i} q_i \right], m = 1, \ldots, N \qquad (7)$$

Since the eigenvectors in $\hat{Q}_n$ differ from $\hat{W}_n^T$ by a multiplication with an orthogonal matrix, i.e. $\hat{Q}_n = \hat{Q}_{n-1} \hat{W}_n^T$, quantization errors in $\hat{Q}_n$ will have the same matrix norm as those of $\hat{W}_n^T$. Consider the matrix $C - y_n y_n^T$. The solution to the AIEP is $C = \Lambda_n$, the eigenvalues of R_n. On the other hand, the AIEP solution to $C - \hat{y}_n \hat{y}_n^T$ is $C = \hat{\Lambda}_n \neq \Lambda_n$. Letting $\Delta\Lambda_n = \hat{\Lambda}_n - \Lambda_n$ and $\Delta\hat{y}_n = \hat{y}_n - y_n$, gives

$$F = \frac{1}{\epsilon} \left(\Delta\Lambda_n + \Delta\hat{y}_n \hat{y}_n^T + \hat{y}_n \Delta\hat{y}_n^T + \Delta\hat{y}_n \Delta\hat{y}_n^T \right) \qquad (8)$$

The quantity $\Delta\Lambda_n$ represents the error in the AIEP solution due to quantizing the KLT coefficients. Finding an upper bound on the norm of F is beyond the scope of this paper. However we can exploit Theorem 4 by noting that the perturbation of the eigenvectors in $\hat{W}_n$ is due to the norm of F as well as the separation of the eigenvalues of $\gamma\hat{R}_{n-1}$, the target eigenvalues in the AIEP. Therefore, target eigenvalues that are too tightly clustered can generate unacceptable errors in the eigenvectors $\hat{W}_n$, which can lead to correspondingly large errors in $\hat{Q}_n$ and $\hat{x}_n$. To address this issue, the encoder can modify highly clustered target eigenvalues when it detects an error between x_n and $\hat{x}_n$ that exceeds a given threshold. The following approach was used in the simulations described in the following section. First, compute the normalized difference of the eigenvalues

$$a(k) = \left(\hat{\lambda}_k(n-1) - \hat{\lambda}_{k+1}(n-1) \right) / \hat{\lambda}_k(n-1), k = 1, \ldots, N-1 \qquad (9)$$

Then the minimum of the $a(k), k = 1, \ldots, N-1$, $a(k_{min})$ is found and the $(k_{min}+1)th$ eigenvalue is adjusted as

$$\hat{\lambda}_{k_{min+1}}(n-1) = 0.25\hat{\lambda}_{k_{min+1}}(n-1) + 0.75\hat{\lambda}_{k_{min+2}}(n-1) \qquad (10)$$

The adjustment has the effect of increasing the separation between a pair of clustered target eigenvalues. If $k_{min} = N - 1$, then no adjustments are made. Since the adjustment depends entirely on the target eigenvalues, only one bit of side information is needed in order to inform the decoder that an adjustment must be made. The algorithm currently does not allow for more extensive target eigenvalue adjustments, say, if there are multiple clustered eigenvalues. A listing of the encoder and decoder computations for FAKLT is shown in Table 2.

for $n = 1, 2, \ldots$	E	D
$R_n = \gamma\hat{R}_{n-1} + x_n x_n^T$	•	
$Q_n \Lambda_n Q_n^T = R_n$	•	
$W_n = Q_n^T \hat{Q}_{n-1}$	•	
order columns of Q_n so they agree with $\hat{\Lambda}_{n-1}$	•	
adjust signs of columns of Q_n so that the diagonal entries of W_n are positive	•	
$y_n = Q_n^T x_n$	•	
$\hat{y}_n = \Delta(y_n)$	•	
receive and decode $\hat{y}_n$		•
$c_0 = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^T$	•	•
$c_1 = \begin{bmatrix} \gamma\hat{\lambda}_1(n-1) & \ldots & \gamma\hat{\lambda}_N(n-1) \end{bmatrix}^T$	•	•
for $t = 0, \delta t, \ldots, 1$	•	•
$k = 1$	•	•
while $\lVert\tilde{\Lambda}_n - \gamma\hat{\Lambda}_{n-1}\rVert > \delta$	•	•
compute $f_n(c_k)$	•	•
$B = diag(c_k) - t\hat{y}_n\hat{y}_n^T$	•	•
$\hat{W}_n \tilde{\Lambda}_n \hat{W}_n^T = B$	•	•
$J_{ik} = \hat{w}_{i,n}(k)^2, i, k = 1, \ldots, N$	•	•
solve $J(c_{k+1} - c_k) = f_n(c_k)$ for c_{k+1}	•	•
$k++$	•	•
end while	•	•
end for	•	•
$\hat{\Lambda}_n = diag(c_{k+1})$	•	•
$\hat{Q}_n = \hat{Q}_{n-1}\hat{W}_n^T$	•	•
$\hat{x}_n = \hat{Q}_n \hat{y}_n$	•	•
if $\lVert x_n - \hat{x}_n \rVert > \epsilon$, adjust $\hat{\Lambda}_{n-1}$, repeat computations for frame n, and set adjustment bit	•	
encode and transmit $\hat{y}_n$ and adjustment bit	•	
$\hat{R}_n = \gamma\hat{R}_{n-1} + \hat{x}_n\hat{x}_n^T$	•	•

Table 2. Computations for forward adaptive KLT (FAKLT). Encoder (E) and decoder (D) computations are indicated by bullets. The decoder computations would typically be computed at least one full frame after the encoder computations.

5. SIMULATIONS

A simulation comparing forward and backward adaptive KLT (FAKLT and BAKLT, respectively) was performed. The encoded signal consisted of two time-varying sinusoids: $x(n) = \cos(\omega_1 n) + \cos(\omega_2 n)$ where $\omega_1 = 0.3\pi\cos(0.0001\pi n)$ and $\omega_2 = 0.7\pi\cos(0.0001\pi n)$. The length of each frame was $N = 8$, and FAKLT and BAKLT were run for a total of 1000 frames with $\gamma = 0.98$. A uniform midtread quantizer with a step size of 0.001 was used to quantize the transform coefficients. The initial value of the sample autocorrelation matrix was set to $\hat{R}_0 = 200I_N$, where I_N is the identity matrix. Only a single target eigenvalue adjustment was allowed for each frame. Both FAKLT and BAKLT produced similar errors, the averaged squared error norm for FAKLT was 8.2800×10^{-7} compared to 6.6592×10^{-7} for BAKLT. The slight increase in error exhibited by FAKLT is most likely due to errors in $\hat{Q}_n$ as discussed in Section 4. The square of the quantized transform coefficient vector was averaged over all frames. As shown in Figure 1, FAKLT is seen to provide better energy compaction than BAKLT. Out of the 1000 frames, there were 19 frames that required eigenvalue adjustments.

6. CONCLUSIONS

A framework for signal compression based on a forward adaptive KLT was described. The proposed method solves an inverse eigenvalue problem to compute the KLT basis vectors from the transform coefficients. This eliminates the need to encode the KLT basis vectors. Homotopy continuation was used to insure that Newton's method converges to the correct solution. In simulations, the algorithm was demonstrated to provide better energy compaction than the backward adaptive KLT. A down-side of the proposed method is its high computational complexity, since each Newton iteration requires an eigenvalue decomposition. Complexity can be reduced by reformulating the system of equations in (4) using the formula for the eigenvalues of a diagonal plus rank-1 matrix [11]. Since the matrix $C - t\hat{y}_n\hat{y}_n^T$, is a diagonal plus rank-1 matrix, c can be computed using Newton's method without resorting to an eigenvalue decomposition. The resulting equations also have the same solution paths as (4) and can easily be shown to have the same Jacobian, to within a scalar constant.

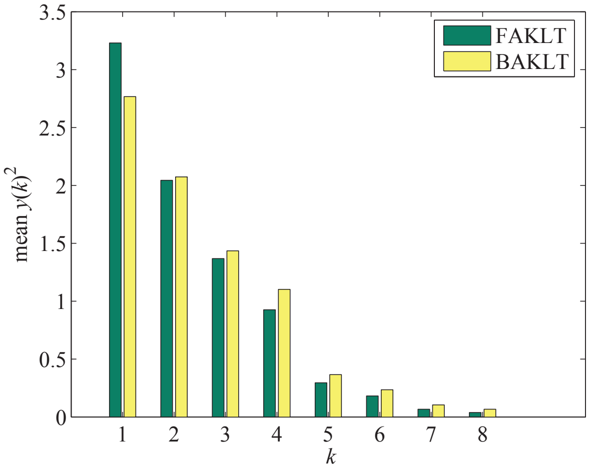

Fig. 1. Averaged squared transform coefficients for FAKLT and BAKLT. FAKLT gives better energy compaction than BAKLT.

7. REFERENCES

[1] V. K. Goyal, J. Zhuang, and M. Vetterli, "Transform coding with backward adaptive updates," *IEEE Trans. on Information Theory*, vol. 46, pp. 1623–1633, 2000.

[2] Michelle Effros, Hanying Feng, and Kenneth Zeger, "Suboptimality of the Karhunen-Loève transform for transform coding," *IEEE Transactions on Information Theory*, vol. 51, pp. 1605–1515, 2004.

[3] Soumya Jana and Pierre Moulin, "Optimality of KLT for high-rate transform coding of Gaussian vector-scale mixtures: Application to reconstruction, estimation, and classification," *IEEE Transactions on Information Theory*, vol. 52, pp. 4049–4067, 2006.

[4] C. E. Davila, "Blind adaptive estimation of KLT basis vectors," *IEEE Transactions on Signal Processing*, vol. 49, pp. 3164–3169, 2001.

[5] D. Mary and D. T. M. Slock, "Rate-distortion analysis of backward adaptive transform coding schemes," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, 2003.

[6] S. Friedland, J. Nocedal, and M. L. Overton, "The formulation and analysis of numerical methods for inverse eigenvalue problems," *SIAM J. Numer. Anal.*, vol. 24, pp. 634–667, 1987.

[7] S. Friedland, "Inverse eigenvalue problems," *Linear Algebera and Its Applications*, vol. 17, pp. 15–51, 1977.

[8] W. I. Zangwill and C. B. Garcia, *Pathways to Solutions, Fixed Points, and Equilibria*, Prentice Hall, 1981.

[9] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, 2006.

[10] M. T. Chu, "Solving additive inverse eigenvalue problems for symmetric matrices by the homotopy method," *IMA Journal of Numerical Analysis*, vol. 9, pp. 331–342, 1990.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 1989.