Characterizing and Estimating Block DCT Image Compression Quantization Parameters

Ramin Samadani
Imaging Systems Laboratory
HP Laboratories Palo Alto
HPL-2005-190
October 20, 2005*

compression, artifact reduction, image processing, sequential estimation

This paper describes an algorithm for estimating quantization matrices from raster images previously compressed with JPEG. First, the space of commonly used quantization matrices is statistically characterized using over 15000 image files. The insights from this characterization are used to design the algorithm. The two stage algorithm first applies a new sequential estimation process to determine some individual Q matrix entries, and then applies shape-gain vector quantization to recover complete quantization matrices. Low average absolute error values are found for 124 images not used during training. In addition, the estimated Q matrices work well when applied to compression artifact reduction.

* Internal Accession Date Only
Published in the Asilomar Conference on Signals, Systems, and Computers, 30 October – 2 November 2005, Pacific Grove, CA, USA
Approved for External Publication

CHARACTERIZING AND ESTIMATING BLOCK DCT IMAGE COMPRESSION QUANTIZATION PARAMETERS

Ramin Samadani

Imaging Systems Laboratory

ABSTRACT

This paper describes an algorithm for estimating quantization matrices from raster images previously compressed with JPEG. First, the space of commonly used quantization matrices is statistically characterized using over 15000 image files. The insights from this characterization are used to design the algorithm. The two stage algorithm first applies a new sequential estimation process to determine some individual Q matrix entries, and then applies shape-gain vector quantization to recover complete quantization matrices. Low average absolute error values are found for 124 images not used during training. In addition, the estimated Q matrices work well when applied to compression artifact reduction.

1. INTRODUCTION

Once an image is decompressed, explicit information about its compression parameters is lost. The compression parameters are useful, however, to: 1) set parameters for compression artifact reduction; 2) avoid loss during recompression; and 3) potentially determine the camera model used during image capture. For example, the cropped original image in the top of Figure 2 contains compression artifacts. The middle image shows results of artifact reduction [1], using the known, true compression parameters. The bottom image shows almost indistinguishable results using compression parameters estimated as described below. The resulting artifact reduction is very similar for all the 124 images tested.

Q matrices are equivalent, when transformed by column-scanning, to $\boldsymbol{q}$ vectors, with components q_i, $i \in \{0 \cdots 63\}$. Known methods for estimating q_i start by re-applying 8x8 forward DCT transforms to non-overlapping image blocks. The resulting coefficients $d_i(k)$, for image blocks k, are, ideally, multiples of the true q_i. Prior work estimated individual q_i values by applying maximum likelihood estimation [2], or other objectives [3], to histograms of $d_i(k)$. As a first step, this paper estimates q_i values using a new sequential method without complete histogram calculation, shown in the top left and the top right dashed block of Figure 1. With any of these methods, the q_i cannot be estimated for some i, because the $d_i(k)$ quantize to zero throughout the image. An extrapolation is needed to recover all the q_i. To my knowledge, no previously proposed methods recover entire vectors $\boldsymbol{q}$ when a number of different types of quantization matrices are candidates for the compression.

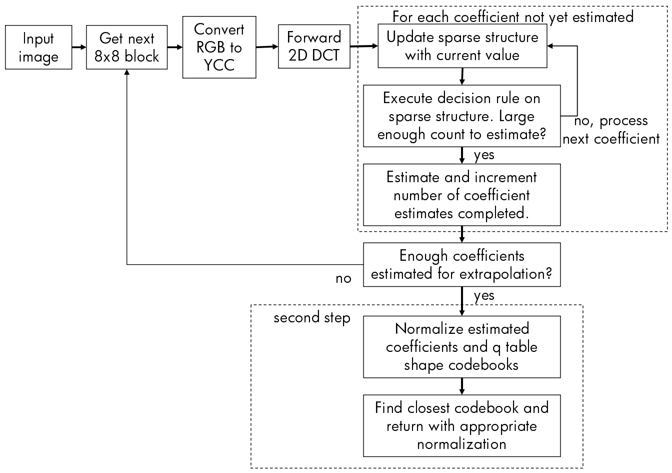
Input image
Get next 8x8 block
Convert RGB to YCC
Forward 2D DCT
For each coefficient not yet estimated
Update sparse structure with current value
Execute decision rule on sparse structure. Large enough count to estimate?
no, process next coefficient
yes
Estimate and increment number of coefficient estimates completed.
Enough coefficients estimated for extrapolation?
no
yes
second step
Normalize estimated coefficients and q table shape codebooks
Find closest codebook and return with appropriate normalization

Fig. 1. Flowchart for the method for the Y component. For the color components, a similar algorithm may be applied with simple modifications that test different possible subsamplings.

Here, a brief overview of the algorithm is given, with details desribed in later sections. Figure 1 shows the flowchart for the method for the Y component. The first steps up to and including the forward 2D DCT occur once for each 8x8 block. The processing in the dashed box on the top right of the figure occurs for each of the 64 DCT coefficients: updating the counts and values of a sparse data structure, deciding based on the magnitudes of the counts, whether to form an estimate for the q_i value or to continue processing. After this loop, a variable that counts the number of coefficients estimated so far is checked against a constant. If there are not enough coefficient estimates, the processing continues at a new image block. If there are enough coefficient estimates, the second and final step conducts a vector quantization decoding. The estimated q_i values from the first step are

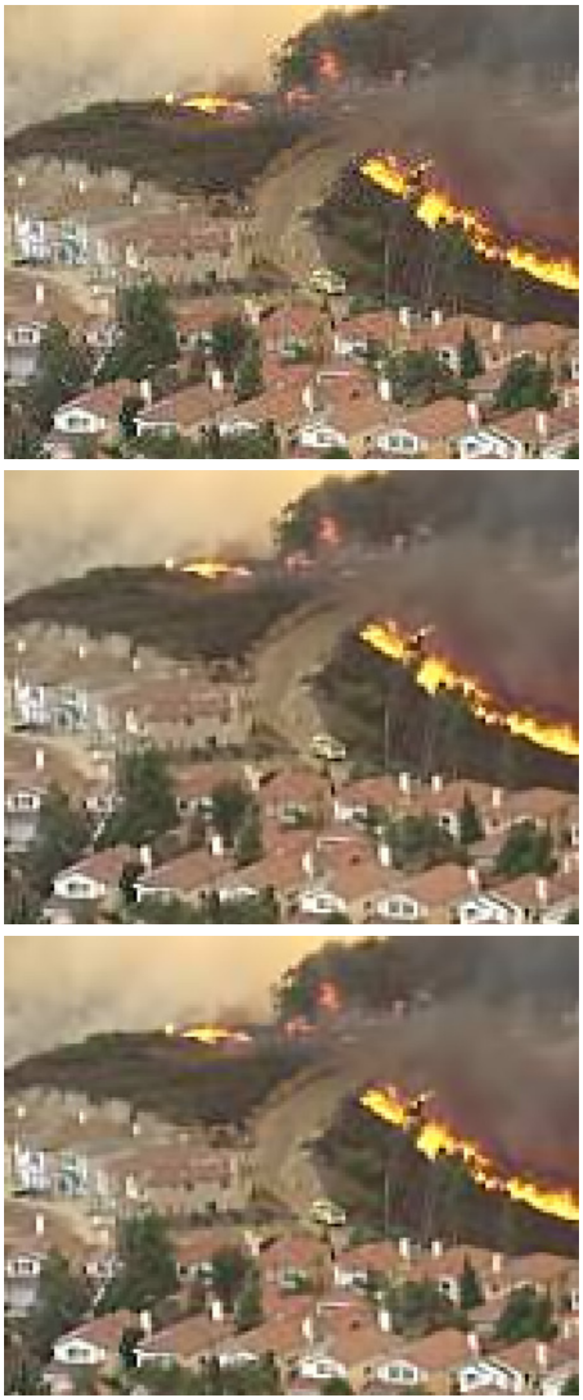

Fig. 2. Image of a fire near a residential neighborhood. Top shows portion of original image with compression artifacts. Middle shows reduced artifacts with known compression parameters. Bottom, indistinguishable to middle, shows reduced artifacts with estimated compression parameters.

shape matched against previously generated VQ codebooks of $\boldsymbol{q}$ vectors (these previous codebooks were generated using the LBG vector quantizer design algorithm on several thousand JPEG $\boldsymbol{q}$ vectors). The closest match is normalized and returned as the final $\boldsymbol{q}$ vector estimate.

Section 2 covers the characterization of quantization matrices, which provided insights during the design of the algorithm. Then, the first stage of the algorithm, the new sequential method for estimating individual estimates q_i, is described in Section 3. Section 4 describes the second stage that uses shape-gain VQ to estimating the entire quantization vector $\boldsymbol{q}$. Finally, Section 5 describes experiments validating the approach.

2. CHARACTERIZING QUANTIZATION PARAMETERS

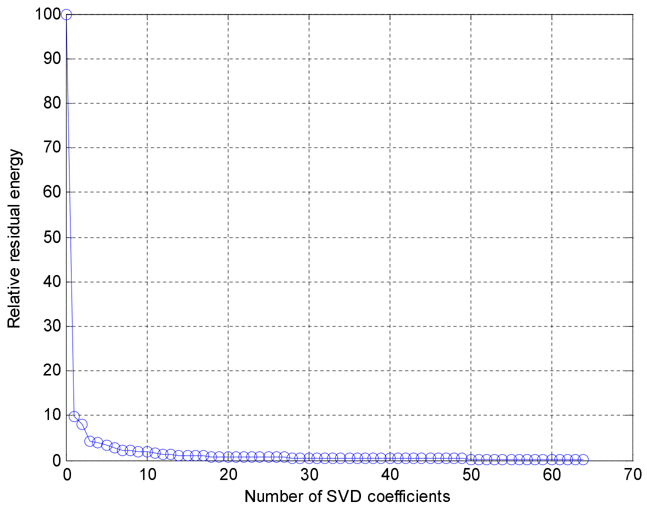

Fig. 3. The relative residual energy as a function of number of singular vectors used to approximate the space of $\boldsymbol{q}$ vectors.

Characterization of the quantization matrices that are used, in practice, is needed. Over 15000 JPEG files, mostly consisting of digital camera images, but also of JPEG files typically found on computer disks, were studied. From these files, 1142 unique $\boldsymbol{q}$ vectors (files originating from the same source often have the same $\boldsymbol{q}$ vector), for the luminance channel, were extracted and used to form a 64 by 1142 matrix M. Then, the singular value decomposition, $USV^T = M$, was computed. Figure 3 shows the relative residual energy, $\frac{\|\hat{M}-M\|}{\|M\|}$, when estimating M using an increasing number of singular vectors (arranged in decreasing order of corresponding singular value magnitude). The figure shows that the first singular value accounts for about 90% of the energy in representing this space. Interestingly, inspection found the first singular vector to be remarkably similar to the JPEG default quantization matrix [4]. Note, however, that there is also energy in some of the remaining singular vectors.

With the knowledge that the first few singular vectors contain most of the energy in representing Q matrices, and the additional knowledge that manufacturers often control compression ratio by scaling the Q matrices, it is natural to apply shape-gain vector quantization [5] (which separates a normalization gain α from a *shape*) to the estimation. This will be described in Section 4. First, however, the next section describes the first stage of the algorithm, the sequential estimation of individual $\boldsymbol{q}$ vector entries.

3. INITIAL SEQUENTIAL ESTIMATION

To estimate an unknown $\boldsymbol{q}$, initial estimates of individual q_i are first found using the method shown in the top left and the top right dashed block of Figure 1. The initial estimate vector, $\hat{\boldsymbol{q}}_1$, has valid estimates for only a subset of components.

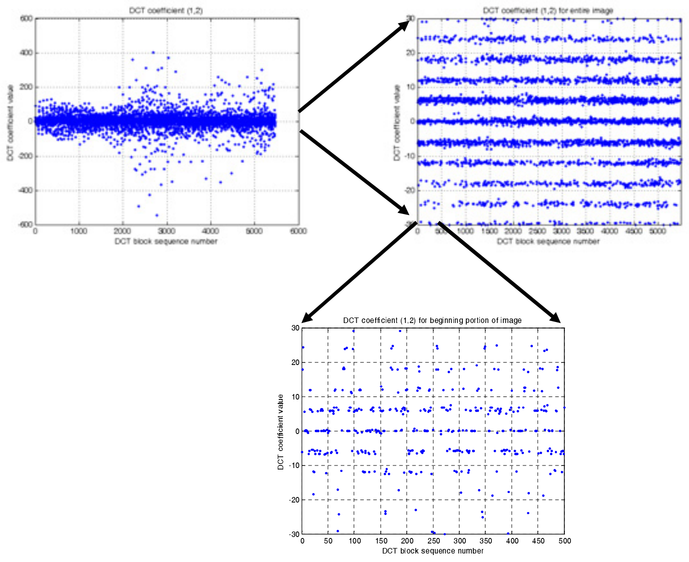

Fig. 4. Reconstructed DCT coefficients as a function of sequential block number. The top right shows a sparse data structure may be sufficient for estimating the q values. The bottom shows that only a fraction of an image may be needed to estimate the q values.

Review of Figure 4 provides motivation for use of a sparse data structure that does not store an entire DCT coefficient histogram, and for the ability to estimate sequentially. The top left of the figure shows the values of a reconstructed DCT coefficient (on the y axis) as a function of sequential block number. The top right of the figure expands the vertical scale, showing that the reconstructed coefficient values cluster at multiples of the true q_i value, and that not all these *horizontal lines* are needed to estimate q_i, so that a sparse data structure that can track peaks may be used instead. In addition, examining the bottom plot of the figure shows that there is enough information to estimate the q_i value even by analyzing only a fraction of the blocks from the image, suggesting the use of a sequential algorithm.

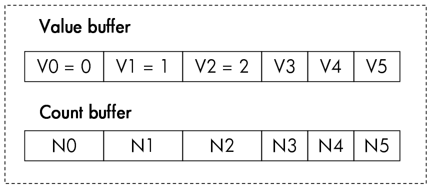

Fig. 5. This sparse data structure, contains nine storage locations (V0, V1 and V2 are implicit and need not be stored) versus the thousand that would be needed when storing entire histograms (for each of 64 coefficients).

Figure 5 shows the sparse data structure used, one for each of the 64 DCT coefficients, meant to track the maxima of the probability distribution function (pdf) for the given coefficient. The structure has six *cells*, each cell with a value and a count. The first three values are *hardwired* to values 0, 1, and 2 and are processed specially by the decision rule (if $v2 > v0$ it is assumed that $q = 1$ or $q = 2$ and additional rules distinguish these two cases). These special values handle the specially difficult case of fine quantization when the true q_i value is very small. The other values, $v3$, $v4$ and $v5$, *float*: they may take any values that occur in the data stream. A replacement rule updates the data structure at each block. If the current data value occurs in the table, its count is incremented. Otherwise, the current data value and the count value 1 replaces the cell with the smallest count. In practice, this procedure finds and keeps the maximum non-zero value of the pdf, which because of the statistics of DCT coefficients of images, most of the time corresponds to an estimate of q_i itself.

The decision rule in the case of large q_i picks the maximum value. With this sparse structure approach, the q_i value is estimated without storing or processing the 1000 element histograms for each coefficient, making the method fast in software and efficient in hardware storage.

4. SHAPE VQ FOR THE FINAL ESTIMATION

Given the first stage estimates of some of the $\boldsymbol{q}$ vector entries, the second stage recovers the complete $\boldsymbol{q}$ vector.

First, based on the insight from Section 2 that a few of the singular vectors have energy, during training, normalized $\boldsymbol{q}$ vectors were input to the LBG algorithm [5], to generate five representative, normalized *codeword* vectors, $\boldsymbol{c_j}$, $j \in \{0 \cdots 5\}$.

After the completion of the first stage of the algorithm, the initial subset of estimates $\hat{\boldsymbol{q}}_1$ are used to form final estimate, $\hat{\boldsymbol{q}}_f$. The shape-gain decoding shown at the bottom dashed block of Figure 1, applied to $\hat{\boldsymbol{q}}_1$, is now described.

A diagonal weight matrix, $\boldsymbol{W}$, with

$$w_{ii} = \begin{cases} 1 & \text{if } q_i \text{ initial estimate exists} \\ 0 & \text{if } q_i \text{ initial estimate does not exist} \end{cases} \tag{1}$$

accounts for the fact that not all the q_i have initial estimates. Given initial estimate, $\hat{\boldsymbol{q}}_1$, for a given a codeword $\boldsymbol{c}_j$, the modified shape-gain distortion is defined by,

$$D(\hat{\boldsymbol{q}}_1, \alpha \boldsymbol{c}_j) = \{(\hat{\boldsymbol{q}}_1 - \alpha \boldsymbol{c}_j)^T \boldsymbol{W} (\hat{\boldsymbol{q}}_1 - \alpha \boldsymbol{c}_j)\}, \tag{2}$$

The value of α at the minimum cost may be shown to be,

$$\alpha_j^* = \frac{\boldsymbol{c}_j^T \boldsymbol{W} \hat{\boldsymbol{q}}_1}{\boldsymbol{c}_j^T \boldsymbol{W} \boldsymbol{c}_j}. \tag{3}$$

With these value for α, the final estimate $\hat{\boldsymbol{q}}_f = \alpha_k^* \boldsymbol{c}_k$ is given by the codeword k with minimum $D(\hat{\boldsymbol{q}}_1, \alpha_k^* \boldsymbol{c}_k)$ in Equation 2.

5. RESULTS

The Q matrix estimation was applied to 124 images that were not used for training, and for which the true Q matrices were known. Statistical results are shown in Figure 6. The top of the figure shows averages of the absolute errors (absolute difference between estimate and true q_i) for each of the 64 Q matrix coefficients estimated. The bottom of the figure shows the estimated standard deviation of the mean absolute error, computed using the bootstrap method.

For illustration, examples of the estimation applied to two previously compressed images are now presented. For the first image, the known true Q matrix, shown on the top of Figure 7, is similar to the default JPEG table. The results of the first step of the estimation are shown in the middle of the figure, with zeros representing values without estimates. The bottom of the figure shows the final estimate, $\hat{Q}_f$, that is very close to the original. The second example, shown in Figure 8, shows the result for a known true Q matrix that is constant. In this example, the true Q matrix is exactly recovered.

As previously mentioned, compression artifact reduction was conducted using both the known Q matrices and the estimated ones, with very similar results in all cases. Also, the same approach was used to characterize the quantization matrices of the color components as well, with the added complication being the unknown subsampling factor of the color components.

$$\hat{\mu} = \begin{pmatrix} 0.96 & 0.39 & 0.40 & 0.31 & 0.26 & 0.26 & 0.67 & 0.67 \\ 0.34 & 0.35 & 0.32 & 0.43 & 0.31 & 0.51 & 0.44 & 0.70 \\ 0.31 & 0.43 & 0.30 & 0.27 & 0.26 & 0.71 & 0.91 & 0.32 \\ 0.32 & 0.54 & 0.23 & 0.47 & 0.66 & 1.04 & 0.65 & 0.39 \\ 0.21 & 0.22 & 0.63 & 0.40 & 0.49 & 1.25 & 1.24 & 0.80 \\ 0.26 & 0.26 & 0.74 & 0.46 & 0.84 & 0.83 & 1.30 & 0.71 \\ 0.72 & 0.52 & 0.61 & 0.99 & 1.25 & 1.35 & 1.09 & 1.10 \\ 0.51 & 0.82 & 1.11 & 0.85 & 0.96 & 0.77 & 1.22 & 1.02 \end{pmatrix}$$

$$\hat{\sigma} = \begin{pmatrix} 0.10 & 0.05 & 0.05 & 0.04 & 0.04 & 0.09 & 0.14 & 0.16 \\ 0.04 & 0.04 & 0.04 & 0.05 & 0.05 & 0.19 & 0.16 & 0.11 \\ 0.04 & 0.05 & 0.04 & 0.05 & 0.08 & 0.15 & 0.19 & 0.08 \\ 0.04 & 0.05 & 0.04 & 0.06 & 0.13 & 0.30 & 0.20 & 0.07 \\ 0.04 & 0.04 & 0.08 & 0.15 & 0.17 & 0.37 & 0.31 & 0.13 \\ 0.04 & 0.06 & 0.16 & 0.14 & 0.20 & 0.31 & 0.33 & 0.21 \\ 0.12 & 0.19 & 0.23 & 0.26 & 0.32 & 0.39 & 0.37 & 0.27 \\ 0.22 & 0.32 & 0.32 & 0.29 & 0.34 & 0.26 & 0.29 & 0.26 \end{pmatrix}$$

Fig. 6. The top shows the average of the absolute error for 124 images, and the bottom shows the standard deviation (computed using bootstrap) of the average absolute error.

6. REFERENCES

[1] R. Samadani, A. Sundararajan, and A. Said, "Deringing and deblocking DCT compression artifacts with efficient shifted transforms," in *Proceedings of the International Conference on Image Processing (ICIP)*, 2004.

[2] Zhigang Fan and Ricardo L. de Queiroz, "Identification of bitmap compression history: Jpeg detection and quantizer estimation," *IEEE Transactions on Image Processing*, vol. 12, no. 2, pp. 230–235, February 2003.

[3] Jessica Fridrich, Miroslav Goljan, and Rui Du, "Steganalysis based on jpeg compatibility," in *SPIE Multimedia Systems and Applications IV*, August 2001, pp. 275–280.

[4] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*, Springer, 2nd edition, 1997.

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

$$Q = \begin{pmatrix} 8 & 6 & 5 & 8 & 12 & 20 & 26 & 31 \\ 6 & 6 & 7 & 10 & 13 & 29 & 30 & 28 \\ 7 & 7 & 8 & 12 & 20 & 29 & 35 & 28 \\ 7 & 9 & 11 & 15 & 26 & 44 & 40 & 31 \\ 9 & 11 & 19 & 28 & 34 & 55 & 52 & 39 \\ 12 & 18 & 28 & 32 & 41 & 52 & 57 & 46 \\ 25 & 32 & 39 & 44 & 52 & 61 & 60 & 51 \\ 36 & 46 & 48 & 49 & 56 & 50 & 52 & 50 \end{pmatrix}$$

$$\hat{Q}_1 = \begin{pmatrix} 0 & 6 & 5 & 8 & 12 & 20 & 26 & 0 \\ 6 & 6 & 7 & 10 & 13 & 29 & 0 & 0 \\ 7 & 7 & 8 & 12 & 20 & 0 & 0 & 0 \\ 7 & 9 & 11 & 15 & 26 & 0 & 0 & 0 \\ 9 & 11 & 19 & 28 & 0 & 0 & 0 & 0 \\ 12 & 18 & 28 & 0 & 0 & 0 & 0 & 0 \\ 25 & 32 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\hat{Q}_f = \begin{pmatrix} 8 & 6 & 5 & 8 & 12 & 20 & 26 & 31 \\ 6 & 6 & 7 & 10 & 13 & 29 & 30 & 28 \\ 7 & 7 & 8 & 12 & 20 & 29 & 35 & 28 \\ 7 & 9 & 11 & 15 & 26 & 44 & 40 & 31 \\ 9 & 11 & 19 & 28 & 34 & 55 & 52 & 39 \\ 12 & 18 & 28 & 32 & 41 & 53 & 57 & 47 \\ 25 & 32 & 39 & 44 & 52 & 61 & 61 & 51 \\ 36 & 47 & 48 & 50 & 57 & 51 & 52 & 50 \end{pmatrix}$$

Fig. 7. First example: Original Q, initial estimate $\hat{Q}_1$, and final estimate $\hat{Q}_f$.

$$Q = \begin{pmatrix} 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \end{pmatrix}$$

$$\hat{Q}_1 = \begin{pmatrix} 0 & 30 & 30 & 30 & 30 & 30 & 30 & 0 \\ 30 & 30 & 30 & 30 & 30 & 30 & 0 & 0 \\ 30 & 30 & 30 & 30 & 30 & 30 & 0 & 0 \\ 30 & 30 & 30 & 30 & 30 & 0 & 0 & 0 \\ 30 & 30 & 30 & 0 & 0 & 0 & 0 & 0 \\ 30 & 30 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\hat{Q}_f = \begin{pmatrix} 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \\ 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 \end{pmatrix}$$

Fig. 8. Second example: Original Q, initial estimate $\hat{Q}_1$, and final estimate $\hat{Q}_f$.