Compressing Genomic and Proteomic Array Images for Statistical Analyses

Rebecka Jörnsten and Bin Yu
Departments of Statistics,
Rutgers University, USA, and
University of California, Berkeley, USA

Abstract Information technology advancements are bringing about innovations for genomic and proteomic research. One of them is the array imaging technology based on which gene or protein expression levels are derived. These images have a fundamentally different purpose to serve than the traditional still images: they are for statistical information extraction, not for visual inspection or comparison. Due to the huge quantity of such images and the limited bandwidth for their sharing among different researchers, for both storage and transmission goals, these images need to be compressed. Dictated by the statistical analyses to follow, in this chapter we lay out the multi-layer data structure as the principle for both lossless and lossy compression of array images. We illustrate this principle in the example of cDNA microarray image compression with results of an average of near 2:1 lossless compression ratio and an average of 8:1 lossy compression ratio. The lossless ratio is comparable with the off-the-shelf lossless compression scheme LOCO, but with the added benefit of a handy structure for statistical analysis; the lossy ratio is obtained with a quantization noise level comparable to that of the imaging technology or the variation between two replicate imaging experiments.

I. Introduction

We live in an exciting era of technology innovations with all their advantages (and disadvantages). These innovations are fueling, if not driving, the progresses in genomic research (the study of genetic material such as DNA and RNA), and the newer proteomics research (the study of proteins which are directly responsible for actions in cells).

A revolutionary innovation has been the DNA microarray imaging technology for genomic research and it takes different forms: cDNA [4], Affymetrics gene chips [3], and Inkjet [12].It provides measurements of mRNA (messenger RNA) material existing in cells to develop an understanding of gene function, gene regulation, and gene interaction through a simultaneous study of expression levels of thousands of genes. Microarrays are also used extensively in clinical research. The goal is to identify disease genes, develop treatments, and diagnostic tools. The purpose of a microarray image is not for visual inspection as in still and video image signal processing, but for extraction of statistical information regarding the gene expression levels. And the statistical inference problems based on extracted gene expression data are plentiful, and often non-trivial. Among these problems are the identification of important genes, and groups of genes that may be linked in terms of functionality.

We have recently seen the emergence of the new research field called Proteomics, where the focus is shifted from mRNA measurements to proteins [2]. Even though mRNAs carry the genetic instructions to produce proteins, i.e. an indirect measurement of protein levels, the proteins themselves are the direct agents to make cells function. Hence direct measurements of proteins are believed to lead to a better understanding of biologically related events in organisms. To measure the protein amount, antibody materials are put in the spot wells on an array for proteins to bind. This protein array technology is still in its infancy and is more complicated than the DNA microarray technology due to the inherent properties of proteins. One major difficulty lies in the logistics and practical procedures in generating thousands of high-quality probes which have been successfully produced for DNA microarrays. Despite the fact that DNAs (mRNAs) and proteins are very different materials, the protein array images share similar characteristics of DNA microarray images and the same statistical information extraction is the purpose, not visual inspection. In Fig. 1 (c) we display a small portion of a 10 MB protein array image, kindly provided to us by Dr. Claudio Caamano, Mental Health Research Institute, University of Michigan.

Both DNA and protein array images contain expression spots laid out in a grid (see Fig. 1). At the onset of an array experiment the spots contain the known mRNA or antibody materials from DNA or antibody libraries. At the conclusion of the experiment, the corresponding materials from sample cells are added to the spots and allowed to hybridize. Through this technique we can detect the amounts of mRNA or protein in the sample cells. The finished arrays are then scanned to produce array images usually in a 16 bits/per pixel format. All these array images, DNA or protein, are very large (tens of Mb). We see an ever increasing demand on using array technology for genomic or proteomic research in universities, research institutions, and private companies. This increase results in a huge quantity of raw array images that researchers store after a certain processing to extract statistical information. Storage is necessary since the images are expensive to obtain and

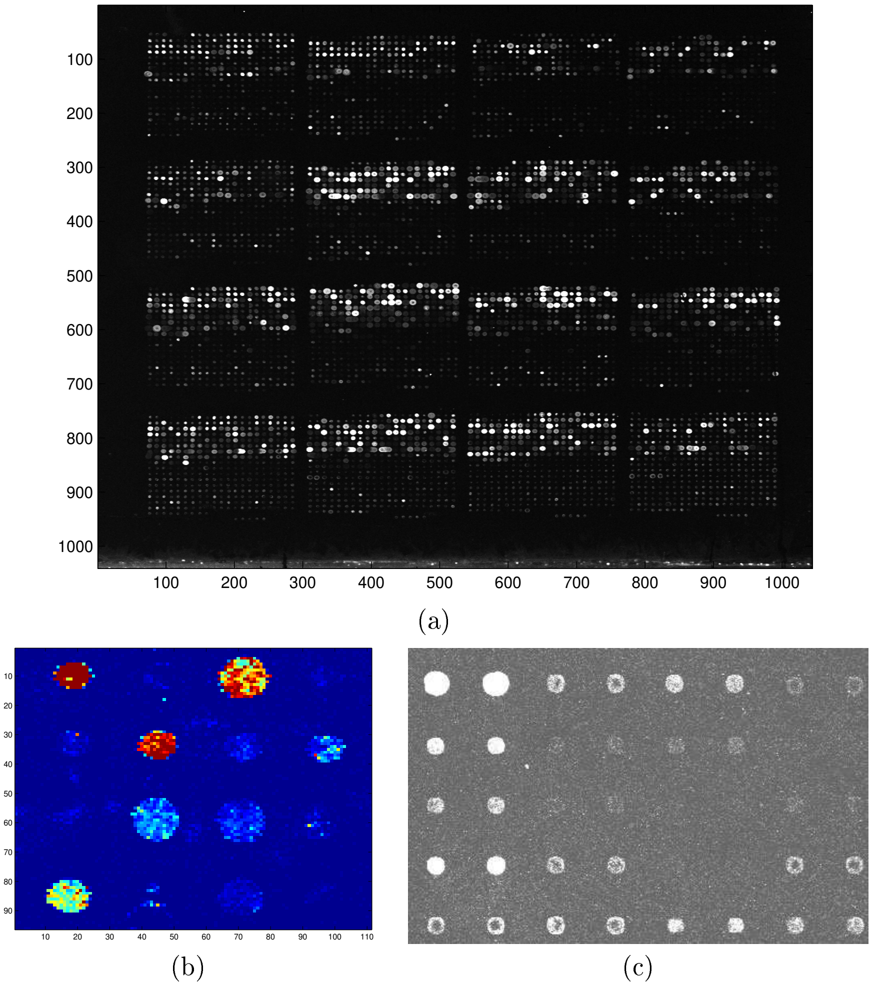


Fig. 1. (a) Microarray image: 4×4 grid set-up, 19×21 spots per grid. (b) A subset of the same image. (c) A subset of a protein array image.

the processing techniques are still under development. It is safer to keep the raw images for possible reprocessing and improved information extraction as and when new processing techniques become available. Since the quantity of such images is huge, it is well worth the efforts to compress them before storage amid the ever falling hard disk price. To best facilitate the statistical analyses downstream, serious considerations are required to address the question of how to compress, if lossless, and how much and what to take out, if lossy. This chapter proposes principles to answer these questions and illustrates the principles in the example of cDNA microarray image compression.

The rest of the chapter is organized as follows. In Section 2 we put forward the importance of a multi-layer data structure to ease statistical analysis. Lossy and lossless compression is dealt in Section 3 which advocates a multi-layer progressive data structure for compressed images. We propose using a variation measure between two replicated images as the desirable quantization noise level in lossy compression, and a partially progressive scheme for lossless image reconstruction. In Section 4, all the proposals are implemented in the example of cDNA microarray image compression ([13],[14]). For a set of microarray images kindly provided by M. Callows via T. Speed's group ([8], [9], [29]) we obtained an average of 2:1 compression ratio for lossless compression and an average of 8:1 compression ratio for a multi-layer progressive lossy scheme with an quantization error comparable to the average difference between two replicated images.

II. Lossy and Lossless compression through a multi-layer data structure

Images can be completely recovered through lossless compression. Therefore, no one should argue that it is not needed in the face of the facts that the savings on the cost of disks to store raw array images could easily run in millions of dollars across different users, and transmission bandwidth is always limited if the images are to be shared by different research groups through internet or a centralized array image library. Even the off-the-shelf lossless compression tools such as LOCO [17] are very useful. Usually it can cut down the cost to half for the current generation of cDNA microarray images. There is not much hope for a lossless compression ratio exceeding 2:1 due to the fact that the last 8 bits of each pixel value contains mainly noise [13].

This treatment of array images as if they are still images does not make the uncompression of them easy for statistical analysis or partial reprocessing later. The whole image has to be uncompressed for a researcher to get the gene expression levels or later go back to look at even one gene or antibody spot to revise the estimated gene expression level when necessary. As discussed in [13], one would like to have the compressed image in a data structure that is easy for downstream statistical analysis and possible partial reprocessing at a later time.

So what is a desirable data structure for statistical analysis? First of all, any statistical analysis based on array images needs expression levels of the mRNAs or proteins. These levels are obtainable from the intensity readings of the images (which are related to the amount of dyed mRNA or protein materials) in the mRNA or antibody spots. To get these expression levels, one has to decide where the spots are, i.e. a segmentation of the image is needed. Furthermore, the non-spot regions of the images also contain important information. The background part is used for local estimation of intensity drifts which could influence the expression level estimation in a systematic way.

A Multi-Layer Data Structure for Compression

- The most needed statistical information should be the first layer in the data structure, that is, the estimated expression levels for each spot (mRNA/protein). For quality control purposes we also include spot standard deviations in this layer. The standard deviations are measures of spot variability or heterogeneity.

- The second layer of the data structure should contain the segmentation map identifying the spot regions.

• The third layer has the entropy coded intensities in the spot regions, to a chosen level of precision (lossy compression).

• The fourth layer has the run-length coded intensities in the background regions, to a chosen level precision (lossy compression).

• The last layer has the residual information, for adaptive refinement of spot and background regions to any level of precision (progressive lossy-to-lossless compression).

A statistician in possession of a compressed image in terms of the data structure as described above would have a quick access to the expression levels from the first data layer and some rough variability measures. If she or he is curious about how good the quality of the image is, the segmentation map at the second layer would give some important clues (smooth spot boundaries indicate good quality and rigged ones poor quality). If an expression level is unusual, it calls for the inspection of the corresponding spot and this doesn't require the decoding of the whole image, only the object corresponding to that spot. For a systematic drift worry, she or he could zoom into the fourth layer directly and uncompress only that part.

A. Lossless compression

For lossless reconstruction the researcher can decode all layers. Moreover, the lossless compression ratio for each array is a measure of array data quality. Noisy arrays have highly random least significant bits, which results in a higher lossless bit-rate. Therefore, the file sizes of the compressed images can be used to sort the images in terms of data quality.

B. Lossy compression: what to take out, how, and how much

Since the multi-layer coded data structure was dictated by the statistical analysis, a useful lossy compression scheme should follow the same structure. Obviously, the first two layers can be done at a very low bit rate and so crucial to the statistical analysis to follow and therefore they should be kept as they are. The next two layers are subject to quantization or lossy compression. Acceptable loss is not readily defined for array images. Recall that these images are not preserved for the purpose of visual comparison, but for statistical information extraction and processing. We define acceptable loss as the level of replicate experiment variation, i.e. the level of noise (scanner noise and otherwise) in the images.

In the following section we illustrate our structured lossless and lossy compression scheme on cDNA microarray images, but emphasize the fact that similar approaches are also appropriate for other types of array experiments.

III. An example: cDNA microarray image compression

The cDNA microarray image technology is a tool geared at measuring the "activity" or expression of a gene. A gene is a segment of DNA that maps into a specific protein. The expression of a gene is a two-stage process whereby the protein product is created. The first stage is *transcription* where the DNA segment is translated into a messenger (m)RNA. The second stage is *translation* where mRNA is used as a blueprint for a protein. Microarray experiments measure the level of activity of a gene at the first stage. Thus, the abundance of mRNA in a cell is related to the amount of the corresponding protein being produced. We measure the abundance of mRNA in a specific sample, *relative* to another sample. DNA probes (each corresponding to a gene, or DNA segment) are placed, or "spotted" onto a microscopic glass slide by a robotic *arrayer.* A reference sample of mRNA is labeled with a green fluorescent dye (Cy3). The sample of interest is labeled with a red dye (Cy5). The two mRNA samples are mixed and allowed to hybridize onto the array. The relative mRNA abundance (for each probe) is measured through the competitive hybridization of the two samples. A laser scan of the array produces two fluorescent intensity images. The intensity ratio for each probe, or spot, is proportional to the relative abundance of mRNA in the two samples. The raw microarray image data thus consist of two high precision (16 bpp) scans. The images are structured, with high intensity spots (corresponding to the probes) located on a grid (see figure 1). The spots are submerged in a noisy and non-stationary background. The spots have roughly circular shape. The background (non-spot regions) can be corrupted by high-intensity speckle noise from dust particles, or water droplets on the glass slide. Spots may "bleed" into each other or be smeared due to imprecision in the spotting procedure, or through "washing-out artifact", as excess sample material is removed from the array prior to scanning.

A. Genetic information extraction

Since relative mRNA abundance is measured in microarray experiments, the genetic information quantity available is the *differential gene expression* between the two samples. In order to accurately estimate this quantity, we have to identify the high intensity regions in the images corresponding to each probe, and where hybridization has occurred. Moreover, we have to estimate, and correct for, the local background intensity or noise level. Various methods for image segmentation, and background correction are used in the processing of microarray images. These methods have had variable success, depending on noise level, average spot sizes and distances, of the arrays (e.g. [29], [7]).

Segmentation. Automatic registration of the image is used to determine the approximate centers, or the grid location, of the spots. The spots are somewhat circular. The most simplistic approach to identify the regions where hybridization has occurred, is through a fixed circle segmentation [7]. A circle, with radius chosen from the estimated grid structure, is used to define the spot regions. The apex of the circle is adjusted locally to maximize the summed signal intensity within the circle. When spot sizes vary across the array (see Fig. 1), which can result from e.g. variations in print-tip pressure, an adaptive circle segmentation is more appropriate [10]. This allows for differences in spot radii, and can significantly improve the identification of spots. When spot shapes are highly non-circular, adaptive shape segmentation techniques such as seeded region growing can improve spot regions identification further still (e.g. [29]). Here, we use a seeded region growing algorithm for initial segmentation, followed by a two-component Gaussian mixture model fit, to further refine the boundaries of the spots, or regions of interest (ROI). Seeds for background and ROIs are available from the registration procedure.
Background Correction. Obtaining estimates of the local background intensity level can be a difficult task. When the arrays are noisy, and the spots are positioned close together, the background estimates will often be highly variable. Background es-

timates are commonly obtained through sampling of the background regions, e.g. by looking at disk shaped regions near the identified ROIs [7], or in the "valley-between-peaks" (regions that are the most distant from the center of gravity of all surrounding ROIs) [10]. These methods work well if the spots are clearly separated, but may otherwise perform poorly. A more robust background estimation scheme is presented in [29], and relies on filtering operations (erosion and dilation). This method exhibits low variance, but tends to underestimate the local background intensities.

Summary Statistics. The summary statistic of main interest is the estimated differential gene expression, commonly measured on a $\log_2$-scale. Pixels in each image are summed within each ROI. We denote by R_i the red (fluor tag) scan pixels, and by G_i the green scan pixels. The differential expression level, $\log_2(\mu_R/\mu_G)$, is then calculated as the log-ratio of the mean ROI intensities:

$$\log_2(\frac{\mu_R}{\mu_G}) = \log_2(\frac{\frac{1}{S}\sum_{R_i \in ROI} R_i - \sigma_R}{\frac{1}{S}\sum_{G_i \in ROI} G_i - \sigma_G}),$$

where σ refers to the estimates of the local background, and S is the number of ROI pixels. The log-ratio, commonly referred to as M, $M = \log_2(\mu_R/\mu_G)$ is used in downstream statistical analyses, such as clustering and classification.

Other summary statistics of interest are measures of quality, such as spot variances, spot shapes, and product intensities $A = \log_2 \sqrt{\mu_R \mu_G}$. The product intensities, A, are often indicative of how reliable the measurements of the differential gene expressions are.

Normalization. Normalization is necessary prior to downstream analysis, since otherwise systematic variation in the data may dominate over chance variation. The fluorescent dyes used to label the two samples have different labeling efficiencies, and there is a dye bias in scanning efficiency. Furthermore, there are spatial systematic errors, e.g. a print-tip in one part of the array may show a significantly higher red intensity than in other parts of the array.

It is common to use the genes that show little differential variation between samples for normalization. In some experiments, most genes are not differentially expressed, and all spots can be used for normalization. In other experiments, a set of housekeeping genes are used. M and A are simply a coordinate transformation of the background corrected spot intensities μ_R and μ_G. Dye-bias normalization is usually conducted on M, with respect to A.

Normalization is still an active area of research. The challenge remains to remove systematic bias while keeping increased variance at bay. In addition, normalization procedures can in some cases introduce bias for low-expression spots. Here we apply the normalization scheme of Dudoit et al [8]. The normalization is non-linear, and print-tip specific. For each print-tip on the array (one of the 4×4 sub-grids of Fig. 1), we estimate a locally linear fit of M on A. The residual vector $\tilde{M}$ is computed, and used as the spatial and dye-bias corrected differential gene expressions in subsequent analysis.

In the subsequent statistical analysis, e.g. identification of important genes, both $\tilde{M}$ and A may play a role. An overview of some methods for gene identification based on $\tilde{M}$ only, or both $\tilde{M}$ and A, can be found in [8] (see also [25], [6], [18]). Classification and clustering of gene expressions, and samples uses the information in $\tilde{M}$ only. Since image processing, background correction, normalization and analysis methods are still under development An image compression scheme for microarray images evidently has to preserve information on both M, and A.

B. Lossy and lossless compression of cDNA microarray images

Lossless compression of microarray images is easier for experimentalists to accept. However, for efficient transmission of image data for data sharing, we need to consider the use of lossy reconstructions of the images for subsequent analysis. Not all processing steps require lossless image reconstructions. In addition, the equivalence of compression and denoising suggests that we can obtain improved genetic information extraction from the images with lossy reconstructions (section IV). Segmentation is a relatively easy task and can be done on crude 8bpp reconstructions ([13]). Background correction and normalization are more sensitive to information loss, especially in the region of low intensity spots. A successful compression scheme thus has to keep more precision for low intensity spot regions, but can use a coarse image reconstruction for high intensity spot regions. Note that the commonly used criterion for lossy compression, mean squared error, does not reflect this requirement.

The variance introduced by compression is much more of a concern than marginal bias in multi-step processing (segmentation, background correction, normalization). Our aim is to keep both bias and variance under control, and ensure that the effect of compression is smaller than the variability between replicated experiments. We define this as *acceptable loss* for microarray image compression. For our sample image data set this acceptable loss corresponds to a bit-rate of ~ 4 bpp (compared to 32 bpp with no compression) (section IV).

Lossless and lossy compression of natural and medical images is a mature field in the engineering sciences. However, the performance of state-of-the-art image compression schemes on microarray data is poor. Here we mention some of the issues that make the compression of microarray images a particularly difficult problem. Microarray images are very noisy, more so than images in many medical applications. We can compare the lossless compression rate of microarray images to those reported for mammograms. Lossless compression ratios of mammograms are $\sim 3 : 1$, with the very efficient SPIHT algorithm ([24], [23]), whereas only $\sim 1.6 : 1$ for microarray images. In medical imaging the final task is visual inspection. Lossy compression of medical images thus often takes the approach of defining a region of interest (ROI), within which high precision lossy reconstructions are used. Outside the ROI, a coarse image reconstruction is used, and this leads to considerable savings in overall bit-rate. For microarray images, the non-ROI regions cannot easily be discarded as less informative, or be preserved at an arbitrary low bit-rate. These regions are used for background intensity estimation. In medical imaging the ROIs are usually large, few (1-2), and arbitrarily picked by the experimentalists (a square or circle centered at a ROI). ROIs in microarray images are the spot regions. These are defined by the segmentation, and have distribution characteristics that differ from the background. There are many ROIs (thousands) for each microarray image. They are located close together, and can be of arbitrary shape. They are small, with an approximate diameter of 8-16 pixels. The small ROIs precludes the use of wavelet based compression schemes. The many and small high intensity regions create large wavelet coefficients over almost all the image subbands. At low bit-rates, algorithms such as SPIHT, or wavelet and zero-tree coding [24] will be dominated by the edges around the high intensity spots. Compression schemes that are not wavelet based, but based on predictions in the spatial domain, also have difficulty with the many high intensity spots. A rowscan based prediction scheme creates a "smearing" bias in the image reconstruction. Thus, it is a non-trivial task to employ the principles of lossless-ROI coding, or coding of ROI and non-ROI at different precision, used in medical imaging, to microarray images ([27], [23]).

To ensure that lossy reconstructions of the images are informative, we choose to take a spatial prediction approach. To avoid the "smearing" bias we encode the spot regions, and the background separately. We first transmit an *overhead* defining the ROI and background, i.e. a segmentation map. We refer to this approach as segmented LOCO (SLOCO, [14]).

Segmented LOCO - SLOCO

Our scheme builds on the JPEG2000 lossless standard, LOCO (LOw COmplexity), [17]. A low complexity scheme is preferable for this application. The characteristics of microarray images can be very different. To ensure that the compression scheme performs reasonably well for images from many labs, simplicity is key. Below we outline the components of LOCO and SLOCO (details can be found in [17] and [13]).

Fixed and Adaptive Prediction, Context Modeling. The LOCO algorithm uses a robust pixel domain prediction scheme, e.g. the causal context (figure III-B) of pixel x is used to predict its value. The initial fixed predictor equals

$$\hat{x}_{fix} = median(a, b, a + b - c).$$

It works on the same principle as a simple edge detector. Note, that a, b and c are *lossy* reconstructions of the corresponding pixel values. We have briefly studied the use of more complex predictors for microarray images, but have seen no gain with the use of a more complex predictor. The different noise level and array characteristics of arrays from various labs makes the building of robust predictors with more structure (e.g. gradient,surface fitting component) extremely difficult.

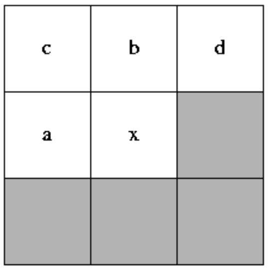


Fig. 2. Causal context of pixel x (a,b,c,d)

To further improve on the fixed prediction scheme, LOCO also uses a context based *adaptive predictor*. The contexts are defined by the vector of local gradients. The local gradients, $g1 = d - b, g2 = b - c, g3 = c - a$, are quantized to e.g. 16 levels. Each triplet of quantized gradients forms a context class. Based on the past performance of the fixed predictor, within each context class, an estimate of the prediction bias, $\hat{R}$, is obtained. The bias is taken as integer valued and estimated as follows. The bias estimate of the context is initially set to 0. Let us assume the current context has been called N times during the coding of the image. Each time we encounter this context the accumulated prediction error B (after adaptive prediction) is compared to N. If $B < -N$ we set $\hat{R} = \hat{R} - 1$. If $B > N$ we set $\hat{R} = \hat{R} + 1$. We do not allow an absolute $\hat{R}$ value greater than a preset R_{max}. The adaptive predictor equals

$$\hat{x} = \hat{x}_{fix} + \hat{R}.$$

Quantization and Encoding of Prediction Errors. The standard LOCO algorithm has been extended to near-lossless compression [17]. There, the prediction errors are quantized with a uniform quantizer (UQ) with bin-widths $2\delta+1$, and reconstruction at the center of each bin. If δ is small, and the number of quantization bins are large, the uniform quantizer is close to the MSE distortion optimal quantizer, for an extensive family of distributions ([17], [28]). The uniform quantizer also puts a bound on the maximum pixel-wise error, i.e. δ. If δ is large, the UQ is far from optimal from MSE distortion point of view. The optimal quantizer, subject to an entropy constraint is indeed uniform for a large family of error distributions, but the reconstruction levels are not at the center of the quantization bins. In SLOCO we use a UQ-adjust quantizer for large δ. Thus, the bins near the center of the error distribution, where most of the probability mass is located, have adjusted reconstruction levels closer to the MSE distortion optimal, as well as smaller bin-widths $2\delta' + 1$, such that the maximum error for all pixels is still bounded by δ. The outer bins have bin-widths $2\delta + 1$, and center bin reconstruction levels. That is, given a prediction error $\epsilon = x - \hat{x}$, the error is quantized using the UQ-adjust quantizer. If the K center-most bins have bin-width $2\delta' + 1$ the quantized error equals

$$Q(\epsilon \mid |\epsilon| \leq (2K+1)\delta' + K)) = sign(\epsilon)\lfloor \frac{|\epsilon| + \delta'}{2\delta' + 1} \rfloor$$

and

$$Q(\epsilon \mid |\epsilon| > (2K+1)\delta' + K)) = sign(\epsilon)(K + \lfloor \frac{|\epsilon'| - 1}{2\delta + 1} \rfloor),$$

where $\epsilon' = sign(\epsilon)(\epsilon - (2K+1)\delta' + K)$.

As in LOCO, we use Rice Mapping to map the quantizer error distribution into a smaller alphabet without loss of information. The distribution of the mapped, and reduced range quantized prediction errors is quite close to a one sided geometric distribution, and encoded with a Golomb code [17]. A quantizer bin index y is thus encoded in two parts. Given a code parameter $m = 2^k$, we first encode in unary the most significant bits of y, $\lfloor \frac{y}{m} \rfloor$. The remainder $y \ mod \ m$ is encoded in binary representation. The total code length for y is thus $k + 1 + \lfloor \frac{y}{2^k} \rfloor$ bits. For optimum performance, we use many Golomb codes for the encoding of the errors. Each context class builds its own code. The Golomb parameter k that best matches the context distribution is estimated adaptively, by $k = argmin_{k'}\{2^{k'} \geq \bar{A}\}$. Here, $\bar{A}$ is the mean absolute quantized (and reduced range) prediction error of the context class.

We need to update the context parameters for optimal Golomb coding. We have already mentioned how the integer bias is updated each time a context is encountered. Other context variables we store are the accumulated prediction errors after range reduction (but before Rice mapping). We also store the accumulated absolute prediction errors, as well as a counter for each context. After a context has been encountered T times we reset all context variables to half their current value. This keeps the context information current. We use $T = 64$.

Runlength Coding. LOCO encodes the smooth regions of an image efficiently by means of a run-length code. If the quantized local gradients indicate a smooth region ($all(g1, g2, g3) \leq \delta'$) we predict the value a (i.e. the causal horizontal neighbor of x) for x. Furthermore, we predict that $l-1$ pixels following x also equal a. Instead of encoding each pixel separately, we only have to encode the deviation of the length of the "run" of values $a \pm \delta$, from the expected run-length l. The length of the observed run is encoded with a Golomb code with parameter g, where $l = 2^g$. When we encounter a value different from $a \pm \delta$ we encode the interruption pixel using similar techniques as for the encoding of regular pixel samples. We use two contexts for the interruption pixels, context 1 is used if $a = b$, and context 2 if $a \neq b$. The expected run-length l is updated based on the recently observed run-lengths. For each run equal to l, a counter index is incremented. If the increment exceeds a threshold, the expected runlength is increased by one unit. Similarly the expected runlength is decreased after a number of interruption have been encountered.

Overhead. The overhead of the Segmented LOCO (SLOCO) algorithm contains the spot means and standard deviations, as well as the local background intensity estimates and the local background standard deviation. The overhead also contains the estimated grid structure of the microarray images, and the segmentation map.

The spot and background means are encoded using adaptive Lempel-Ziv. The cost of the spot mean overhead amounts to approximately $11-15$ bits per spot on the images we have examined. The background means have a much smaller range and are easily encoded using only $3-5$ bits per spot. The spot variance is approximately proportional to the spot mean. Conditioning on the spot mean, we can encode spot and background standard deviations with $5-7$ and $4-5$ bits per spot respectively. The segmentation map is efficiently encoded using the chain code of Lu and Dunham [15]. Using a seeded region growing algorithm the shapes of the spots can be quite arbitrary, and cost between 1.2-1.6 bits times the circumference of the spot to encode. The average cost of the overhead for the images we examined is 0.376 bpp.

If no re-processing of the images is needed, the overhead contains all relevant information for down-stream analysis. In addition, it contains spot quality measurements such as spot shapes, variances and the local background variance.

Coding the Spot Regions. Given the overhead we can compute the signal-to-noise ratio of each spot. We use the following signal-to-noise ratio measure;

$$SNR = \frac{m_{spot}^2}{m_{bg}^2 + s_{bg}^2},$$

where m_{spot}, m_{bg} denote the mean spot and background intensities, and s_{bg}^2 the estimated background variance.

Based on the SNR we can pick a bound on the maximum pixel-wise error δ for each spot (see below). The size of each spot is too small to allow for any adaptive prediction step, or for adaptive estimation of the Golomb parameter. We therefore use a fixed Golomb code, and only the fixed predictor $\hat{x}_{fix}$ within each spot. The spot Golomb parameter could be estimated on the encoder side, after applying the fixed predictor, and transmitted as overhead to the decoder. However, we can do nearly as well by using an approximate estimate of the optimal Golomb parameter k obtained as follows.

The overhead contains the spot standard deviation. The expected value of the absolute prediction error is well approximated by $s_{spot}/(\sqrt{2} \times 1.3)$, where s_{spot} is the estimated spot standard deviation. The factor 1.3 has been estimated from the microarray image data. The fixed predictor is local, and gives smaller prediction errors than using the spot mean as a global predictor. Given the bound on maximum pixel error δ, we thus estimate k as

$$\hat{k} = max(0, \lceil \log_2(\lfloor \frac{\bar{A}/1.3 + \delta}{2\delta + 1} \rfloor) \rceil),$$

where $\bar{A}$ is the MAD estimate of s_{spot}, divided by $\sqrt{2}$. We encode the spots in a row scan manner. When the fixed predictor is applied, missing context pixels are filled in with the spot mean value from the overhead.

Coding the Background. The background is encoded in a row scan fashion for sub-blocks of the images. Larger blocks are more efficient to encode, but we find that encoding 4×4 blocks for the background for each print-tip configuration gives approximately the same bit-rate as encoding the background as one solid block. This allows image subset reconstruction.

To use the fixed predictor, we fill in the missing context pixels (the spot pixels) with a value equal to $3\times$ the local background intensity estimates (from the overhead). The factor 3 provides a gradient near the spots. Filling in the missing context pixels with the background intensities gives a lower bit-rate, but including a factor of 3 reduces the prediction variance.

Given the spot SNR values near a specific background region, and the local estimate of the background intensity, we pick a bound on the maximum error δ (see below). The background region of a spot consists of pixels closer to this spot than any other. The context Golomb parameter k is estimated as in standard LOCO. Because in SLOCO the maximum error bound varies from region to region we need to store the reconstructed prediction errors, not the quantization bin indices as in LOCO. The context parameter is estimated as $\hat{k} = max(0, \lceil \log_2(Au(Q)/N(Q)) \rceil)$, where $Au(Q) = \lfloor A(Q)/(2\delta'' + 1) \rfloor$, $\delta'' = \delta$ if $|\hat{\epsilon}| > K$, and δ' otherwise. $A(Q)$ is the context variable containing the accumulated absolute prediction errors, after reconstruction. If the context of pixel x indicates a smooth region we apply the following runlength coding strategy.

The runlength coding of Segmented LOCO differs from LOCO. We do not allow runs to cross from a region with higher maximum error bound δ, into one with smaller δ. If a spot is encountered during a run, we skip ahead to the next background pixel. However, we do not necessarily continue the run. Let us denote the current pixel by x. If x is on the boundary of a spot we skip ahead to the next background pixel in the same row as x, and denote this pixel by y. If $\delta(x) \leq \delta(y)$ we compute the vector of local gradients of x and y; $\tilde{g}(x)$ and $\tilde{g}(y)$. If $max|\tilde{g}(x) - \tilde{g}(y)| \leq \delta(y)$ we continue the run. Note that in this case the gradients are computed with fill-in context values equal to the local background estimates, *not* with a factor 3. Runs interrupted by a decrease in δ, or by $max|\tilde{g}(x) - \tilde{g}(y)| > \delta(y)$ are encoded as "expected interruptions", by appending a 1 to the bit-stream. Other interruptions are encoded in the same manner as in standard LOCO. The criteria for continuing or interrupting a run are available at the decoder from causal information, or from the overhead.

Maximum Error Bounds. We choose different δ for the background regions because we recognize the need for higher precision near low intensity spots. We use thresholds on the SNR values to pick the maximum error bounds. For the spot regions we use $\delta = \{511, 255, 127, 63\}$. The $\log_2 SNR$ thresholds corresponding to each δ level are chosen as the quantiles (90,50,10%) of the SNR distribution of the array images.

The near lossless LOCO design scheme assumes that δ is small, and that the number of quantization bins are large, such that the quantization errors are approximately uniformly distributed within each quantization bin. For the 16 bpp images, using a large $\delta = 255$ still corresponds to a considerable number of quantization bins. However, the error distribution is very "peaked", in the innermost quantization bins. Depending on the δ used, as discussed above we let the $K(\delta)$ innermost quantization bins have width $2\delta' + 1$ where $\delta' < \delta$. The corresponding reconstruction levels are adjusted such that the maximum error is still bounded by δ. We pick K and δ' such that the increase in the number of quantization bins is small compared to the UQ quantizer used by standard LOCO. For $\delta = \{511, 255, 127\}$, we use $\delta' = \{444, 224, 115\}$, and $K(\delta) = \{6, 10, 20\}$. For smaller δ, the UQ is used. With this setup the additional number of quantizer bins for the 16 bpp images is between 3 and 5.

Flowchart of Segmented LOCO.

Coding the spot regions.
0 Initialize: Given the Spot SNR, set the maximum error bound to equal $\delta(SNR)$. Estimate the Golomb parameter k by

$$\hat{k} = max(0, \lceil \log_2(\lfloor \frac{\bar{A}/1.3 + \delta}{2\delta + 1} \rfloor) \rceil),$$

where $\bar{A}$ is the mean absolute value of (spot pixels - spot mean) $\simeq s_{spot}/\sqrt{2}$.
1 For current pixel x, apply the fixed predictor, replacing missing pixel values with the spot mean. Compute the prediction error ϵ.
2 Quantize ϵ using the UQ-adjust quantizer. Reduce the range of the quantized residuals and apply the Rice mapping $\rightarrow \hat{\epsilon}$.
3 Encode $\hat{\epsilon}$ with Golomb($\hat{k}$).
Coding the Background.
0 Initialize: Set context variables $R, B, N = 0$ and $A = 4$. Using the overhead, pick $\delta(x)$ and $\delta'(x)$ for all background pixels x. R is the integer bias, B the accumulated prediction error, A the accumulated absolute prediction error, and N the context counter. Pick the UQ-adjust parameter K.

1 For current pixel x, fill in any non-background pixels with 3$\times$ local spot background estimate. Compute gradients $g1, g2, g3$ and quantize each to 16 levels. Denote the context index by Q.
2 if $all(g1, g2, g3) \leq \delta'(x)$ go to runlength coding mode, o/w go to 3.
3 Apply the fixed and adaptive predictor to x. Compute the prediction residual $\epsilon = x - \hat{x}_{fix} - \hat{R}(Q)$.
4 Quantize ϵ using the UQ-adjust quantizer. Reduce the range of the quantized residuals and apply the Rice mapping $\rightarrow\ \hat{\epsilon}$.
5 Estimate the context Golomb parameter k by $\hat{k} = max(0, \lceil \log_2(Au(Q)/N(Q)) \rceil)$, where $Au(Q) = \lfloor A(Q)/(2\delta'' + 1)$, $\delta'' = \delta$ if $|\hat{\epsilon}| > K$, and δ' otherwise.
6 Encode $\hat{\epsilon}$ with Golomb($\hat{k}$).
7 Update all context variables. $B(Q) < -B(Q) + \rho(\hat{\epsilon})$, $A(Q) < -A(Q) + |\rho(\hat{\epsilon})|$, where ρ is the reconstruction levels of the UQ-adjust quantizer.
Runlength Coding.
0 Initialize: Set runcount $ct = 0$. Set the runerror variable $\delta_r = \delta(x)$. Set the runvalue $r = a$, where a is the left neighbor of pixel x.
1 While $|x - r| \leq \delta$ and the class index of x is "background", add 1 to the runcount ct and go to the next pixel.
2 If $ct = l$, where l is the expected runlength, add a '1' to the output bit-stream and reset ct to 0.
3 If the class index of x is "ROI" skip ahead to the next background pixel y. If $\delta(y) \geq \delta_r$ and the gradients of y are within $\delta(y)$ of the gradients of x, continue the run. Otherwise interrupt the run by appending a '1' to the bit-stream.
4 If encounter "end-of-line", interrupt the run and append a '1' to the bit-stream
5 If $|x - r| > \delta(x)$, interrupt the run, append a '0' to the bit-stream, and encode the length of the interrupted run ($< l$).
6 If the run was interrupted in 5, encode the run-interrupt sample x with the two special contexts depending on whether $a = b$ or $a \neq b$, where b is the top neighbor of pixel x.
7 Update the expected runlength parameter l based on the number of runs encoded. Go back to standard coding of the next background pixel.

To summarize the section, SLOCO (segmented LOCO) differs from standard LOCO in mainly these four aspects: (i) the spots and backgrounds are encoded separately to avoid bias by smearing, (ii) a UQ-adjust quantizer is used instead of the UQ quantizer, (iii) we allow for different maximum error bounds δ in the background regions, and (iv) we use a runlength code that takes the segmentation map, and the varying δ into account.

The varying δ presents us with some difficulties. Ideally, we should use different contexts for each δ. However, we cannot hope to gain enough information to "train" the context based predictors if we use separate contexts. We limit ourselves to use one set of context variables. Therefore, we need to store the quantizer reconstructions, rather than the bin indices as in standard LOCO.

Lossless Compression and Progressive Transmission.
As we will see in the next section, lossy reconstruction at bit-rate ~ 4 bpp may be sufficient for genetic information extraction with a variety of existing methods for information extraction. However, methods are still under development, and there

is no guarantee that using 4 bpp image reconstructions will suffice when applying new methods. Therefore our compression scheme can be extended to a fully lossless reconstruction of the microarray images. Given the initial lossy reconstruction, the image reconstruction can be refined, spot by spot, background region by background region, or even pixel by pixel, to any bit-rate above the minimum decodable bit-rate. Our scheme is thus partially progressive.

The overhead provides us with the maximum error δ in each region of the lossy reconstruction of the images. We use a simple, and flexible way of encoding the residual image. After prediction and quantization, the quantization errors are close to uniformly distributed in the range $[-\delta, \delta]$. We cannot reduce the first order entropy ($\sim \log_2(2\delta + 1)$) much via predictive coding. We therefore encode bit planes of the residual image. We choose this simple coding scheme since it gives us total freedom to encode any part of the residual image at any rate we desire, independently of what we choose to do in other regions of the image. Despite this apparently inefficient code, we get comparable lossless compression results using the segmented LOCO and bit plane coding, as we get with standard LOCO. However, we stress that Standard LOCO is *not* progressive so to obtain that bit-rate the whole image has to be encoded and decoded in full. We improve considerably over the SPIHT algorithm lossless bit-rate with our scheme.

IV. Results and Comparison of Methods

A. Image Data sets

Our image compression scheme has been tried out on microarray images from three sources. Here, we will discuss the results obtained from a replicate experiment, courtesy of Matthew Callows, Genome Sciences, Lawrence Berkeley National Lab. The replicate experiment allows us to more convincingly evaluate the effects of compression on the statistical analysis of extracted genetic information. A discussion on the statistical analysis of this data set can be found in [9]. Results on images from P. Brown's lab, Department of Biochemistry, Stanford university, and the Lawrence Livermore National Lab are not shown here in order to conserve space. However, bitrate results were similar to those obtained on the image data set from the replicate experiment.

The replicate data set consists of 16 pairs of images; 8 pairs from 8 "control" mice, and 8 pairs from a "treatment" group. The experiment is referred to as Apo AI, and is described in greater detail in [8]. Apolipoprotein AI is related to lipid metabolism in mice. The treatment group has a knocked-out AI gene. The reference sample used for both the control and the treatment group is a pooled sample from all 8 control mice. Each image pair provides relative measurements of the expression levels of 6384 probes, 200 of which are expected to be related to lipid metabolism. With replicate experiments, we can evaluate the compression using a number of test statistics. We compare the estimated gene expression levels (after normalization) $\tilde{M}$ using the full image data, and lossy reconstructions of the images. We also look at the extracted product intensities A.

To conserve space, we present results using two different methods. We compare seeded region growing and Gaussian mixture segmentation, to adaptive circle segmentation. We also compare the robust filtering background correction scheme [29], to the valley-between-peaks method [10]. We find that the choice of background correction scheme has the largest impact on performance. Similar conclusions were drawn in [29]. Here, we therefore compare only two combinations; seeded region growing + robust filtering (hereafter referred to as Method 1), and adaptive circle segmentation + valley-between-peaks (Method 2). We can think of the methods as two extremes. Method 1 has low variance but is possibly biased, whereas Method 2 is variable and susceptible to noise in the data.

B. Bit-rate results

Method	**Bit-rate (cmp 32 bpp)**	**Compression ratio**
LZ (gzip)	21.62	1.48:1
SPIHT	19.44	1.65:1
WT+ZT+EC	18.64	1.72:1
LOCO	17.28	1.85:1
Segmented LOCO	17.45	1.83:1

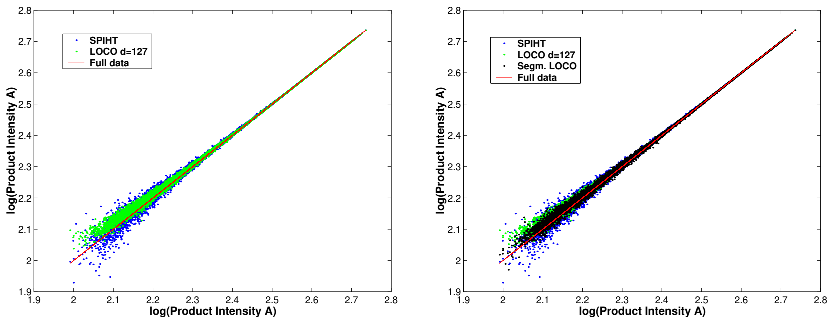
TABLE I
Lossless bit-rates.

For lossless compression we compare the results using our segmented LOCO + bit plane coding scheme, to standard LOCO, and to the SPIHT algorithm. As a baseline for lossy + bit plane coding, we look at a wavelet lossy scheme with zero-tree coding + entropy coding of the residual image (WT+ZT+EC). This method is widely recognized as one of the best in both lossy and lossless compression of natural and medical images. The lossy rate of (WT+ZT+EC) used is the same as for segmented LOCO. The reported lossless bit-rate for this last scheme is optimistic, i.e. based on an entropy calculation of the (WT+ZT) residual image. In table IV-B, bit-rates and compression ratios are shown. The results using LZ (Lempel-Ziv) and SPIHT are dismal. The compression ratios are 1.48:1 and 1.65:1 respectively. Note that in medical imaging applications, compression ratios with LZ is commonly $\sim$ 2:1, and 3:1 with SPIHT. The (WT+ZT+EC) has lossless compression ratio 1.72:1. The LOCO lossless compression ratio is 1.85:1. We get very close to this result with our scheme, 1.83:1. This bit-rate result includes the overhead ($\sim$ 0.376 bpp). We can achieve a better compression ratio than LOCO (1.87:1) if we use "fill-in" context pixels equal to the background estimates, instead of 3 times the values used here. However, this introduces too much variance in the background region of the initial lossy reconstruction. Since we want the lossy reconstructions to act as good substitutes for the full data for the purpose of re-processing, we sacrifice a lower lossless bit-rate for good progressive performance. We cannot hope to achieve much better lossless compression ratios than this with any method. The 8 least significant bits are close to random for microarray images, i.e. have marginal entropy 8 bpp, and are unpredictable. This puts a ceiling of 2:1 on the lossless compression ratio.

C. Lossy compression.

We compare SPIHT, LOCO and segmented LOCO in terms of how well the genetic information is preserved in the *lossy* image reconstructions. We compare the extracted information from *lossy* microarray images at a certain bit-rate, to the extracted information from the full (lossless) image data. The results obtained with the (WT+ZT+EC) compression scheme are very poor, and this method is dropped from the comparison. The wavelet transform has a wide support, such that the spots are blended with each other.

Figure 3 (a) shows the spot product intensity A (on a log-scale) obtained from the SPIHT reconstruction at 4.1 bpp, and the LOCO reconstruction at 4.4 bpp. The LOCO algorithm requires a fixed δ input, and 4.4 bpp was the closest available bit-rate to 4.1. The solid red line corresponds to the full data, the blue markers to SPIHT, and the green markers to LOCO. Note that the variance of A is high using the SPIHT algorithm, whereas LOCO shows significant positive bias in A for low A values as a result of the smearing of coding spots and backgrounds in one segment. In figure 3 (b) the same results are shown with segmented LOCO (black markers) at bit-rate 4.1 bpp superimposed. Note that segmented LOCO controls both the bias and variance better than LOCO and SPIHT respectively.

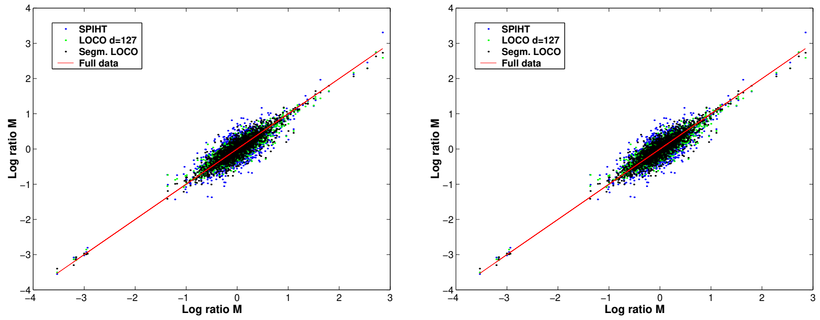


(a) (b)

Fig. 3. (a) SPIHT (blue) at rate 4.1 bpp and LOCO (green) at rate 4.4 bpp (b) Variance and Bias both under control with Segmented LOCO (black) at 4.1 bpp

Figure 4 shows the normalized differential gene expression levels $\tilde{M}$ obtained using SPIHT, LOCO and segmented LOCO at the above bit-rates. We again see that the wavelet based SPIHT algorithm performs very poorly. The segmented LOCO scheme fairs slightly better than standard LOCO. Note that all schemes are able to preserve the large absolute $\tilde{M}$ values with great accuracy. Only for small $\tilde{M}$ values does the compression seem to have an impact.

Our scheme is progressive. We can opt to add more precision to the regions of the images corresponding to small $\tilde{M}$, where the effect of compression is more noticeable. However, we find that the variability introduced by the lossy compression, in the extraction of the differential gene expression levels, is smaller than the array-to-array variability. We can thus hypothesize that the information "lost" due to compression is below the level of the noise in the data. We also find that the difference in extracted gene expression levels between Method 1 and Method 2 is much greater than the difference between lossless and lossy reconstructions of the images. In figure 5 the baseline for comparison is the gene expression levels extracted from Array 2 using Method 1. We compare the results we get using Method 2 on the same array, and Method 1 on Array 3 (blue and green markers) will the full data. We also compare the results we get using Method 1 on a 4.1 bpp lossy reconstruction of Array 2 (red markers). We compute the SD from the 8 replicated arrays of the



(a) (b)

Fig. 4. (a) Differential gene expression $\tilde{M}$ for control group (b) treatment group. SPIHT and Segmented LOCO at 4.1 bpp (blue, black) and LOCO (green) at 4.4 bpp.

same type as Array 2. The array-to-array variability is much greater than the

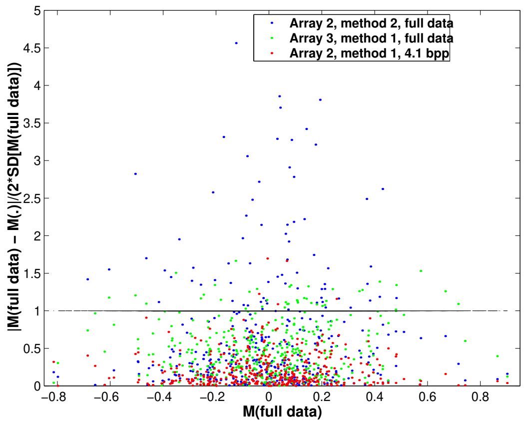
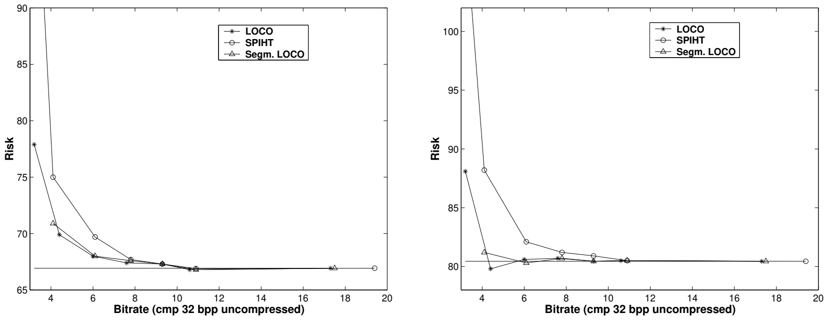


Fig. 5. 'Standardized plot', i.e. z-statistic/2. The marker display differences in M from array 2 using i) method 2 on full data, ii) method 1 on array 3, iii) method 1 on 4.1 bpp reconstruction of array 2.

lossy-lossless difference. The method-to-method difference is far greater than the array-to-array variability, indicating that great care has to be taken in which method is used for information extraction. Since the SD bands are hard to distinguish in a plot, we show a standardized plot of z-statistics (divided by 2) in figure 5. We obtain similar results for all other arrays, and when the roles of Method 1 and Method 2 are reversed.

D. Risk, Denoising and Shrinkage

The replicated experiment allows us to construct a "ground truth" for the gene expressions. We compute the mean gene expression vector from the 8 replicates using Method 1, and denote this by M^0. We construct a measure of risk for the compressed data at various bit-rates as follows. Let $\widehat{M}^j$ the be vector of extracted gene expressions from a lossy reconstruction or array j. If there are p genes on each



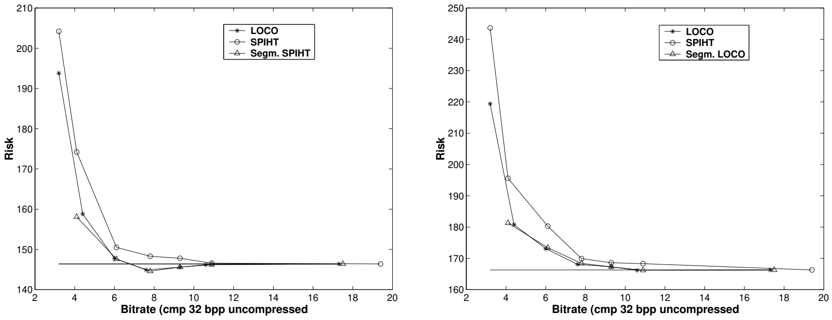
(a) (b)

Fig. 6. Method 1: (a) Risk (summed over all genes on array) vs bit-rate. Control group. (b) Treatment group.

array the risk R is computed as

$$R = \sum_{k=1}^{p} \frac{1}{8} |\widehat{M}_k^j - M_k^0|.$$

We use the L_1 norm to avoid having the large gene expressions dominate the calculated risk.



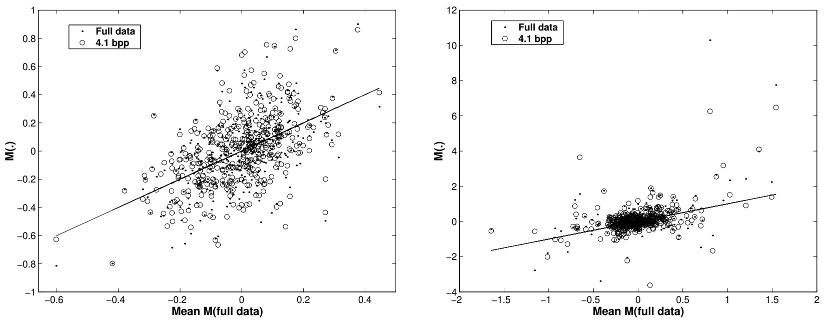
(a) (b)

Fig. 7. Method 2: (a) Risk (summed over all genes on array) vs bit-rate. Control group. (b) Treatment group.

In figure 6 (a) and (b) the risk, using Method 1, is plotted as a function of bit-rate, using the SPIHT, LOCO and segmented LOCO algorithms. The minimum decodable bit-rate for the segmented LOCO is 4.1 bpp. The standard LOCO results are computed using LOCO at each bit-rate separately, since LOCO is not a progressive scheme. SPIHT performs poorly at all bit-rates, compared to the other algorithms. The segmented LOCO and standard LOCO show comparable risk results. However, there is a caveat in this comparison. We know that standard LOCO leads to a considerable bias in the estimate of A for low bit-rates, which figure 6 does not show. The overall risk for the control group is smaller than the treatment groups (figure 6 (a) vs (b)), since the (absolute) gene expressions levels extracted from the treatment arrays are overall larger. Both standard LOCO and segmented

LOCO improves on the risk of the full data for bit-rates $\sim$ 6-8 bpp for the treatment arrays (b). We can think of the compression at these bit-rates as *denoising* of the microarray images. For both groups of arrays, we see only a marginal increase in risk for rates > 4 bpp.

In figure 7 (a) and (b) we plot the risk using Method 2. The overall risk is much higher using Method 2 than Method 1, since Method 2 introduces a lot of variability in the extracted information. We see again a denoising effect for part of the bit-rate range in 7 (a). For the entire bit-rate range displayed both standard LOCO and segmented LOCO show only marginal increase in risk over the risk obtained using the full data.



(a) (b)

Fig. 8. M(Array 2, full data) vs Mean M, (.), and M(Array 2, 4.1 bpp) vs Mean M , (o). Shrinkage toward Mean M for large absolute M values. (a) Method 1, (b) Method 2.

The equivalence of compression and denoising has been discussed in the literature previously (e.g. [?]). If the noise level is lower than the "signal", i.e. the extracted gene expression, we can get a better estimate of the gene expressions using compressed data, than we get using the full data. This is what figures 6 and 7 illustrate. Moreover, the effect is more apparent when the method of extraction is susceptible to noise (Method 2). In figure 8 the shrinkage, or denoising, effect of the compression on the extracted gene expressions are shown. The solid line is the mean gene expression vector M^0. The black dots correspond to the estimate of the gene expressions using a lossless reconstruction of Array 2. The black rings are obtained from a lossy reconstruction. We see that for large absolute values of M^0, the lossy reconstruction $\widehat{M}$ is closer to M^0, than the lossless reconstruction M. For the small and highly variable M^0, compression, or denoising does neither good, nor harm. If there is no significant signal present, we have no chance of retrieving it, no matter which method we use.

V. Discussion

We have presented a lossless and progressive compression scheme for microarray images. At bit-rate 4.1 bpp, we find that the tasks of image segmentation, and genetic information extraction with a variety of methods, are only marginally affected by the compression. The effect is smaller than the array-to-array variability. The effect of compression is also smaller than the difference between alternative methods for information extraction.

Since our scheme is partially progressive, experimentalists can opt to refine the precision of the images. The flexible structure of our scheme allows for lossless, or refined precision reconstruction for any subset of the images. Our scheme is robust to specific microarray image characteristics, and has been tried on images from three different labs.

We find that compression can in fact improve the estimation of gene expression levels. Using replicated arrays we show that compression acts as a form of shrinkage for large absolute gene expression, toward the mean over the replicated arrays.

For simplicity we have opted to code the two pairs of microarray images separately. However, a joint coding scheme may not only improve the lossless bit-rate, but the lossy performance as well. Near low intensity spots, lossy reconstruction of the background region can have huge impact on the extracted information. We control this by using higher precision in these regions. A joint coding scheme can control the variance introduced by the compression by ensuring that the *joint* loss is smaller than a specific value. The worst case scenario in separate coding is that maximum errors of opposite sign are obtained in the two scans. This can be avoided with a joint coding scheme.

Even though we believe the multi-layer data structure should work for emerging array technologies such as the protein array, modifications specific to the make-up of a particular array and the application of the array in clinical diagnostic situations might be required to overcome challenges not seen in the cDNA microarray case. For example, from articles in [2], it is not hard to imagine that the antibody array technology would be one day applied in physician's offices and an on-line version of the multi-layer data structure would be needed and statistical analysis or expression level extraction should be based on compressed objects to speed up the computation or waiting time of the patient.

Array technology is only one of the many imaging or recording technologies which provide challenges and new opportunities for the compression or signal processing community. More and more, we will be seeing the need to integrate compression and other signal processing tasks with data or statistical analysis to follow.

References

[1] Adams, R. and Bischof, L. Seeded region growing. (1994). *IEEE Trans Pattern Analysis and Machine Intelligence*, **16**, 641-647.

[2] *Supplement to BioTechniques: Hight-Throughput Proteomics: Protein Arrays I* Borrebaeck, C. A. K and Wingren, C. (eds), December, 2002.

[3] `http://www.affymetrics.com`

[4] Brown, P. `http://www-genome.stanford.edu`

[5] Chang, G., Yu, B. and Vetterli, M. (2000). Adaptive Wavelet Thresholding for Image Denoising and Compression. *IEEE trans of Image Processing,* **9**, 1532-1546.

[6] Chen, Y., Dougherty, E.R., Bittner, M.L. (1997) Ratio-based decisions and the quantitative analysis of cdna microarrays. *J Biomed Optics*, 2:364-374.

[7] Eisen, M. Scanalyze (1999) `http://rana.stanford.edu/software`

[8] S. Dudoit, Y. H. Yang, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T. P. Speed (2002). Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Research*, Vol. 30, No. 4, e15

[9] Dudoit, S., Yang, Y.H., Callow, M.J. and Speed, T.P. Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statistica Sinica*, Vol. 12, No. 1, p. 111-139.

[10] Axon Instruments, GenePix 4000A, User's Guide 1999.

[11] Hansen, M., Yu, B. Wavelet Thresholding via MDL for Natural Images. *IEEE trans Inform. Theory*, special issue on information theoretic imaging, vol 46:1778-1788, 2000.

[12] `http://www.rii.com`

[13] Jornsten, R. Data compression and its statistiscal implications: with an application to the analysis of microarray images. PhD thesis, Department of Statistics, UC Berkeley, 2001.

[14] Jornsten, R., Wang, W., Yu, B., Ramachandran, K. Microarray image compression: SLOCO and the effect of information loss *EURASIP signal processing journal*, vol 83/4, p 859-869, 2003.

[15] Lu, C. and Dunham, J. (1997) Highly Efficient Coding Schemes for Contour Lines Based on Chain Code Representation, *IEEE trans Comm*, Vol 39(10) 1511-1514.

[16] Nakachi, T. (1999). Pel Adaptive Predictive Coding Based on Image Segmentation for Lossless Compression. *IEICE trans fundamentals*, **E82(6)**, 1037-1045.

[17] Nerhav, M., Seroussi, G., Weinberger, M. (1996) Modeling and low-complexity adaptive coding for image prediction residuals. *Int'l Conference on Image Processing*, Lausanne.

[18] Newton, M.A. et al. On differential variability of expression ratios. Technical report 139, Department of biostatistics and medical informatics, UW Madison 1999.

[19] NHGRI report 1998; `http://www.nhgri.nih.gov`

[20] Nister, D. and Christopoulos, C. (1999). Lossless Region of Interest Coding. *Signal Processing*, **78**, 1-17.

[21] Ortega, A. and Ramchandran, K. (1998). Image and Video Compression, *IEEE Signal Processing Magazine*.

[22] Perlmutter, SM., Cosman, PC., Tseng, CW., Olshen, RA. and others. (1998). Medical image compression and vector quantization. *STATISTICAL SCIENCE*, **V13 N1**,30-53.

[23] Perlmutter, S., Cosman, P., Tseng, C.W., Olshen, R.A., Gray, R.M., King, C.P., Bergin, C. Medical Image compression and vector quantization, *Statistical Science*, 1998, 1:30-53.

[24] Said, A. and Perlman, W.(1993). Reversible image compression via multiresolution representation and predictive coding, *Proc SPIE*, **2094**, 664-674.

[25] Sapir, M., Churchill, G.A. Estimating the posterior probabilities of differential gene expressions from microarray data., Poster Jackson lab, 2000, `http://www.jax.org/research/churchill/research/index.html`

[26] Shen, L. and Rangayyan, R. (1997). A Segmentation-Based Lossless Image Coding Method for gene High-Resolution Medical Image Compression, *IEEE Trans. Image Proc.*, **16(3)**, 301-307.

[27] Ström, J. and Cosman, P. (1997). Medical Image Compression with Lossless Regions of Interest, *Signal Processing*, **59(2)**, 155-171.

[28] Sullivan, G.S. Efficient scalar quantization of the exponential and laplacian random variables. *IEEE trans inform theory*, vol 42:1365-1374, 1996

[29] Yang, Y., Buckley M.J, Dudoit, S. and Speed, T. (2002) Comparisons of methods for image analysis on cDNA microarray data. *J. Computat. Graph. Stat.* 11