Best Analogs for Replacing Missing Image Data

Chen Ning
Faculty of Information & Control Engineering, Shenyang Jianzhu University,
Shenyang, China, 110168
e-mail: n_chen@126.com

Clifford A. Reiter
Department of Mathematics, Lafayette College, Easton, PA 18042 USA
e-mail: reiterc@lafayette.edu

Abstract

Identifying the historical data that is the best analog with a pattern from which a forecast is sought allows time series data to be extrapolated. That technique of best analogs is most effective when the data contains underlying deterministic chaos. Here we apply similar techniques, modified to use two space dimensions instead of one time dimension, to fill-in and extrapolate missing image data. The technique is successful at replacing significant amounts of missing data with reasonable data derived from the image itself.

Keywords: Image Processing, Method of Analogs, Fractal Forecasting, Missing Data, Image Completion

1. Introduction

Casdagli explicitly describes [1] a method for extrapolating time series data by identifying the best analogs to the current situation as compared with historical data patterns regardless of when that historical data pattern occurred. This method of best analogs has also been called fractal forecasting [2-4]. A similar idea is implicit in the earlier work of Lorenz on approximating growth of errors in weather patterns [5]. In the context of time series, the main idea is to assemble a table of windows of a fixed time segment length that record the historical patterns that appeared in the time series, and then to extrapolate a current time series pattern by utilizing the historical windows most like the current pattern (that is, use the best analogs). The technique's effectiveness is comparable to other extrapolation techniques used in classical statistics [2], but it seems best for situations where there is deterministic chaos in the time series data. It can be quite accurate for a dozen or so time steps and remains realistic much longer for time series such as the Henon map [3-4].

Sprott [6] showed how to use a probabilistic voter model to create images with an appearance similar to a plasma cloud. That probabilistic automaton replaces pixels with neighbors, so it is by design coherent both with the original boundary of missing region and in the interior of the region. Moreover, his automaton runs quickly. However, the local patterns produced do not reflect the local patterns in the image itself and the illustrations primarily were images with simple palettes and fractal like regions for each color. Iterated function systems use fractal methods to represent image data [7,8] and these are known to be able to produce fake details derived from the image itself. However, the emphasis for application of iterated function systems appears to be compression rather than replacement of missing data.

We develop a method using best analogs for the purpose of providing values for data missing from 24-bit images. Ideally such an algorithm could create a realistic (but nonexistent) image that contains any particular given partial image. For example, it is common for digital photographers to expend considerable energy to crop or remove undesirable elements from images. When power lines or people are removed from an image, it may require considerable effort and time to clone regions to replace the removed data in such a way that the boundary between real and cloned data is not apparent. Algorithms that automate or mitigate that work would be welcome. The technique we describe seems remarkably effective in that way.

The basic idea of using best analogs on images is simple: create a table of patterns of data in the image; for missing pixels on boundaries, find patterns in the table that are the best match to the pattern around the missing pixel (the best analog), and then replace the missing pixel with data suggested by the best analog. In practice there are many details to consider and important ways to greatly reduce the computational effort required.

2. Using Analogs to Forecast Time Series

Before fully describing the algorithm for using best analogs for replacing missing image pixels, we will describe in more detail the algorithm for time series. Suppose we have historical time series data T_k for $1 \le k \le n$ where we might typically expect thousands of data points. We can select some window size, w, typically a relatively small number, say between 2 and 20. Then we use data windows to create tables of historical patterns and the historical result of those patterns. For example, we might take all the sets of three successive points in the series as our windows and use the first two points in each window as part of a history table of local patterns and the third point is used in a result list that could be used for the prediction. That is, we could create a matrix with rows of historical data, $h_i = \{T_i, T_{i+1}\}$, and a list of results, $r_i = T_{i+2}$. Given a data pattern $u = \{U_1, U_2\}$ with unknown following term that we want to forecast, we identify several of the historical times when the data in h_i best fits *u*. Then that historical data, along with the corresponding results, r_i, are used in a least squares linear model to predict the term that follows *u*. Of course, the historical windows can be of various widths, the predicted point need not be the next point in the sequence, the number of best fits used in the linear model can be varied and the process can be repeated.

Figure 1 shows the result of this type of forecasting taken from [4]. The first 900 terms from iteration of the (chaotic) Henon map is used to create 2-step historical data with the results being the next element of the sequence. A pair of terms from the Henon series is used as an initial pattern for which a forecast is sought (however, that pair does not appear in the historical data set -- it appears 50 terms beyond the historical data). For each forecast, the best five pairs from the historical data are used and the process is repeated using the prediction as the new last element of the data pair. Figure 1 shows the Henon sequence and the fractal forecast for 20 terms. The actual Henon sequence is shown in green while the predictions are shown in black. Notice the match is visually very good for around a dozen steps, and then prediction wanders from the actual pattern, but the prediction remains realistic as a portion of the time series data even when the match is inaccurate.

3. The Basic Algorithm for Best Analogs on Images

We now suppose we have an image with missing data that has been somehow marked. For example, we will usually mark missing data with a magenta color since that is an unusual color in nature. Figure 2 shows two images blended into a panorama using Autostitch [9]. That application automatically blends the images in a way that attempts to remove distortion caused by the different angles used with the camera in the individual images. The result produces a region outside of the image data, which we have slightly enlarged (to remove edge artifacts) and marked in magenta. Such panoramas can be trimmed by selecting the largest rectangular window so that the boundary is not visible in the trimmed version. However, such trimming loses a significant portion of the original image data. In Figure 2 the sense of the largeness of the rock and some of the puffy white clouds would be lost with such trimming. Our goal is to fill the magenta region, which amounts to slightly more than one-eighth of the pixels, with realistic data in an automated way. If successful, the resulting image would be good enough to use, or at least to allow a larger trimmed region to be used. Figure 3 shows an example of the result produced by the standard algorithm that we will describe in Section 4. In this section, we focus on a basic version of the algorithm.

We consider all the patches in the image conforming to various patterns. Figure 4 shows two pattern templates. On the left is a 3 by 3 patch that uses the eight pixels marked with an asterisk in order to estimate the location marked with a question mark. We denote this pattern *ne*, for "northeast", the direction of the pixel to be forecast. Of course there are analogous patterns given by 3 rotations: *se*, *sw*, *nw*. The second pattern in Figure 4 is a portion of a 3 by 4 patch where eight pixels are used to predict the two pixels at the positions marked with the question marks. This pattern we denote *N*, for "North". There are analogous *E*, *S* and *W* patterns. For each of the patterns we look at all the patches in the image where the marked positions are non-magenta pixels and store them in suitable pattern history (the asterisk positions) and result (the question mark positions) tables.

Before describing how we organize the replacement of the missing pixels, we describe how to forecast a single missing pixel using the basic algorithm. Suppose we are using the forecasting algorithm where a *ne* pattern is matched; thus we assume that history and result tables for *ne* patterns have been assembled. We search through the *ne* history table for the single historical pattern that is the best fit to the pattern of asterisk marked positions under consideration. Rather than using several best patches and a least squares fit, we select just one, in order to avoid the graying expected from averaging several results. The result of the forecast is the entry in the result list corresponding to that best fit of the pattern to the historical table data. That result pixel is then used to replace the magenta pixel in the *ne* corner of the patch.

Create *ne*, *se*, *sw*, *nw* history/result tables

Create *N*, *E*, *S*, and *W* history/result tables

Repeat until there are no changes to the image during a (A)-(H) loop
(A) *nw* pass:
(i) identify all image patches that match the pattern of known and unknown pixels required for a *nw* forecast
(ii) attempt to forecast *nw* each of the patches identified in (i); update pixels when successful
(B) *ne* pass: repeat (A) for *ne* patterns
(C) *se* pass: repeat (A) for *se* patterns
(D) *sw* pass: repeat (A) for *sw* patterns
(E) *S* pass: repeat (A) for *S* patterns
(F) *E* pass: repeat (A) for *E* patterns
(G) *N* pass: repeat (A) for *N* patterns
(H) *W* pass: repeat (A) for *W* patterns
End repeat

Table I. Overview of the basic forecasting algorithm.

Now we are ready to give an overview of how we organize the basic forecasting algorithm. The outline is summarized in Table I. The general algorithm first creates the history/result tables for the various patterns we will use. We repeat trying all the patterns until the image remains unchanged. We look for all possible opportunities to apply one technique, say *nw*, then apply the technique to those patches and then try another pattern. A typical sequence might be to apply *nw* wherever possible (but not pursuing the new opportunities for *nw* that are generated by new pixels), then applying *ne*, *se*, *sw*, *S*, *E*, *N*, and *W*. and then repeating that process until the image can not be updated.

Notice that the eight patterns we have described would not allow all patterns of missing pixels to be fixed. For example, an alternating checkerboard arrangement of known and unknown pixels could not be forecast using our patterns. However, in practice, for the types of missing data we used, these patterns are adequate. There are several further practical details to change before we describe the standard forecasting algorithm that we use, but running the algorithm that we have described so far results in an image very much like Figure 3.

4. The Standard Algorithm

While the basic algorithm described in the previous section is successful, we will describe in this section, our "standard algorithm", one that includes modifications to the basic algorithm to improve efficiency and the quality. Notice that the number of magenta pixels is likely to be substantial in the sense that we will usually be interested in using forecasting when 10% or more of the pixels are missing. If only a few scattered pixels were missing, replacing them with neighbors would quickly yield a reasonable image. Thus, we may think of the number of pixels that need to be forecast as being proportional to the number of pixels in the image. Also, the number of patches used to build each of

the history/result tables is also proportional to the number of pixels in the image. By the basic algorithm we have described, each pixel forecast would require comparing each surrounding patch to all the entries in the appropriate history table in order to find the best matching historical pattern. Thus, the number of comparisons required by the basic algorithm is expected to be proportional to the square of the number of pixels in the original image. Moreover, each pattern contains 8 pixels and hence 24 values when each of the RGB values are all considered. Thus, making a comparison of two patterns is significantly more arithmetic than comparing two numbers. Running the basic algorithm on the 217 by 500 image in Figure 2 takes a couple of hours on a microcomputer. That is not an implausible amount of time, but we will describe a variation that allows the time to be reduced to a few minutes.

In this section we will describe our "standard algorithm". Many variations from this standard will be obvious and we will consider some of them. The standard algorithm differs from the basic algorithm in three ways: (i) a paletted image is used to greatly reduce the number of full color comparisons used, (ii) weights are used to bias the imporatance of pixel position, and (iii) the order in which various patterns are applied is modified to produce a rough, rapidly changing edge.

We begin by creating a 256-color paletted version of the image using quantization tools in the Image3 Addon for the programming language J [10-11]. Along with the full color history/result tables, we create analogous tables for the quantized image. In Figure 5 we see the positions shown in Figure 4 with the asterisks replaced by weights. Each pattern has two pixel positions marked with weight 4. These are the pixel positions closest to the result positions (the question marks). Our plan is to only compare patches if the corresponding paletted image patches have the same entry at one or both of the two positions marked with a 4. This would offer little (but some) benefit if these comparisons had to be done each time a patch was considered. However, the histories of the 24-bit images can be pre-organized according to the palettes at those positions. Thus, when we have a patch under consideration, the palette entry at the position of a 4 essentially gives a pointer to all the full color image patches that we need to compare with our given patch. There is some bookkeeping to do (pixels in the quantized image need to be updated), but this variation makes the time required for forecasting very reasonable. It is also true that organizing the history tables can be done efficiently. In fairness, we note that this approach has very substantial memory requirements. The history tables are several times larger than the image itself, and we are using several history tables.

A practical consideration arises: it may be that there are no patterns in the history table with a pixel coming from the required palette. That is, the list of patterns in the history table that should be checked can be empty. In that case, our strategy is straightforward: do nothing. In our experience, the missing pixel will be forecast later using one of the other patterns.

The second feature that is part of our standard algorithm is that we weight pixels in the patches, somewhat favoring pixels nearer the result positions as more important than those further away. More specifically, if $p_1 =< r_1, g_1, b_1 >$ and $p_2 =< r_2, g_2, b_2 >$ are RGB triples, then we take the square of the distance between them to be $dis2(p_1, p_2) = (r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2$.

If the pixels in two patches are given (in some fixed order) by $P = \{p_1, p_2, ... p_8\}$ and $Q = \{q_1, q_2, ... q_8\}$ and the corresponding weights are $\{w_1, w_2, ... w_8\}$, then the square of the weighted distance between those patches is

$$wdis2(P,Q) = \sum_{i=1}^{8} w_i \, dis2(p_i, q_i)$$

We select the result from the patch where that weighted distance (squared) is smallest. The standard weights we use are those shown in Figure 5.

Lastly, we comment on the strategy for controlling the sequence of patterns to be used. Doing a *ne* pass involves identifying all the patches that have the proper sequence of known and unknown pixels so that they might be updated using a *ne* pattern. We do not update that list as forecasts are made, but instead move on to using other patterns. We only give up on an image if we have applied our entire sequence of patterns and made no changes to the image. Some general remarks are in order. Each pattern has its own advantages and disadvantages. The patterns that move *N* or in other directions parallel to edges sometime introduce obvious linear artifacts such as vertical streaks in clouds. However, there is overhead in running a pattern pass and these horizontal/vertical patterns can move entire edges of the image forward in one pass. Thus, they tend to be fast when they are satisfactory. The *ne* and other diagonal patterns tend to do the best job of avoiding artifacts, but can not move a rectangular edge outward, and tends to take (slow) advantage of the stair-step edges, such as those in Figure 2. Our standard algorithm is to mix these by running three rounds of the 4 diagonal passes, followed by applying a round of the horizontal/vertical passes restricted to only updating a random third of the pixels that could be updated. Thus, steps (A)-(D) from Table I are repeated three times, and then only a third of the possible patches from (E)-(H) are used in each pattern pass in that sequence. This tends to lead to lots of diagonal processing, but with steady outward motion also occurring. Of course, using best analogs forecasting is sensitive to these details and for any given image one might well be more pleased with a slightly different choice for ordering the passes, using different weights, or selecting a different version of the paletted image, but our experience suggests that this standard choice is reasonable. Figure 3 shows the result of the standard forecasting algorithm applied to Figure 2. A movie consisting of 31 frames, one at the end of each (A)-(H) loop, is available at [12]. Some examples illustrating the sensitivity to the small changes mentioned above may also be found there.

5. Further Experiments and Some Variants

As a next experiment, consider the left half of the original image that appeared in Figure 2 where a magenta eraser has been used to remove evidence of humans. That is, the hiker (Steve), his shadow, and the trail blaze have been marked out. Figure 6 shows the input image and the resulting standard forecast. The forecast might be acceptable as it is, but it could be improved by erasing any offending smaller region and applying the algorithm again. In any case, we view this as a reasonable image of "wilderness" and success for the algorithm.

Figure 7 shows the original image of the right half of Figure 2 with 10 by 10 blocks of pixels randomly removed so that 25% of the original image has been removed. The bottom half of the image shows the recovery by the standard forecast. While artifacts

appear, they are not apparent under casual observation and we again consider the forecast successful. Figure 8 shows the magnitude of the difference of the original image with the forecast shown in Figure 7. We reversed RGB values to enhance visibility of those differences; thus, white corresponds to no difference. Notice that there is very little difference for the sky but a vague sense of the missing blocks for the foreground. Figure 9 shows the frequency distribution for the difference in magnitudes (averaged over RBG) shown in Figure 8. For example, more than one percent of the forecast pixels were perfectly correct and 12% had an average (over RGB values) difference of 1 out of 256. Higher average differences appeared, but the data shown represents 93% of the missing pixels. Notice that many forecast pixels were little different from their correct values.

Thus far, we have considered examples where we view our forecasting as having been successful. Now we turn to some other illustrations where the algorithm has only partial success in order to illustrate the limitations of the algorithm. Figure 10 begins with the same original as the previous figure, but a large 160 by 160 square has been removed from the center of the 480 by 640 image. The standard forecast may improve things, but it leaves disturbing artifacts near the center. This illustrates that the algorithm does not force interior coherence and probably is more effective at extrapolation as compared to interpolation. Still, one could again erase any offending portion and apply the algorithm again on a smaller unknown region. Figure 11 shows the magnitude of differences of the original image with the forecast. Note that the central missing square is apparent and some of its central portions are quite dark, which is indicative of the poorness of the fit. A histogram showing the distribution of the average differences appears in Figure 12. In this case 87% of the data is shown and the shift, compared to Figure 9, of the differences upward may be observed.

Figure 13 shows the image from Figure 2 with the boundary widened so that more than half the pixels are unknown. We do not consider this a test of meeting our goals, but rather a test of how far the idea might be carried. Here we use a variant on the standard algorithm where we apply two rounds of the (A)-(D) passes of the diagonal patterns (versus three rounds in the standard algorithm). Notice that the image is not outrageous. However, there are small amusing artifacts such as a piece of Steve's head floating in the brush on the left. A new trail marker and piece of his elbow appears on the rock toward the bottom right of the figure.

Figure 14 shows an alternate scheme used to extrapolate the same wide boundaries as appeared in the source in Figure 13. Here an initial threshold of 500 was set and if the weighted square of the distance between patches was not less than that threshold, the pixel was not updated. If running through the pattern sequence loops made no changes, the threshold was raised by a factor of 2 and processing continued. The pattern sequence did one loop of the horizontal/vertical loops and then 3 of the diagonal loops. Notice that using thresholds resulted in very good fits being used for almost all pixels, forcing significant regions to be cloned. The price paid for that fine detail is that the boundaries between those cloned regions are obviously unnatural. We also see evidence of the streaking in the clouds that occurs when N is a dominant pattern. Nonetheless, we consider the idea of thresholds to be very interesting.

Conclusion

Forecasting using best analogs is a simple technique for using good matches to patterns in historical data to forecast from similar data. We have seen that basic idea can be used to forecast data missing from images. Several details beyond those basic ideas can be used to make the algorithm run quickly and to improve its results. We have seen the success of the resulting algorithm for modest extrapolation and on images with scattered small regions of missing data.

Acknowledgement

This work was accomplished while Professor Chen Ning was a visiting scholar at Lafayette College. The support Natural Science Foundation of Liaoning province of China (20032005) and the Foundation of Science and Technology Bureau of Shenyang (200143-01) and the hospitality of Lafayette College are greatly appreciated.

References

[1] Casdagli M. Chaos and deterministic versus stochastic non-linear modeling. Journal of the Royal Statistical Society, Series B 1992;54: 303-28.

[2] Balkin SD. Fractal and chaotic time series analysis. Master of Arts Paper: The Pennsylvania State University; 1996.

[3] Reiter CA. Fractals, visualization and J. 2nd ed. Toronto: Jsoftware, Inc; 2000.

[4] Reiter C. With J: Fractal forecasting. APL Quote Quad 2003;34 1:15-8.

[5] Lorenz, EN. Atmospheric predictability as revealed by naturally occurring analogues Journal of the Atmospheric Sciences 1969;26:636-46.

[6] Sprott JC. A method for approximating missing data in spatial patterns. Computers & Graphics 2004;28:113-7.

[7] Barnsley, MF. Fractals everywhere. 2nd ed. Cambridge: Academic Press; 1993.

[8] Fisher Y, ed. Fractal image encoding and analysis. Berlin: Springer-Verlag; 1998.

[9] Brown M. Autostitch. http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html.

[10] Jsoftware, Inc. The J programming language. http://www.jsoftware.com.

[11] Reiter ZX, Reiter CA. The image3 addon. http://www.jsoftware.com/jwiki/Packages/Image3

[12] Chen N, Reiter, C. Auxiliary materials for best analogs for replacing missing image data. http://www.lafayette.edu/~reiterc/mvq/bafrmid/index.html.

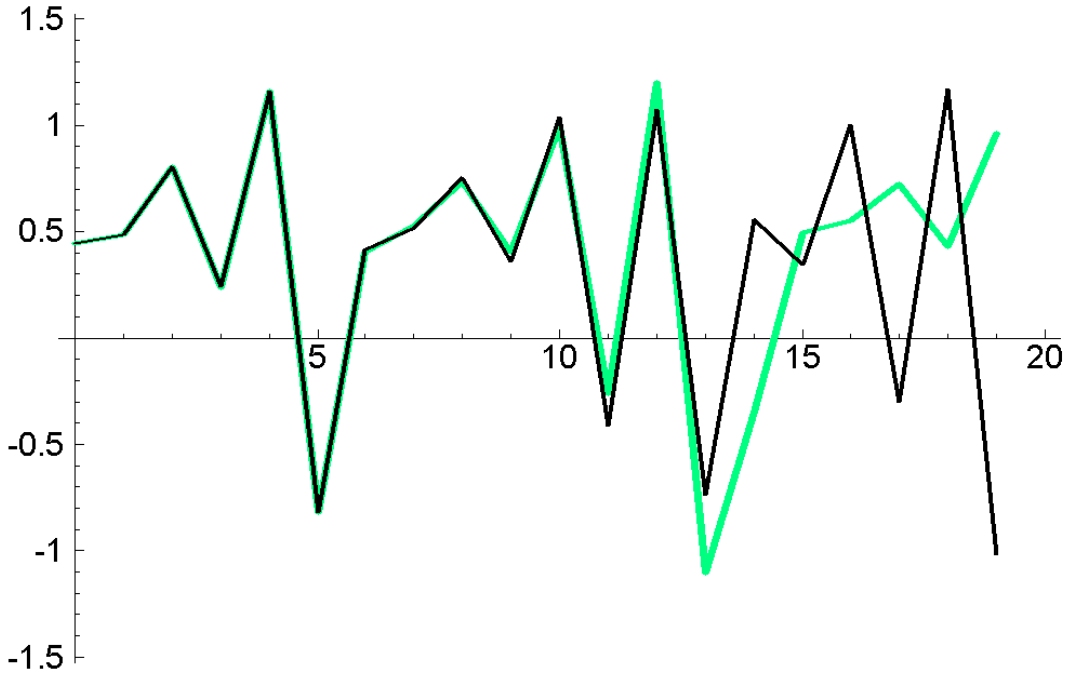

Figure 1. Time series best analogs forecast for the Henon map.

Figure 2. A panoramic image with missing pixels along the edge.

Figure 3. The standard forecast of the panoramic image from Figure 2.

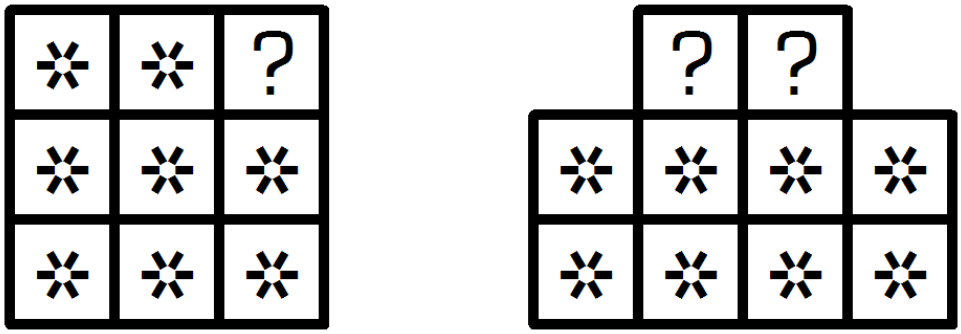

Figure 4. The northeast and north patterns used for the history tables.

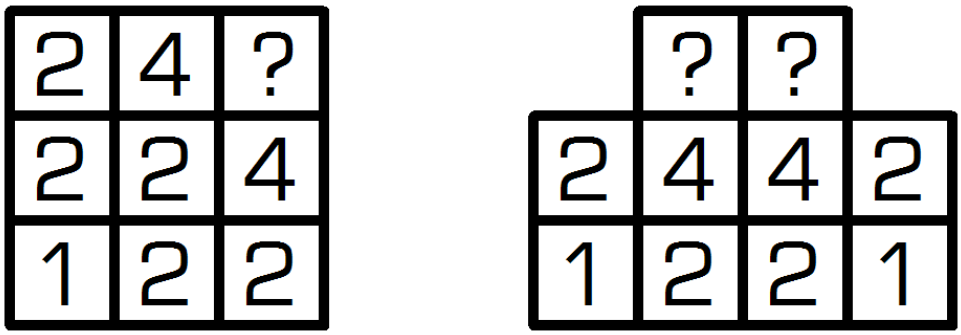

Figure 5. The northeast and north weighted patterns.

Figure 6. Removal of human artifacts via the standard forecast.

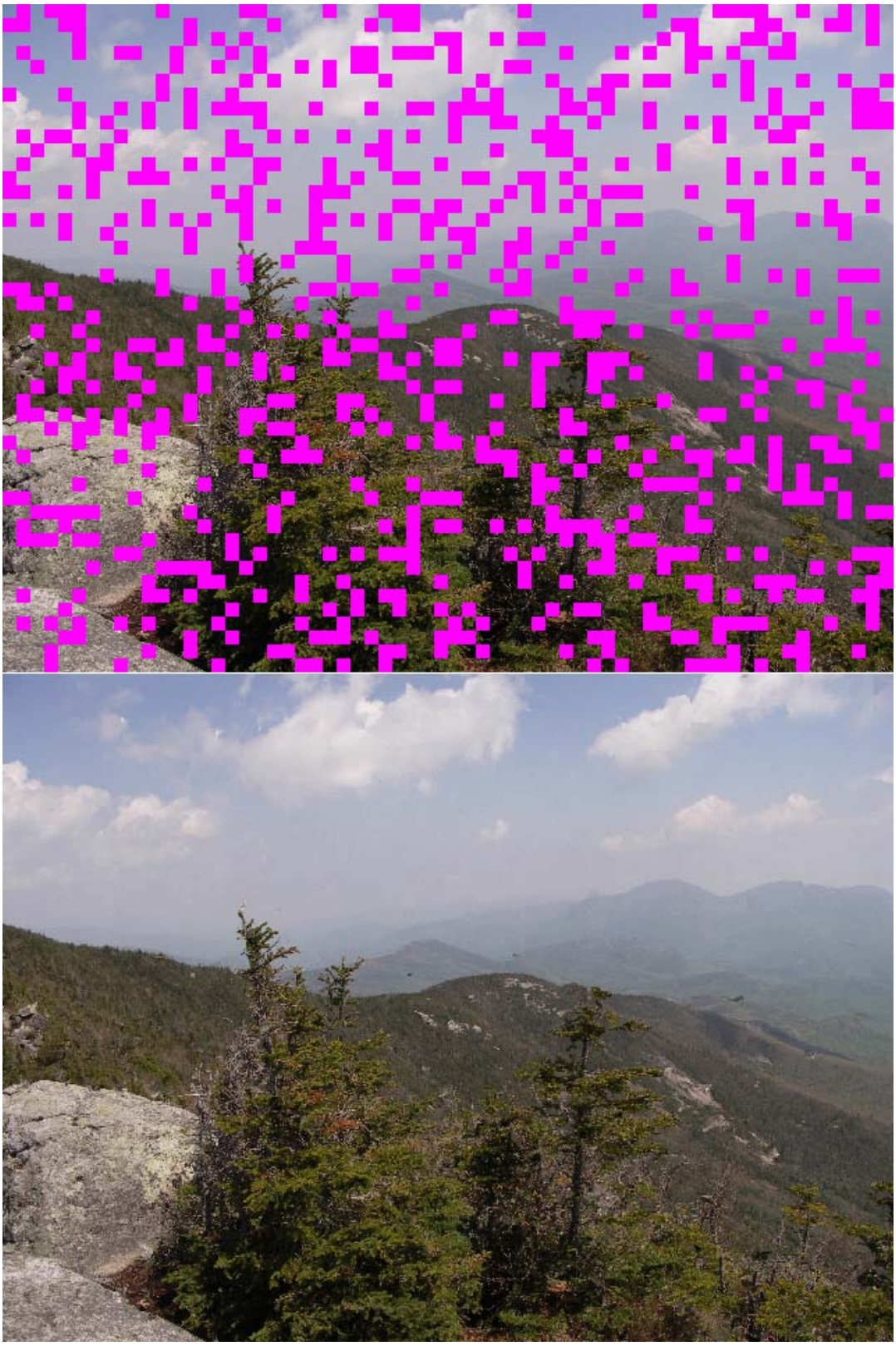

Figure 7. Recovery of 25% of an image via the standard forecast.

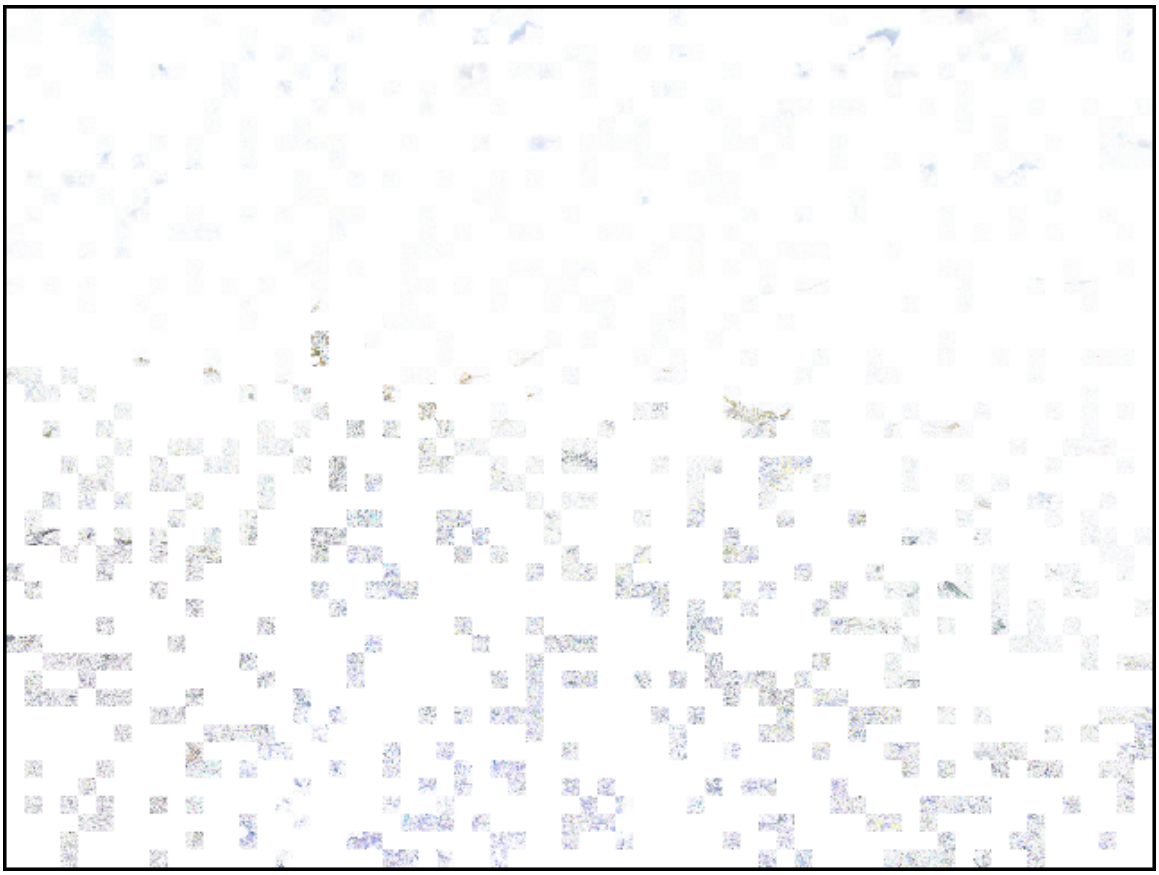

Figure 8. Magnitude of differences of the original and the forecast from Figure 7.

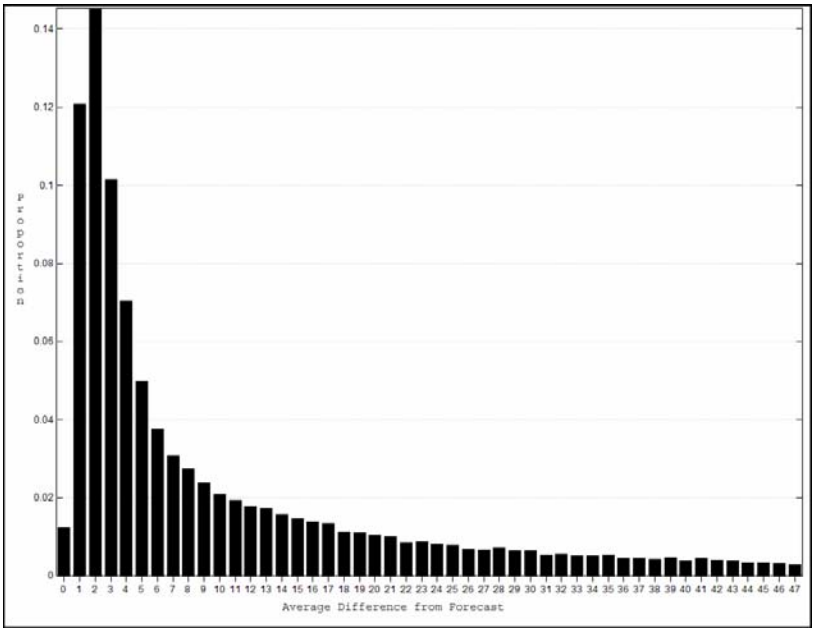

Figure 9. Distribution of average differences of the original and the forecast pixels from Figure 7.

Figure 10. The standard forecast leaves artifacts in large interior regions.

Figure 11. Magnitude of difference from original for Figure 10.

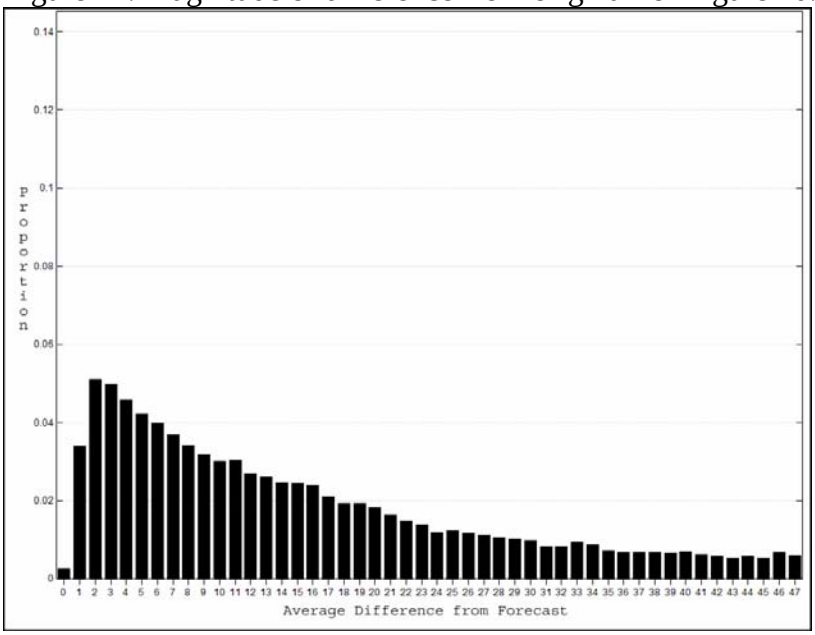

Figure 12. Distribution of average differences of the original and the forecast pixels from Figure 10.

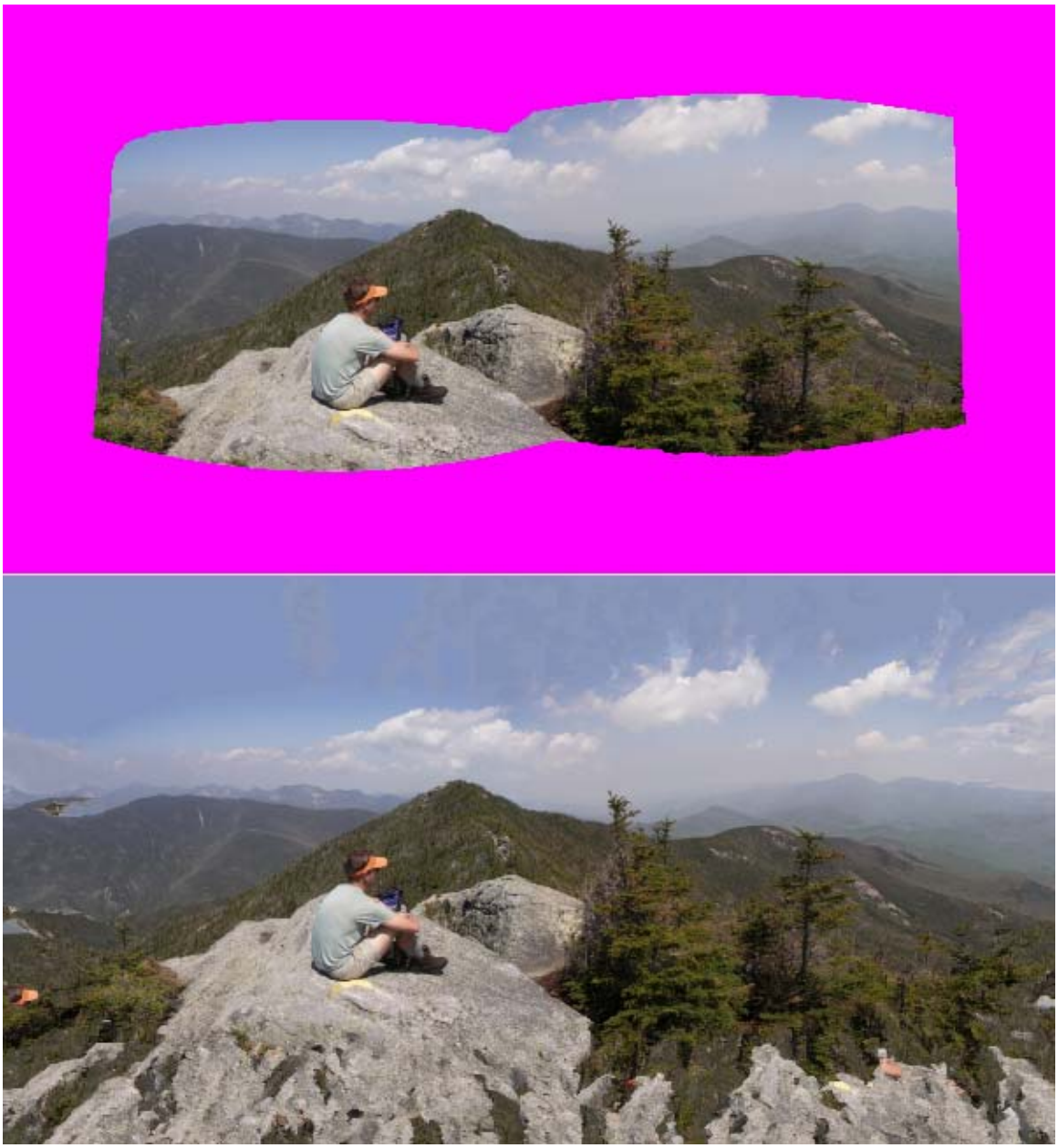

Figure 13. Forecast of more missing pixels than original pixels.

Figure 14. Using thresholds for the forecast.