

Missing Data Interpolation with Gaussian Pyramids

Satyakee Sen

ABSTRACT

I describe a technique for interpolation of missing data in which local operators of many scales but identical shape serve as basis functions. A data structure known as the Gaussian pyramid is developed to represent image information at different scales. This data structure in essence consists of a series of lowpass filtered versions of the original image stacked up one on top of the other forming a pyramid like structure. I first show how to generate a set of reduced images which stack up to form the Gaussian pyramid structure and then show how we can use this Gaussian pyramid structure to fill in missing data. Several examples of filling in missing data with this algorithm are shown and in most cases the results are comparable with those estimated using a prediction filter approach.

INTRODUCTION

The importance of analyzing images at many scales arises from the nature of the images themselves. Any analysis procedure that is applied at a single scale will tend to miss information at other scales. The immediate solution to this problem would be to carry out the analysis at all possible scales simultaneously. This is especially true for seismic data where the low frequency components in an image represent the general trend of the image, while the high frequency components add details to this underlying trend. Recently methods have been suggested (Abma and Kabir, 2005) in which the missing data is reconstructed by adding in frequency components to the missing region with each iteration. While the Fourier transform is a natural way of efficiently separating the various scales in an image, but in the Fourier domain we can no longer recognize the spatial features in their usual form as in the original non-transformed domain. A desirable alternative to the Fourier domain representation is an approach that describes an image at multiple spatial resolutions and also preserves the local spatial structure of the image at each of these multiple scales.

The simplest way to detect a pattern that may appear in an image at any scale is by simple convolution of the target pattern, constructed at various scales, with the image or to convolve a pattern of a fixed size with different versions of the image represented at different resolutions. The immediate bottleneck to this convolution-based method for detecting a pattern is the enormous cost involved in carrying out all the required convolutions. The computer graphics industry has developed a method termed as the image pyramid data structure for efficient scaled convolutions through reduced image representation (Burt, 1981). This pyramid data structure consists of a stack of copies of the initial image such that both spatial density and

resolution decrease as we move from one level of the stack to the next. This data structure can be generated with a highly efficient iterative algorithm that requires fewer computational steps to generate a series of reduced images than are required by the FFT method to compute a single filtered image (Burt, 1981). Once a fast algorithm is available for generating multi-resolution images in the spatial domain, missing regions of the image can then be filled up also at different scales, starting from the coarsest scale and proceeding to more and more finer scales. In this paper I show how interpolation of seismic data can be carried out using the pyramid structure. The local n point operator that is used as the basis function in the pyramid generation process represents a Gaussian distribution in the limit $n \rightarrow \infty$, hence the pyramid structure is termed Gaussian pyramid. I first show how the pyramid structure is generated and then show how interpolation is carried out at different levels of the pyramid to restore missing data.

GAUSSIAN PYRAMID GENERATION

The Gaussian pyramid generation is done by starting with an initial image \mathbf{g}_0 and then lowpass filtering this image to obtain a "reduced" image \mathbf{g}_1 . The image is "reduced" in the sense that both spatial density and resolution are decreased. This operation can be continued to obtain a set of images $\{\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_n\}$ that form the pyramid image structure. The low pass filtering is done by a procedure equivalent to convolution by a set of local symmetric weighting functions (for example a Gaussian distribution). A fast algorithm for generating such a Gaussian pyramid structure is outlined in the next subsection. Along with the Gaussian pyramid generation, there is also an approximate reverse process of the pyramid generation that tries to expand a reduced image to its original scale. The same algorithm that is used to form the pyramid can also be used to perform the expansion operation as outlined in the following subsection.

Gaussian pyramid generation

Suppose we start of with an initial image having N columns and M rows. This image forms the base or the zeroth level of the pyramid. Each point in the next level is computed as a weighted average of values in level 0 within a 5-by-5 window, termed as the weighting function. The size of the weighting function is not critical in the pyramid generation process (Burt, 1981).

The pyramid generation process can now be represented as :

$$\mathbf{g}_l(i, j) = \sum_{n=-2}^2 \sum_{m=-2}^2 \mathbf{w}(m, n) \mathbf{g}_{l-1}(2i + m, 2j + n). \quad (1)$$

where $w(m, n)$ is the weighting function. This same weighting function is used to generate the pyramid at each level. Notice that for each dimension the density of nodes is reduced by half from one level to the next.

The weighting function is chosen subject to certain constraints (Burt, 1981). For simplicity

it is made separable:

$$\mathbf{w}(m,n) = \mathbf{w}(m)\mathbf{w}(n). \quad (2)$$

It is normalized to 1 and also made symmetric:

$$\mathbf{w}(i) = \mathbf{w}(-i), \quad i = 0, 1, 2. \quad (3)$$

It is also stipulated that each node contributes equally to nodes at the higher level so that a typical 5 point weighting function looks like:

$$\begin{aligned} \mathbf{w}(0) &= a, \\ \mathbf{w}(-1) &= \mathbf{w}(1) = 0.25, \\ \mathbf{w}(-2) &= \mathbf{w}(2) = 0.25 - 0.5a. \end{aligned} \quad (4)$$

where a is a free parameter that controls the shape of the weighting function. For $a = 0.4$, the weighting function represents a Gaussian distribution in the limit.

Pyramid expansion

An approximate reverse of the process of generating a pyramid structure is termed as pyramid expansion. The main function of the expansion algorithm is to enhance a pyramid level of size $M + 1$ by $N + 1$ to size $2M + 1$ by $2N + 1$ by interpolating new sample values between those at the current level. Thus the expansion process applied l times to an image at pyramid level j would yield an image $\mathbf{g}_{j,l}$ which is of the same size as an image at the pyramid level $j - l$ that is \mathbf{g}_{j-l} . Formally the expansion operation can be defined as:

$$\mathbf{g}_{l,n}(i, j) = 4 \sum_{n=-2}^2 \sum_{m=-2}^2 \mathbf{w}(m,n) \mathbf{g}_{l,n-1} \left(\frac{i-m}{2}, \frac{j-n}{2} \right). \quad (5)$$

Notice that during the sum only values for which the indices are integer are included as

Figure 1: Three levels of the Gaussian pyramid formation (top panel) and expansion (bottom panel). In the bottom panel each level is expanded to the next lower level. ssen1-shot2eg [ER]

Figure 2: Four levels of the Gaussian pyramid formation (top panel) and expansion (bottom panel). In the bottom panel each level is expanded to the next lower level. ssen1-seisw [ER]

Figure 3: Four levels of the Gaussian pyramid formation for data with holes. ssen1-stlinesz [ER]

Figure 4: Filling up the holes at each level of the pyramid starting from the topmost pyramid level (upper left). `ssen1-stlinesf` [ER]

contribution to the next higher level. A simple example consisting of two shot gathers is used to demonstrate the pyramid generation and expansion operation in Figure 1. The top panel of the figure shows three levels of the pyramid while the bottom panel shows the process of expansion from one level to the next. The next example (Figure 2) consists of a stacked seismic. Again the top panel of the figure shows the levels of the Gaussian pyramid while the bottom panel shows the expansion. Each of the images in the top panels of Figures 1 and 2 have been obtained by applying the same 5-by-5 weighting function. Notice that as the pyramid levels stack up the sample density along each of the two axes reduces by a factor of 2. During the expansion process (bottom panels of figures 1 and 2) on the other hand the number of sample point increase by a factor of 2 along each axis.

Mathematical basis of Pyramid Generation

This method of pyramid formation as outlined in the previous section is equivalent to convolving an image g_0 with a weighting function h_l as :

$$\mathbf{g}_l = \mathbf{h}_l \otimes \mathbf{g}_0. \quad (6)$$

Here the size of the weighting function doubles from one level to the next as does the distance between the sample points. The shape of the weighting function infact converges rapidly to a characteristic form with successive higher levels of the pyramid. By characteristic form we mean the shape of the weighting function with a particular choice of the free parameter a (for example for $a = 0.4$ it will approach a Gaussian distribution). The effect of convolving the image with one of the equivalent weighting functions, \mathbf{h}_l , is to low-pass filter the image. The pyramid algorithm mimics this low-pass filtering operation using a small compact two dimensional weighting function and uses a fast algorithm for generating different filtered versions of the original image.

Laplacian Pyramid

The Laplacian pyramid (Burt and Adelson, 1983) is a sequence of error images $\mathbf{L}_0, \mathbf{L}_1, \dots, \mathbf{L}_n$ such that each error image is the difference between two levels of the Gaussian pyramid, that is:

$$\mathbf{L}_j = \mathbf{g}_j - \mathbf{g}_{j+1,1}. \quad (7)$$

where $\mathbf{g}_{j+1,1}$ is the image at pyramid level $j + 1$ expanded to size of the image at level j . Thus it is immediately clear tht the Gaussian pyramid formation and expansion process is exact, in the sense that the original image \mathbf{g}_0 is fully recoverable, as :

$$\mathbf{g}_0 = \sum \mathbf{L}_i. \quad (8)$$

The way to do this is to first expand the top pyramid level, \mathbf{L}_n , and then add the expanded version to L_{n-1} to form \mathbf{g}_{n-1} . This process is repeated for each level until we reach the base of the pyramid where the original image is fully recovered. Since the top of the pyramid does not have an error image we can treat the image at the top of the pyramid as the error image \mathbf{L}_n :

$$\mathbf{g}_n = \mathbf{L}_n. \quad (9)$$

Notice that the value of each node of the Laplacian pyramid is the difference between the convolutions of two equivalent weighting functions h_l and h_{l+1} . This operation is similar to convolving the image with an appropriately scaled Laplacian weighting function and hence the name Laplacian pyramid. But the cost involved with this operation would be substantially more than constructing error images as a difference between two pyramid levels. The Laplacian pyramid can be treated as a set of band-pass filtered versions of the original image just as the Gaussian pyramid represents low-pass filtered versions of the original image. In the next section I use the concepts of both Gaussian pyramids and Laplacian pyramids and the fact that the pyramid forming process is exactly reversible to show how interpolation of missing data can be done using the pyramid scheme.

GAUSSIAN PYRAMID INTERPOLATION

In this section I show how the Gaussian pyramid structure can be used to perform missing data interpolation. Consider the image on the upper leftmost side of Figure 3. This image has a 30 point hole in it. Now as the Gaussian pyramid structure for this image is generated both the image dimension as well as the size of the hole shrinks. At level 3, the hole is only 1 points long. Thus each level of the Gaussian pyramid contains information about the missing hole at different scale/size. This shrinking of the holes in the pyramid generation process gives a strategy to interpolate missing data without the use of prediction filters. Instead of using prediction filters the image is considered as a sum of patterns at many scales and restoration of the missing data is carried out at many scales as well.

If the size of the missing piece is comparable to the finest scale features of an image, then a good estimate of the unknown value at $x + \delta x$ can be estimated through a simple first order linear prediction based on the first derivative of the image at the known point x as:

$$\mathbf{g}_0(x + \delta x) = \mathbf{g}_0(x) + \delta \mathbf{x} \nabla \mathbf{g}_0(\mathbf{x}). \quad (10)$$

However if the missing piece is large then more derivatives would be needed, in other words we need to examine variations at larger scales as well. An immediate analogy to this problem would be to look at the missing information itself at various scales. As the Gaussian pyramid is generated different versions of the image \mathbf{g}_0 is obtained at different resolutions. The missing data in the original image is also missing at any other level of the pyramid \mathbf{g}_i , the only difference being that the scale or size of the missing piece is reduced. Thus at a coarser spatial scale we do not need to use a polynomial of high degree to fill in the missing data. At some level of the pyramid the hole would be of a size comparable to the finest scale features in that

pyramid level. At this level we can use simple linear prediction to fill in the missing data. Once the missing piece is filled up at some pyramid level we can then expand, starting from that level, and wherever a missing piece of information exists we can simply use the value of the expanded level. The algorithm for the missing data interpolation would then be:

1. Start with the initial image \mathbf{g}_0 .
2. Form pyramid levels $\mathbf{g}_1, \mathbf{g}_2$, and so on
3. If at pyramid level \mathbf{j} , missing piece is very small fill \mathbf{g}_j using linear prediction.
4. Expand \mathbf{g}_j to $\mathbf{g}_{j,1}$.
5. Fill in missing space in \mathbf{g}_{j-1} using $\mathbf{g}_{j,1}$.
6. Expand the filled \mathbf{g}_{j-1} and continue filling and expanding.
7. Finally fill in original image \mathbf{g}_0 using the filled $\mathbf{g}_{1,0}$.
8. If needed iterate with the filled \mathbf{g}_0 .
9. Stop.

This algorithm is illustrated with the dataset with the hole, shown in Figure 3. First the right lowermost panel of this figure is filled up using linear prediction (shown on the upper most left panel of Figure 4). Starting from this filled level, expansion of the filled level followed by subsequent filling up is done till the original image is filled up (lower right panel of Figure 4).

EXAMPLES

In this section I demonstrate some of the results of applying the Gaussian interpolation scheme to different datasets. In each of the examples a rectangular hole is punched in dataset which is then filled up using Gaussian pyramid interpolation. I compare the results obtained using the Gaussian pyramid scheme with those obtained from filling the hole using a regular prediction filter approach. The first example is the "Wood" example from Jon Claerbot's book (Claerbot, 2005). Figure 5 shows four levels of the pyramid structure. At the fourth level the hole is only 2 point long. We use linear prediction to fill in this hole. This filled level is then expanded to the next lower level which has a missing region. This missing region is filled up using the information available from the expanded filled level and this process is repeated till we reach the base of the pyramid. The results of filling up the hole by this method is shown in Figure 6. The top most panel on the left is the upper most level of the pyramid which is first filled up using linear prediction. The lower right panel is the original image that has been filled up. For this example 2 iterations of the pyramid formation and expansion process is done. The reason for this is to improve the pixel quality in and around the hole during the expansion process. Figure 7 compares the result of the interpolation process using the pyramid scheme with that obtained using a prediction error filter (PEF) approach. The top two panels show the

original data (right) and the data with holes punched into it (top left). The lower two panels show the results using a PEF (lower left) and the pyramid scheme (lower right). The results are of comparable quality, except at the top and lower boundaries where the pyramid scheme has troubles due to edge effects during the expansion and reduction process.

The second example consists of the "Brick" dataset again from Jon Claerbout's book (Claerbout, 2005). Here three levels of the Gaussian pyramid are constructed and 2 iterations are done for the pyramid formation and expansion process. Figure 8 compares the interpolation result outlined here with the PEF based approach (lower two panels of figure 8). The interpolation result is pretty satisfactory, except for the blurring effect in the missing region. The third example consists of the "Herring bone" dataset. Figure 9 compares the interpolation result with that of the original dataset. The hole is again being filled relatively well.

Figure 5: Four levels of the Gaussian pyramid. [ssen1-woodhole](#) [ER]

Figure 6: Filled levels of the Gaussian pyramid starting from the uppermost level (top left) to the lowermost level (bottom right). [ssen1-woodfill](#) [ER]

Figure 7: Data with holes (top left), original data (top right), filled using PEF (bottom left) and filled using the pyramid scheme (bottom right). [ssen1-woodcomp](#) [ER]

The fourth example consists of a stacked seismic section. Again a hole is punched in the middle (left panel of Figure 11) which we try and restore. Three levels of the pyramid structure for this example is shown in Figure 10. After the third level the hole is four points long, which is filled up using normalized linear prediction. The expansion and filling up, starting from the third level of the pyramid is started, and the results of interpolation are shown in Figure 11. The method has trouble trying to interpolate the uncollapsed diffraction hyperbolas and other artifacts present in the stacked section.

CONCLUSION

In this paper I have outlined a method for interpolation of missing data at multiple scales. The Gaussian pyramid structure has been used to represent data at various scales or sizes. Using this data structure interpolation can be effectively carried out starting from the top level of the pyramid and then expanding downwards, until we reach the base of the pyramid where we can fill up the missing piece in the original data. Several iterations of the reduction and expansion process might be needed. The chief reason for this is to improve the pixel amplitude in the hole at the top level of the pyramid, from where we start the interpolation process. The pyramid forming process is extremely inexpensive and thus several iterations can be easily afforded. There are however some boundary issues with this method. Artifacts can be generated at the

Figure 8: Data with holes (top left), original data (top right), filled using PEF (bottom left) and filled using the pyramid scheme (bottom right). `ssen1-brickcomp` [ER]

Figure 9: Data with holes (left), original data (middle) and filled data (right). `ssen1-herrcomp` [ER]

boundary primarily because a boundary node never gets a contribution from the whole 5 point window either during reduction or expansion. Also amplitude balancing for the pixels in the missing part of the dataset at the top of the pyramid, from where the interpolation starts, is an important issue. Developing sophisticated and systematic method for this amplitude balancing has not been attempted and remains a topic for future research. The blurring effect seen in most of the results is due to this amplitude balancing issue.

Another important issue with this method, with regards to seismic data, is that it may not be possible to construct too many levels of the pyramid. In such a case however we can easily construct a small prediction filter to fill in the data at the coarsest pyramid that we can have and start filling up from there. Moreover since the hole is bound to shrink during the pyramid formation process the filter estimation may actually be more convenient. The method outlined in this paper looks at the problem of missing data interpolation at many different spatial scales, rather than trying to estimate the missing data directly from the original image. It is however different in principle from many multi-scale approaches proposed earlier. The main issue that most multi-scale methods try and address is the problem of estimating a PEF effectively. The pyramid scheme on the other hand tries to stay away from the process of PEF estimation and instead tries to fill in data at multiple spatial resolutions.

REFERENCES

- Abma, R. and N. Kabir, 2005, 3-D interpolation of irregular data with a POCS algorithm.: 75th Annual Internat. Mtg., Society Of Exploration Geophysicists, Expanded Abstracts, 2150–2173.
- Burt, P. J. and E. H. Adelson, 1983, The Laplacian Pyramid as a Compact Image Code: IEEE Transactions on Communications, **31**, 532–540.
- Burt, P. J., 1981, Fast filter transform for image processing: Computer Graphics and Image processing, **16**, 20–51.
- Claerbout, J. F., 2005, Image Estimation by Example: Class notes, <http://sepwww.stanford.edu/sep/prof/index.html>.

Figure 10: Three levels of the pyramid for the stacked seismic section. `ssen1-WGstackpyr` [ER]

Figure 11: Data with holes (left), original data (middle) and filled data (right).
`ssen1-WGstackcomp` [ER]

