geoconvention
Calgary • Canada • May 15-19 2017

Recovering data from seismic images by colourmap estimation

Matt Hall[1] *and Matteo Niccoli*[2]
[1]*Agile,* [2]*My Carta*

Summary

Many scientific images — false-colour rasters or bitmaps — are published without obvious or easy access to the data that was used to make them. We present a semi-supervised algorithm, implemented in Python, to recover data from such images without a priori knowledge of the colourmap. We use image processing to detect the data area, then quantize the colours in the input image using k-means clustering. Using a novel approach, we apply heuristics to order these colours into a colour table, with which we can look up the data values. Once we have the data, we can recolour the image using any colourmap, or load it to a seismic interpretation package.

Introduction

Scientific publications are often not accompanied by the data required to reproduce their results. The data problem seems to be especially bad in applied sciences, such as exploration geophysics, where authors may feel oppressive corporate or cultural requirements to keep data and algorithms proprietary. We highlighted this problem in a GeoConvention 'unsession' in 2014.

But there is a large amount of 'pseudo-open' data: images of data that are conceptually the same as data, but practically different because images are difficult to load into a seismic interpretation package.

We have devised a method to recover data from some scientific images, especially seismic data, with no knowledge of the colourmap and usually with no input from the user.

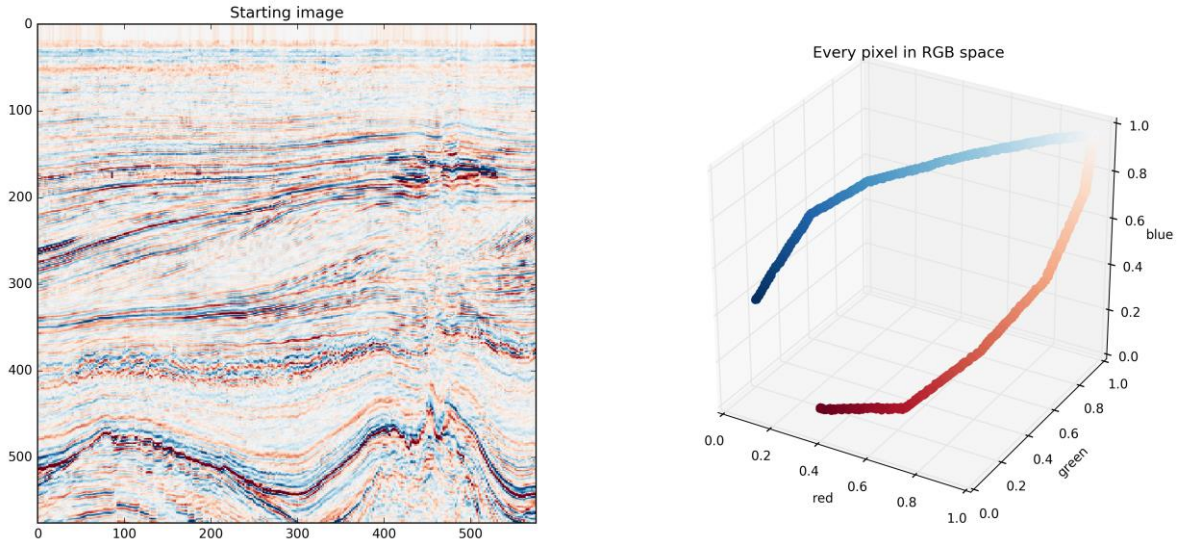

Figure 1. Left: the original image, in this case a dataset coloured with a known colourmap. Right: every pixel in the image plotted into RGB space. It seems to form a line representing the colourmap, but the points are not in order.

Plotting any false-coloured image (i.e. a dataset plotted with a colourmap) reveals the colourmap (almost always) forms a continuous line in RGB space, as shown in Figure 1. This line represents the linear sequence of colours that is the colourmap. However, the points are not in order — if we want to use the apparent line as a lookup table, we have to organize the points into sequence first. In our example, we can then map dark blue to a value of 1, light grey to 0, and dark red to -1, and everything in between.

It turns out this is not as easy as it might appear.

Method

In many cases, there is little to be done. If the colourmap is perceptually linear, essentially just a ramp from dark to light (or vice versa), then it is enough to, say, convert it to greyscale. On the other hand, if the colourmap is reproduced with the image, and is of high enough resolution, then recovering the data is also relatively straightforward. Tools like *yoink* exist to simplify the process of isolating the data and using the colourmap to recover the data from the image.

But in a large number of cases, especially for seismic data, the colourmap is not perceptual and is not present in the image. Our algorithm for handling those cases is as follows:

1. The image may contain non-data (annotations, marginalia, etc). We use scikit-image (van der Walt et al. 2014) to find and isolate the data area in the image. The steps are: threshold, filter features by size, keep the largest feature, and finally close holes.

2. We use unsupervised K-means clustering in scikit-learn (Pedregosa et al. 2011) to quantize the image to some reduced number of colours, e.g. 128. This provides the colours in the colourmap.

3. Using an original method, we heuristically find the start of the colourmap, then step over the colour quanta by solving a modified traveling salesman problem using the LKH solver (Helsgaun 2009). This sorts the colours and, together with the distances between quanta, results in the estimated colourmap.

4. To transform colours into scalar values, we look up the image pixels against a k-d tree of the quanta. This results in a 2D array of scalars representing the seismic data.

5. Finally, we save a SEG Y file of the amplitude data, and optionally recolour the image using a perceptual colourmap.

Depending on the size and quality of the original image, this process often works with default settings. But some judgment helps with the colour reduction in step 2, and with the locus crawling in step 3.

Some things are not easy to cope with, however:

- The magnitude of the samples in the original image is impossible to guess. In seismic data at least, it is not usually important.

- The polarity of the data is impossible to guess. Our heuristic for the 'most positive' end of the colourmap is: 'the point with fewest neighbours that is closest to black'. In some cases, this will result in the polarity being reversed.

- Data from very small images, or images that have been subjected to lossy compression (almost all JPEG images), can usually not be recovered. This affects many scientific images unfortunately.

- Annotations inside the image area, if they are in a colour that appears in the colourmap, cause problems. If they are in a different colour, they can potentially be handled.

Example

To continue the example we started with, Figure 2 shows a simple example from the F3 dataset, offshore Netherlands (dGB and TNO, 1987). We made the data image from the seismic data, so we know the underlying data and the colourmap exactly. The middle plot shows the recovered colourmap plotted into RGB space. The uneven spacing is usually a result of the roughly Laplacian (or double exponential) distribution of amplitudes, but it's not important because the colour quanta were fit to the data — so these colours are the 'best' colours to represent the actual data distribution.

The result is qualitatively very good, certainly interpretable. The RMS error is 0.06 and seems to be a global scaling error, rather than a local misfit error. That said, unless we can improve the error, it seems wise treat the amplitudes as relative and probably not quantitatively reliable.

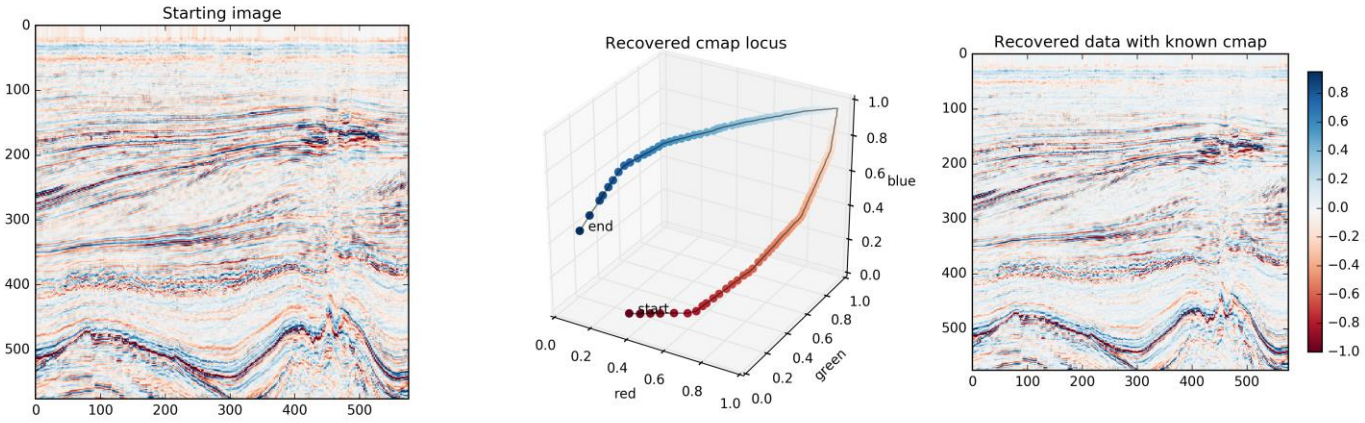

Figure 2. Left: the original image. Middle: the recovered colourmap plotted as a locus in RGB space (the black point is at the lower left, white at the upper right; the other corners of the cube represent the primary and secondary colours). Right: The recovered data, normalized here to [-1,1], displayed with the known colourmap.

Conclusions

Image are data. With the aim of making scientific claims more reproducible, we hope this work makes it easier for scientists to recover scalar data from scientific images, even when the colourmap is unknown. We implemented an algorithm in Python to perform data recovery on images, especially of seismic data, without knowing the colourmap. With some caveats, the algorithm performs well enough in most cases to enable data to be converted to SEG Y data and interpreted in standard software packages. Our code is open source and available at http://github.com/cseg/2017-hall-niccoli.

Acknowledgments

Thanks to Diego Castañeda and Chris Cousins for fruitful discussions leading to this work. An open access version of this abstract is available at http://ageo.co/geocon2017-colour.

References

dGB and TNO (1987). F3 seismic data, offshore Netherlands. Open Seismic Repository. http://www.opendtect.org/osr/. Licensed CC-BY-SA.

Helsgaun, K (2009). General k-opt submoves for the Lin–Kernighan TSP heuristic. *Mathematical Programming Computation*, 2009, doi: 10.1007/s12532-009-0004-6.

Peddregosa, F, et al. (2011). Scikit-learn: Machine Learning in Python, JMLR **12**, pp. 2825-2830, 2011.

van der Walt, S, JL Schönberger, J Nunez-Iglesias, F Boulogne, JD Warner, N Yager, E Gouillart, T Yu and the scikit-image contributors (2014). Scikit-image: Image processing in Python. PeerJ 2:e453 doi: 10.7717/peerj.453