Visualizing (Number Theoretic) Functions with Portraits

Donald Spector

Dept. of Physics, Hobart and William Smith Colleges, Geneva, NY, USA; spector@hws.edu

Abstract

I offer a way to determine properties of mathematical functions through a new visualization method that takes advantage of human perception of images. Since the digitization of images turns images into arrays of numbers, one can apply mathematical functions to images on a pixel-by-pixel basis. Thanks to the pattern recognition skills of humans, the form of the transformed images can give ready insight into the behavior of those mathematical functions. I explain the method and give some examples here, with a particular focus on number theoretic functions.

Introduction

Developing an understanding of the behavior of mathematical functions takes time and experience, often dependent on possessing a certain degree of comfort with translating abstract results into broadly conceptualized properties. This can be particularly true of arithmetic functions in number theory, for which continuity arguments are not generally available.

Here, I offer a novel technique to understand the behavior of functions: are they monotonic? do they fluctuate? how rapidly do they grow or fluctuate? My initial motivation was to obtain an effective way to visualize certain aspects of the behavior of number theoretic functions, but the method is perfectly general. The fundamental idea is that when mathematical functions are applied in a particular way to images, visual comparison of the transformed images to the original can readily reveal aspects of the behavior of the function. In this paper, I first describe the technique and then give a variety of examples of how it works in practice.

Generating and transforming images

Consider a grayscale image. A standard way to represent such an image digitally is as a rectangular array of real numbers, each in the interval $[0, 1]$, one number assigned to each pixel. These real numbers represent how dark that pixel is; a black pixel is assigned 0, a white pixel is assigned 1, and the numbers in between represent the various shades of gray, with the smaller values darker than the larger values.

For our purposes, it is better to represent these grays using integers from 0 to N_H, rather than real numbers, binning the various shades of gray into $N_H + 1$ distinct hues through a linear scaling. Representing images this way is valuable for two reasons. First, because we will frequently be considering functions (from number theory) defined on integers, we will need to represent the images using integers. Second, being able to vary N_H will allow us to study the variation of functions in an important way. We call the image represented using real numbers I_0, and the one represented using integers I_A (which is, of course, a function of N_H).

Given an image I_A, we consider what happens when we transform it according to some function Γ applied pixel by pixel. We thus take an input image I_A with pixels A_{ij}, each of which is an integer anywhere from 0 to N_H, and convert this to an array B with pixels $B_{ij} = \Gamma(A_{ij})$. We restrict the allowed functions Γ to be non-negative. Depending on the function, there is no guarantee that the output pixels will still fall in the range from 0 to N_H, or even that they will be integer-valued in some cases. Consequently, we rescale the pixel values so the image can easily be rendered. Let m to be the maximum value of any of the B_{ij}, $m = Max_{\{i,j\}} B_{ij}$. Then construct the array C defined by $C_{ij} = \frac{1}{m} B_{ij}$. The entries C_{ij} are real numbers in the interval $[0, 1]$, and so we can render the array C as a grayscale image, which we call I_C.

We find that by a simple visual comparison of the transformed images to the original images at varying values of N_H, one can readily determine mathematical properties of the transformation function. Before demonstrating this through a number of examples in the subsequent section, we address some technical features. First, because of the final rescaling, in the output image, the brightest pixel must be fully white. There is not a similar constraint on the black side; for most images of interest, with a broad range of brightnesses, this asymmetry is not significant. However, it is worth noting that a simple identity transformation applied to I_A can yield an I_C that is (slightly) brighter than I_A, if no pixels in I_A are fully saturated white. Second, to study a function that can only be evaluated on positive inputs, we increment each pixel value to $A_{ij} + 1$ before applying the function to get the array B. Third, the method works with color images, too; one separates the color image into separate channels (e.g., using RGB or CMYK), transforms each channel individually, and then reassembles the output images using the same color model used to separate the original image.

Understanding functions by their action on images

For this paper, we will take two images, each of which we will transform in a variety of ways. Comparing different transformations on the same images will help highlight the insights our methodology yields. The images are two photographs: one of Handmann's portrait of 18th century Swiss mathematician Leonhard Euler [1], and one of 20th century American mathematician Dorothy Vaughan [2].

Figure 1: *Images of Leonhard Euler, in color and in grayscale, and of Dororthy Vaughan, in grayscale.*

To begin, let us transform both grayscale images using the Euler totient function ϕ. Given a natural number k, $\phi(k)$ is the number of positive integers less than k relatively prime to k [3]; for ease, here we set $\phi(0) = 0$. (As an example, $\phi(6) = 2$, as 1 and 5 have no factors in common with 6, while 2, 3, and 4 do.) In Figure 2, we see the totient transforms of both images, each first with $N_H = 50$, and then with $N_H = 250$. From these transformed images, we can discern aspects of the behavior of $\phi(n)$. On average, ϕ is larger for

Figure 2: *Totient transforms at $N_H = 50$ and $N_H = 250$ of the grayscale Euler and Vaughan images.*

larger n, but fluctuates quite dramatically. If ϕ were strictly monotonically increasing, the output images

would retain the relative darkness of each pixel, but here they only do so on average; regions such as Euler's cheeks or the background in Vaughan's photo, quite uniform in the original, become sets of fluctuating lighter and darker grays. Still, because of the tendency of ϕ to increase on average, these fluctuations create an effect akin to halftoning [4], which becomes better at higher N_H, with more different gray shades available to be averaged by the observer in sections where the hue does not change dramatically. The key point is this: we can infer these aspects of the totient function's behavior simply from looking at these images.

Next, we transform the images using the Möbius inversion function μ, which is defined on positive integers as follows: if n is divisible by a perfect square, $\mu(n) = 0$; if n is squarefree, then $\mu(n)$ is $+1$ or -1 if n has an even or odd number of prime factors, respectively. For simplicity, we take $\mu(0) = 0$. To get our transformed images, rather than using $\mu(n)$, we use $\mu(n)+1$, so the outputs are 0, 1, or 2. Taking the transform with the increasing number of gray hues, we get the images in Figure 3. Since the Möbius inversion function

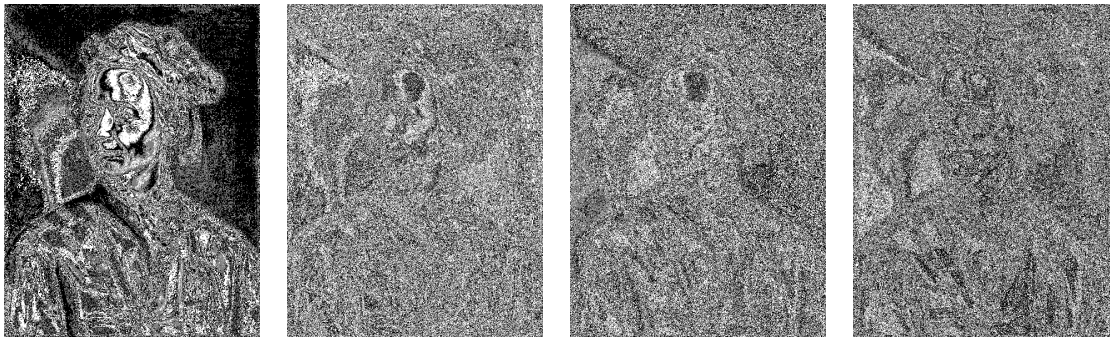

Figure 3: *Möbius inversion function transforms of the grayscale Euler image,* $N_H = 50, 300, 500,$ *&* 1050.

yields only three values, these images contain only white, middle gray, and black pixels. While at the low value of N_H, gross features of the original image are still apparent, this is primarily an artifact of the limited number of gray hues into which the original picture was divided. For the larger N_H, the resulting images seem essentially random. There are faint hints of the original image, but without the original image to compare to, they would be easy to miss. In short, these images indicate that, as the input increases, the Möbius inversion function goes among its three possible outputs in a way barely distinguishable from random variation.

We now compare two different transforms of the Vaughan image in Figure 4. In the first image, we replace each pixel value by the closest prime greater than the pixel value. In the other three, we replace each pixel value by the interval to the next prime (thus if the first prime greater than n is p, $\Gamma(n) = p - n$), at three N_H values. Replacing each pixel value by the next prime preserves the ordering of hues from dark to light,

Figure 4: *Image of Vaughan with pixel values transformed to the next prime,* $N_H = 100$*; and to the gap to the next prime,* $N_H = 20, 200,$ *and* 2000.

albeit reducing the number of grays used, so the output is much like the original image. The other images

show that the distance to the next prime fluctuates, producing a jumble of light and dark in the output. Not only does the gap to the next prime fluctuate, the average gap tends to increase as the input does; this is why as N_H increases, the output images become so much darker. Everything is normalized such that the brightest output pixel is fully white. Consequently, one large value will suppress the rest, producing a darker image. Of course, the distribution of primes has been much studied [3], but it is striking to see the behavior determined analytically to be so apparent via these image transforms.

To close, we consider three different transforms of the Euler image: one in which each pixel with value n is replaced by the n^{th} harmonic number, $\sum_{k=1}^{n} \frac{1}{n}$; one in which the transforming function is the number of partitions of n (how many ways n can be written as a sum of positive integers); and lastly where the transform is to take the square of the sine of each pixel value. The results are shown in Figure 5. In the first, we see

Figure 5: *Euler image transforms: Harmonic number and partitions at* $N_H = 100$*;* $\sin^2$ *at* $N_H = 10$ *&* 100*.*

that the image is washed out and pushed towards the white end, without losing the basic form of the picture, showing that the harmonic numbers grow monotonically, but much less than linearly (logarithmically, in fact). In the second, since the number of partitions of n grows extremely quickly, and the grays in final images are scaled relative to the largest output value, very few pixels wind up distinguishable from black. The last two images arise from computing $\sin^2(n)$, where n is the pixel value, for $N_H = 10$ and $N_H = 100$. The picture appears recognizable in the first case, due to the limited bins into which hues are placed, but at $N_H = 100$, we see an apparently random blur; as the period of $\sin^2$ is incommensurate with 1, the original gray hues are mapped to seemingly random new gray hues, leading to a result that looks like static on an old television.

Conclusion

This paper presents a novel method for gaining an insight into the behavior of a variety of functions, focusing on number theoretic functions. Having such a concise visual representation of how a function behaves is an interesting case of feedback, using images to provide insights into mathematics, but relying on having a mathematical representation of images. The examples given here are but a small sampling of what I have explored, including a range of other functions and the results of using this method on color images in both RGB and CMYK formulations. I also anticipate experimenting with using this approach pedagogically.

References

[1] E. Handmann. Portrait of Leonhard Euler. 1753. Kunstmuseum Basel. Online collection accessed 1 March 2020.

[2] Photograph of Dorothy Vaughan. NASA. Link accessed 1 March 2020: https://www.nasa.gov/langley/hall-of-honor/dorothy-j-vaughan

[3] T.M. Apostol. *Introduction to Analytic Number Theory*. Springer-Verlag, 1976.

[4] D.C. Stulik and A. Kaplan. *Halftone*. Getty Conservation Institute, 2013. Link accessed 1 March 2020: https://www.getty.edu/conservation/publications_resources/pdf_publications/pdf/atlas_halftone.pdf