Fooling around with QuantLib

Laplace Interpolation

Laplace Interpolation

November 21, 2015 | Peter Caspers | Laplace Interpolation, Missing Values | Leave a comment

Consider a (two dimensional) data matrix with some values missing and you want to fill the holes by interpolated values. One particularly simple but at the same time powerful method is called Laplace interpolation.

The easy explanation goes as follows: for each missing value take the average over the 4 surrounding values, i.e. of the entries above, below, left and right. This holds for the inner points of the matrix, special rules apply to the borders, see below. Of course, maybe some or even all of the surrounding values are missing, too. In general this leads to a system of linear equations one has to solve.

The more involved explanation: search for a *harmonic* function interpolating the non-missing data. A harmonic function $f(x, y)$ satisfies

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) f = 0$$

from which the mean value property follows. This says that each function value is the average of the function values on or in a surrounding sphere S

$$f(x, y) = \frac{1}{2\pi r} \int_{\partial S} f(x, y) ds = \frac{1}{\pi r^2} \int_S f(x, y) d(x, y)$$

where $S = \{(x', y') | (x' - x)^2 + (y' - y)^2 < r\}$. That's deeper math of course. To find the harmonic function we have to solve its defining partial differential equation, which we discretize on the grid given by the data matrix itself assuming equidistant nodes (with distance say h). For an inner point we have

$$\frac{f(x+h,y) - 2f(x,y) + f(x-h,y)}{h^2} + \frac{f(x,y+h) - 2f(x,y) + f(x,y-h)}{h^2} = 0$$

which is the same as

$$f(x, y) - \frac{f(x+h,y) + f(x-h,y) + f(x,y-h) + f(x,y+h)}{4} = 0$$

matching the easy explanation from the beginning. For each inner point of the matrix we either get this averaging equation, or if the data point is known to be $z = z(x, y)$ simply

$$f(x, y) = z(x, y)$$

This way we collect nm equations, if the original data matrix has m rows and n columns. The borders have modified equations, which I do not list here explicitly, but which are based on the conditions $f_{xx} = 0$ for the upper and lower border, $f_{yy} = 0$ for the left and right border and $f_x + f_y = 0$ for the four corners.

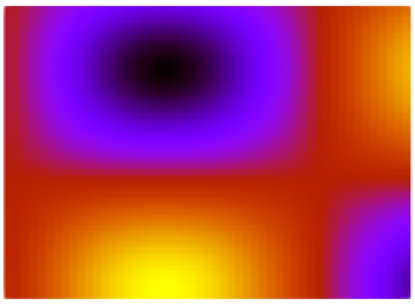
Here (https://github.com/pcaspers/quantlib/blob/master/QuantLib/ql/experimental/math/laplaceinterpolation.hpp) is a tentative implementation in QuantLib using the BiCGStab (https://github.com/pcaspers/quantlib/blob/master/QuantLib/ql/math/matrixutilities/bicgstab.hpp) solver at its heart.

Let's do some examples (the code can be found here (https://github.com/pcaspers/quantlib/blob/master/QuantLib/Examples/laplaceinterpolation.cpp), together with a gnuplot script (https://github.com/pcaspers/quantlib/blob/master/QuantLib/Examples/laplaceinterpolation.gp) generating the pictures.

The original data is generated by filling a 50×50 matrix `sample` like this

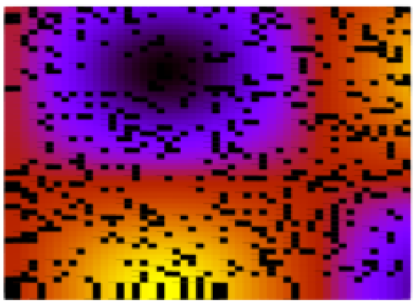
```
for (Size i = 0; i < N; ++i) {
    for (Size j = 0; j < N; ++j) {
        sample(i, j) =
            sin((double)(i + 1) / N * 4.0) * cos((double)(j + 1) / N * 4.0);
    }
}
```

By the way, the function generating the test data is *not* harmonic. Plotted as a heatmap it looks like this:



(https://quantlib.files.wordpress.com/2015/11/laplace_orig.png)

Now we delete 20% of the points, randomly.



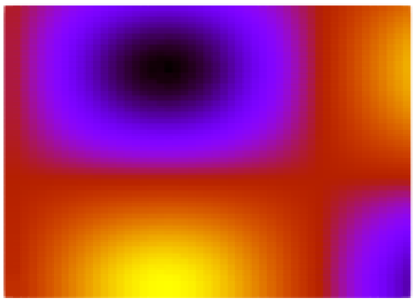
(https://quantlib.files.wordpress.com/2015/11/laplace_des_random_20.png)

We reconstruct the missing values by applying Laplace interpolation, which is done as easy as this

```
laplaceInterpolation(sample2);
```

where `sample2` is the destructed matrix with missing values marked as `Null`. The function `laplaceInterpolation` replaces these values by interpolated ones and writes them back to the input matrix `sample2`.

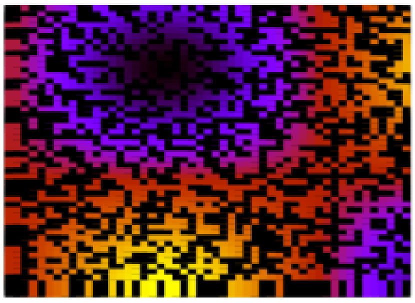
The reconstructed matrix looks is this:



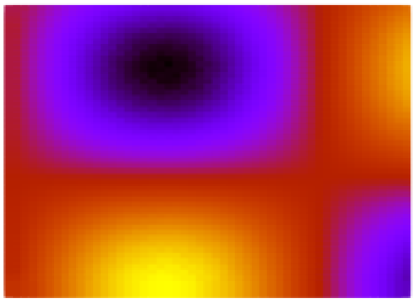
(https://quantlib.files.wordpress.com/2015/11/laplace_rec_random_20.png)

As beautiful as the original, isn't it. Let's delete more points and look and the reconstruction.

For 50% we get

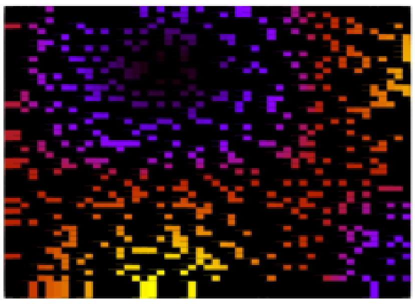


(https://quantlib.files.wordpress.com/2015/11/laplace_des_random_50.png)

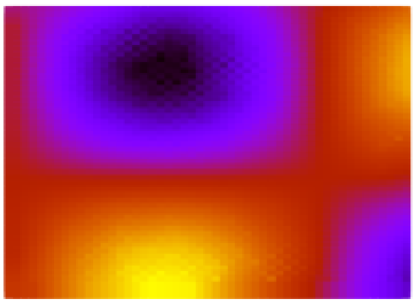


(https://quantlib.files.wordpress.com/2015/11/laplace_rec_random_50.png)

for 80% it is

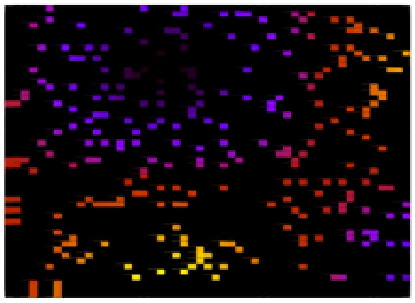


(https://quantlib.files.wordpress.com/2015/11/laplace_des_random_80.png)

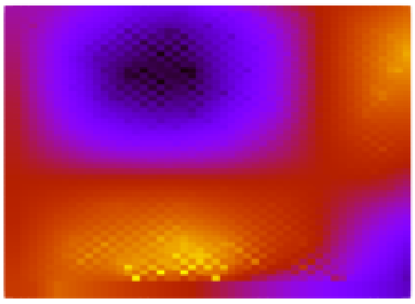


(https://quantlib.files.wordpress.com/2015/11/laplace_rec_random_80.png)

for 90%

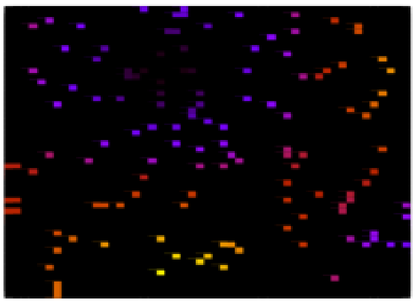


(https://quantlib.files.wordpress.com/2015/11/laplace_des_random_90.png)

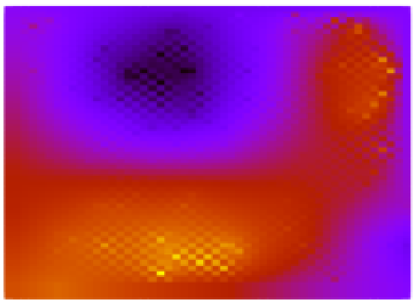


(https://quantlib.files.wordpress.com/2015/11/laplace_rec_random_90.png)

and for 95%



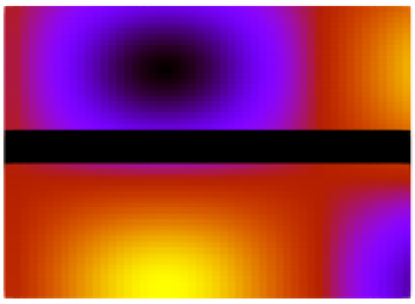
(https://quantlib.files.wordpress.com/2015/11/laplace_des_random_90.png)



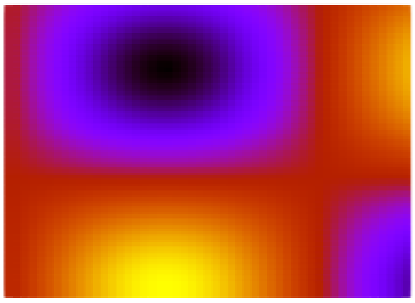
(https://quantlib.files.wordpress.com/2015/11/laplace_rec_random_90.png)

Quite amazing. Let's look at how well the method works if we do not delete points randomly, but erase a whole stripe at the center of the matrix.

If the stripe is 10% of the height

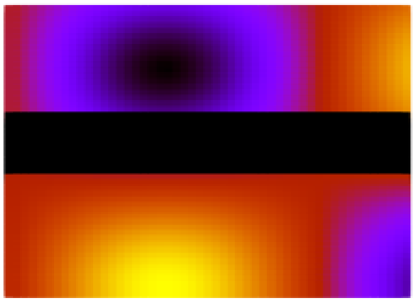


(https://quantlib.files.wordpress.com/2015/11/laplace_des_stripecenter_10.png)

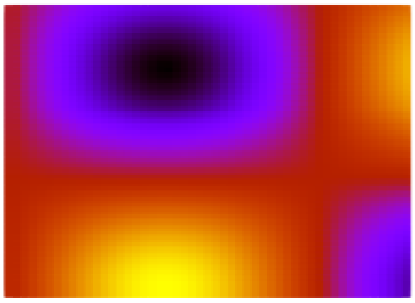


(https://quantlib.files.wordpress.com/2015/11/laplace_rec_stripecenter_10.png)

for 20%

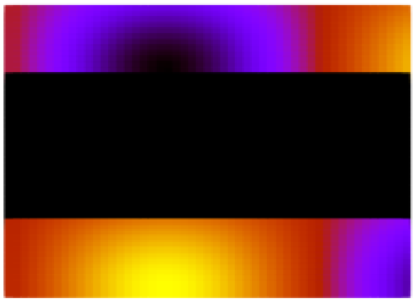


(https://quantlib.files.wordpress.com/2015/11/laplace_des_stripecenter_20.png)

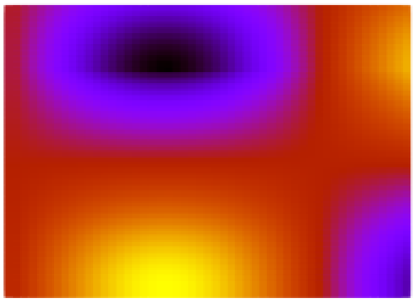


(https://quantlib.files.wordpress.com/2015/11/laplace_rec_stripecenter_20.png)

and for 50%



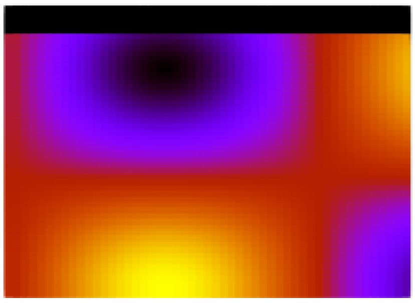
(https://quantlib.files.wordpress.com/2015/11/laplace_des_stripecenter_50.png)



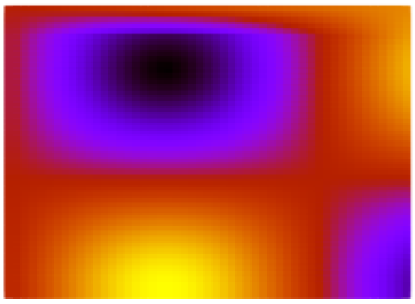
(https://quantlib.files.wordpress.com/2015/11/laplace_rec_stripecenter_50.png)

What about extrapolation? So let's delete a stripe at the top of the matrix.

For 10% we get

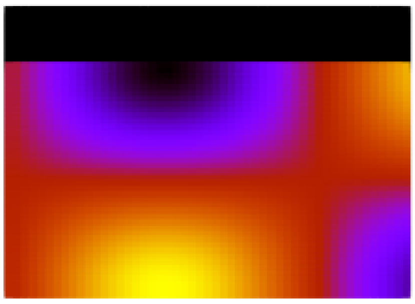


(https://quantlib.files.wordpress.com/2015/11/laplace_des_striptop_10.png)

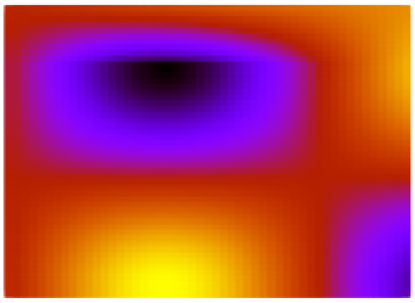


(https://quantlib.files.wordpress.com/2015/11/laplace_rec_striptop_10.png)

for 20%

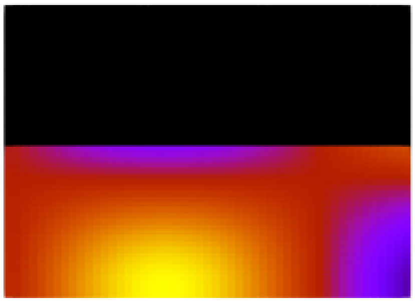


(https://quantlib.files.wordpress.com/2015/11/laplace_des_striptop_20.png)

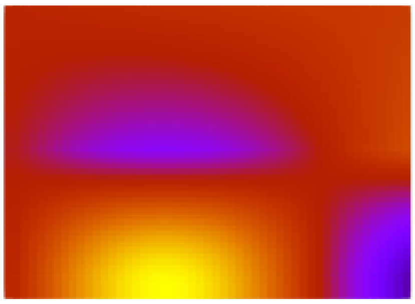


(https://quantlib.files.wordpress.com/2015/11/laplace_rec_striptop_20.png)

and for 50%



(https://quantlib.files.wordpress.com/2015/11/laplace_des_striptop_50.png)



(https://quantlib.files.wordpress.com/2015/11/laplace_rec_striptop_50.png)

Of course the method can not see what is no longer there. Still it is working surprisingly well if you ask me.