REU Week 3 Writeup

Sayem Hoque

July 3, 2017

1 Bake and Shake Algorithm

In the "bake" step of the process, the perona-malik anisotropic diffusion algorithm is used to get information from the jittered image that will help build the cleaned version of the image. The idea is to go through each row and find an ideal shift amount per row to get as close as possible to the ideal image (before the image was jittered).

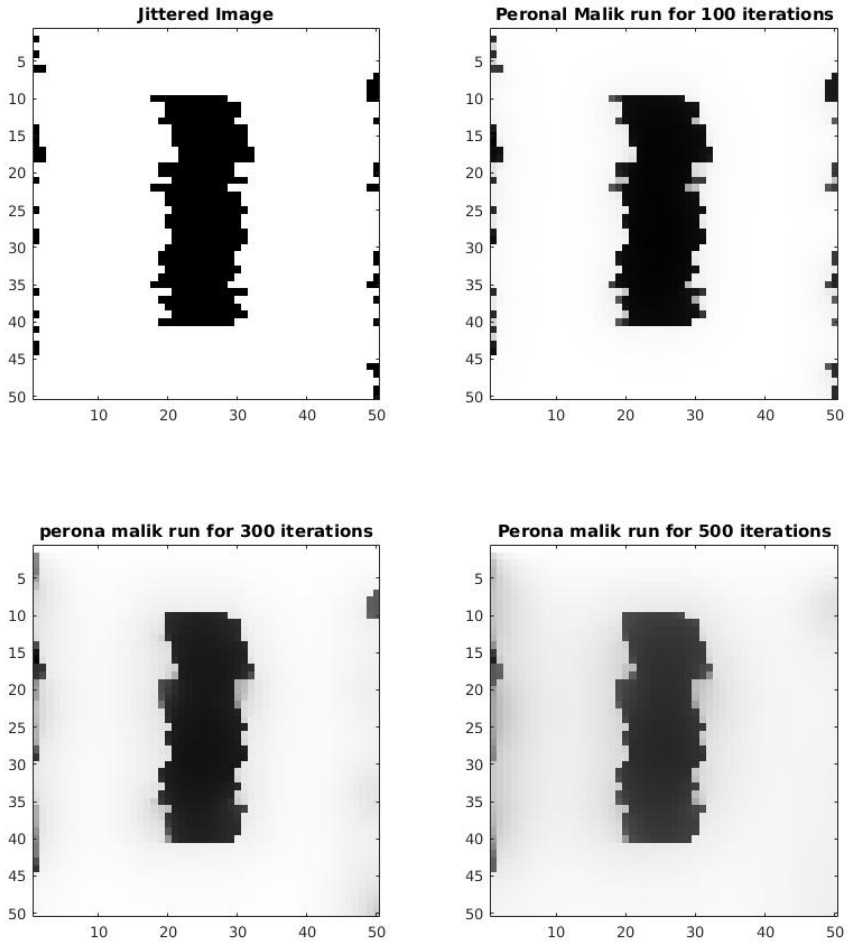
‘peronamalik.m‘

```
function u = peronamalik(u, iterations)


a = 10;
u = double(u);

dt = 0.1;
h = 1;
[r,c] = size(u);

for t = 1:iterations

        uxf = (u(:,[2:c,c]) - u) ./ h;
        uyf = (u([2:r,r],:) - u) ./ h;

        %uxb = (u - u(:, [1, 1:c-1])) ./ h;
        %uyb = (u - u([1, 1:r-1], :)) ./ h;

        p = ((uxf.^2 + uyf.^2).^0.5);
        % p = ((uxb.^2 + uyb.^2).^0.5);
        gu = 1 ./ (1 + p.^2 ./ a.^2);

        %gu = 1 ./ ((1 + p.^2) ^0.5);

        guxf = gu .* uxf;
        guyf = gu .* uyf;
```

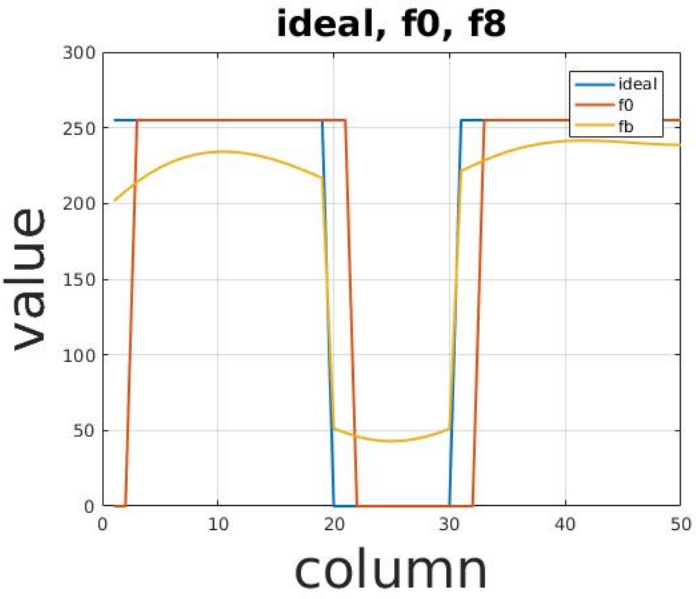
```
        guxfb = (guxf - guxf(:, [1, 1:c-1])) ./ h;
        guyfb = (guyf - guyf([1,1:r-1], :)) ./ h;

        u_out = u + dt .* (guxfb + guyfb);
        u = u_out;
end

end
```

After each image is given a random jitter, Heat Equation, TVR, and perona malik at different iterations are compared. The hope is that even with the jittering of the image, the major features in the image will allow the anisotropic diffusion to let those features become more apparent again.

1.1 Images with perona malik run



As it can be seen, running perona malik for a little longer allows the image to start to form into the shape of it's original rectangle more and more. A randomly selected profile of one row of this image is shown below. As it can be seen, in this row (row 24), there are three versions of row 24 plotted. The ideal image (before the jitter), the jittered image, and the image after it has been baked according to perona malik. The baked image is to be used to shift the rows from the jittered image into place as best as possible.



The jittered image is shown as f0. As it can be seen, the entire row is shifted to the right in this row. The fb however was found without any access to the ideal image. You can see that perona malik after 500 iterations converged right on target with where the rectangle was in the original image. Although the edges were slightly messed up, f0 serves as a good baseline to solve for the jitter s_n using newton's method.

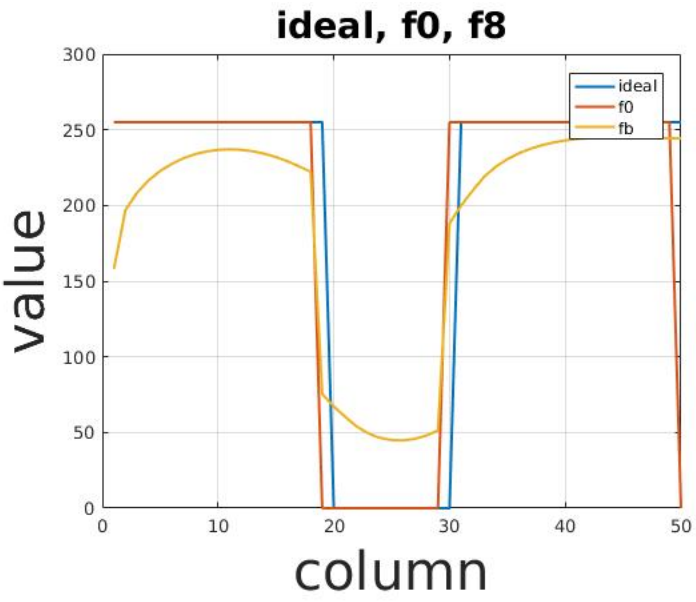
Some more rows are shown to show how different rows had different jitters. The hope is for each row, to slowly shift the row from fb towards the direction the f0 row went during the bake step. This would hopefully converge the row as close as possible to the ideal image.

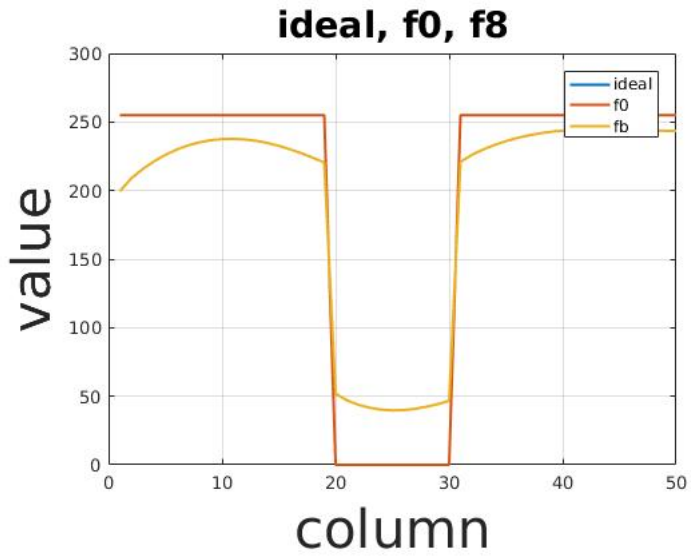
In this plot of row 20, the perona malik baked row is still not converged with the ideal image. Because of this, and the fact that f0 - fb is going to be essentially 0, the shift won't do what it necessarily needs to for this row.

In some rows such as this last one, the jitter was so small or zero. In these cases, the shift will not occur.

2 Shake code

```
% shake.m
function [] = shake()

I = 255 * ones(50, 50);
I(10:40, 20:30) = 0;
```






```
I2 = jitter(I, 2);
I3 = peronamalik(I2, 500);

[r, ~] = size(I3);

abserr = 0.1;
itmax = 5;
s0 = 0;

I2o = I2;

for row = 1:r
  xk = s0;
  xk1 = 0;
  k = 1;
```

```
  sn = s0;


  row = 20;

  ideal = I(row, :) %ideal
  f0 = I2(row, :); %f_original
  fb = I3(row, :); %f_baked
  dxfb = fb([2:end, 1]) - fb;

  fontSize = 30;
  numberOfPoints = 20;
  %'
  compare = [ideal; f0; fb]';

  % Now plot them all.
  plot(compare, 'LineWidth', 1.5);
  title('ideal, f0, fb', 'FontSize', 20);
  xlabel('column', 'FontSize', fontSize);
  ylabel('value', 'FontSize', fontSize);
  grid on;
  legend('ideal', 'f0', 'fb');

  2*dot((f0 - fb), dxfb) / (dot(abs(dxfb), abs(dxfb)))
  sn1 = round(sn - 2*dot((f0 - fb), dxfb) / (dot(abs(dxfb), abs(dxfb))))

  while (k < itmax)
    sn = sn1;
    sn1 = round(sn - dot((f0 - fb), dxfb) / ((dot(abs(dxfb), abs(dxfb)))));
    f0 = shift(f0, -sn1);
    k = k + 1;
  end

  I2(row, :) = f0;

end



end
```