This document presents a brief tutorial and description of ripple image processing code developed by Adam Skarke (adam.skarke@msstate.edu) with contributions by Anna Crawford (DRDC), Christopher Englert (UNH), and Carter Duval (U. Del.). The code is based in part on a set of functions created by Peter Kovesi and Raymond Thai (see references section). The concept of applying fingerprint analyses approaches to bedform imagery was initially motivated by the work of Janice Felzenberg (2009). This code is written for Matlab and requires access to the Matlab image processing toolbox. See commented sections in associated code for full documentation of all input variables.

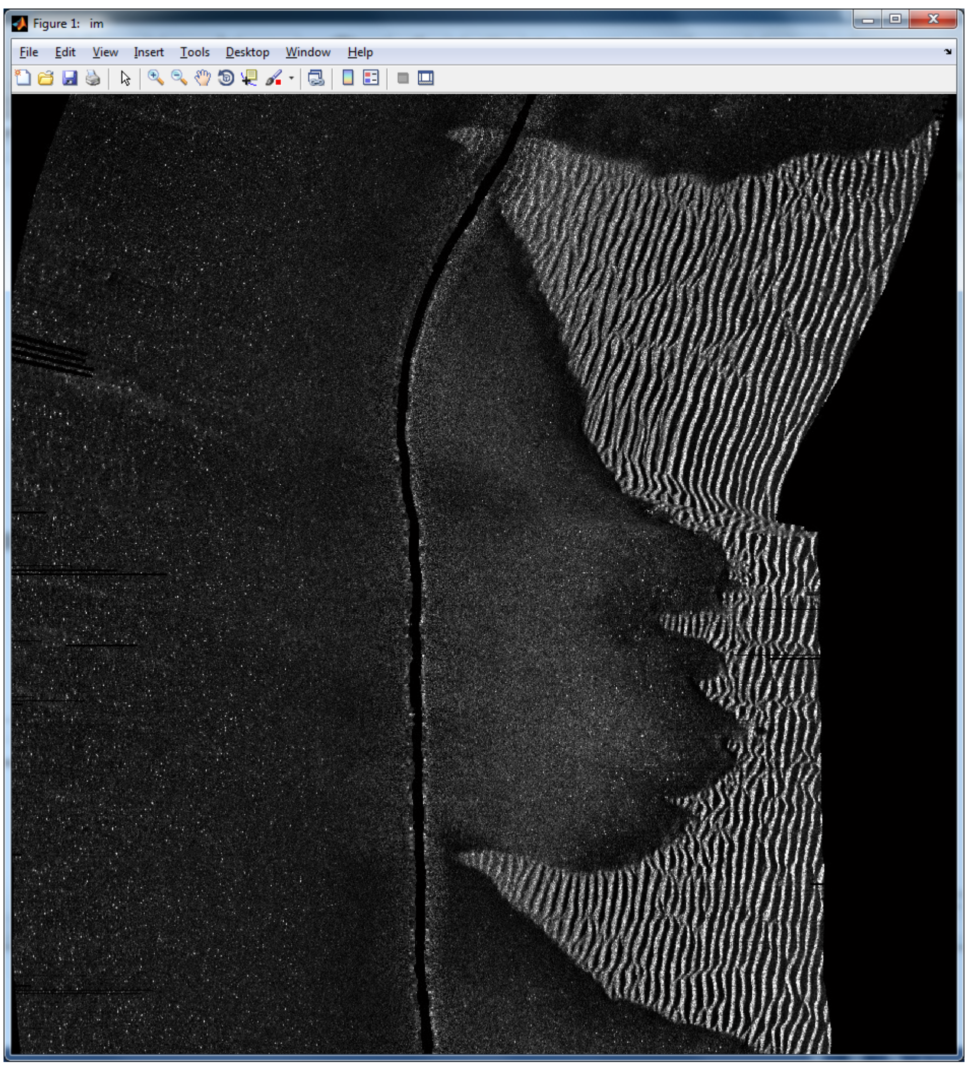
Tutorial:

1) Import image into Matlab workspace as a matrix and convert to grayscale:

im = imread ('SampleRipplesCrop.jpg');
im = rgb2gray (im);

2) Visually check results using the Matlab function imshow:

imshow (im);

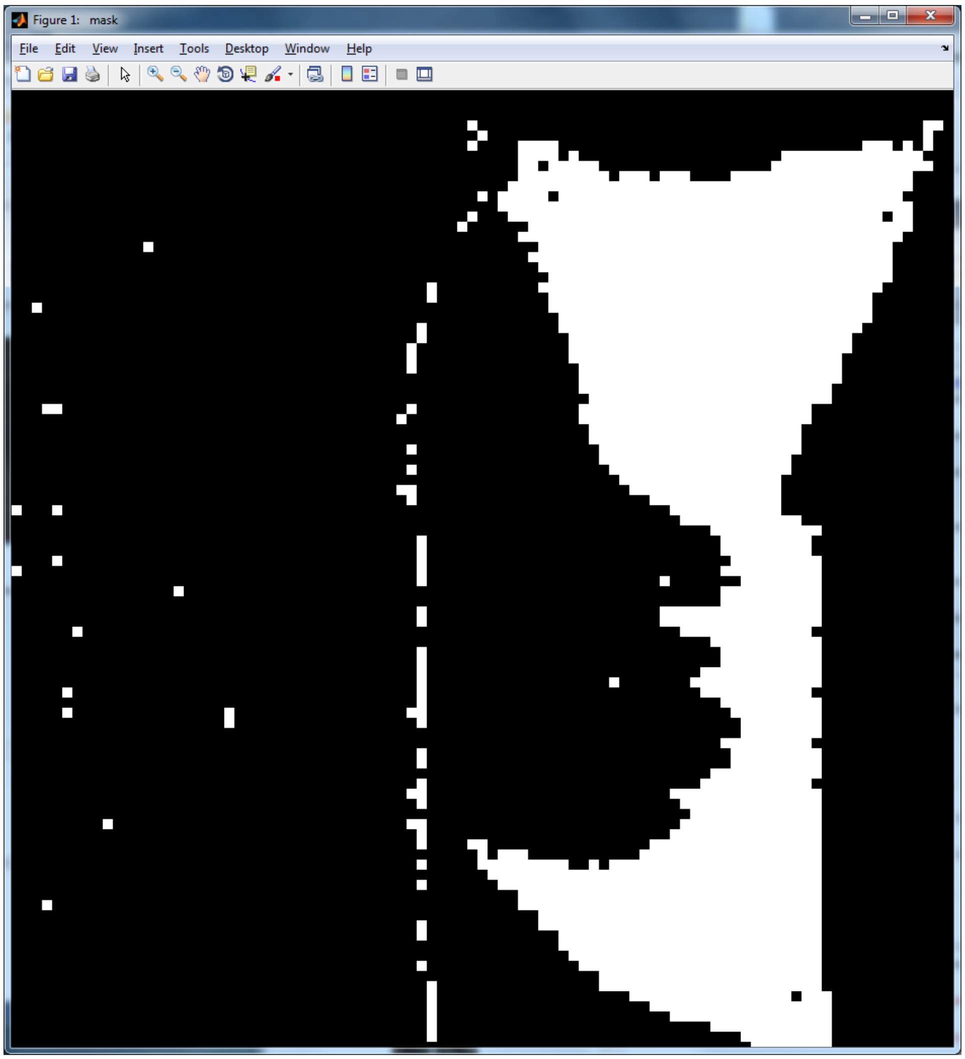


3) Identify regions of the image with high isotropic variance (first order indication of rippled regions) and normalize backscatter intensity values to a mean of zero:

[normim,mask] = ridgesegment (im, 10, 0.9);

4) Visually evaluate if the mask is approximately delineating rippled vs. non-rippled areas:

imshow (mask);



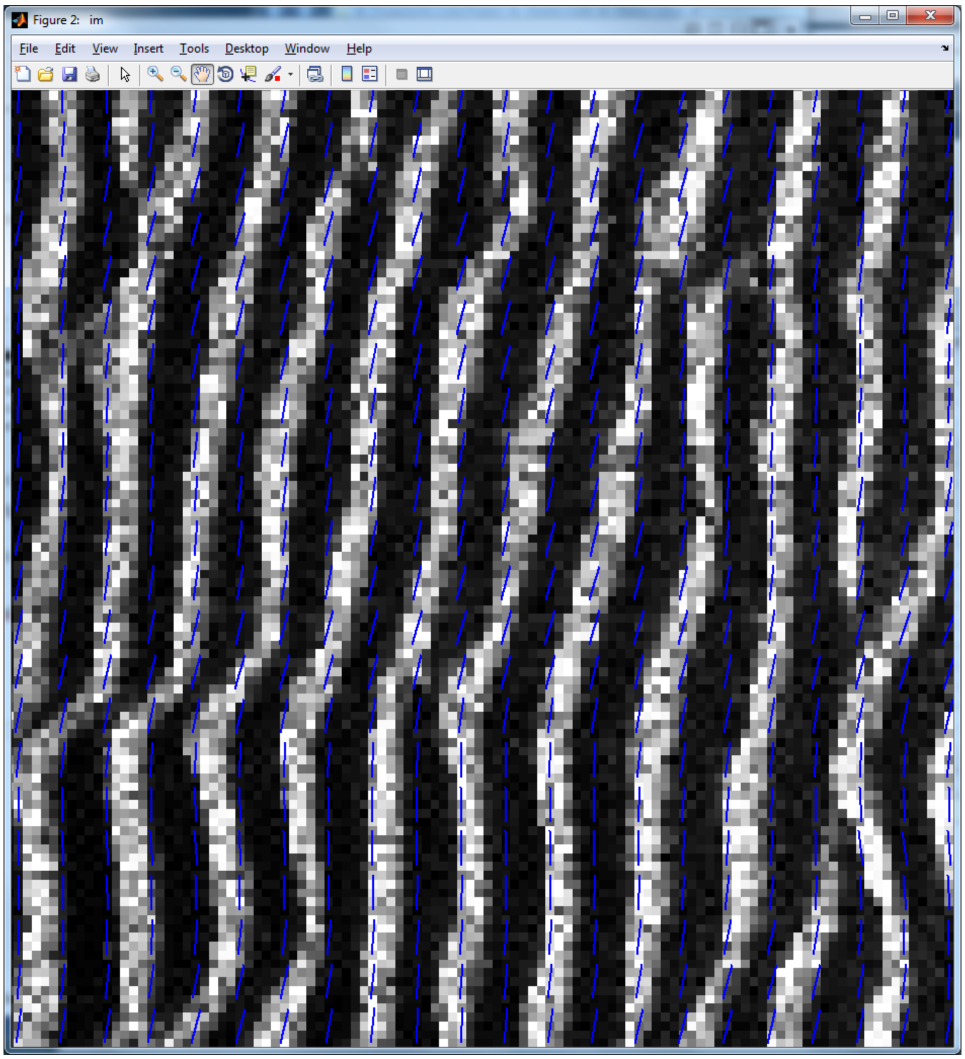
This masking code does a reasonable job of approximating rippled portions of the image. However, it exhibits issues with delineating the precise boundaries of the rippled area and gets tripped up by the nadir. This mask is substantially refined, based upon anisotropic variance, in subsequent processing steps.

5) Calculate ridge orientation and ripple “reliability” across image:

[orientim, reliability] = ridgeorient(normim, 1, 5, 5);

6) Plot progressive vector diagram of ridge orientations over image to check for agreement:

plotridgeorient (orientim, 5, im, 2);



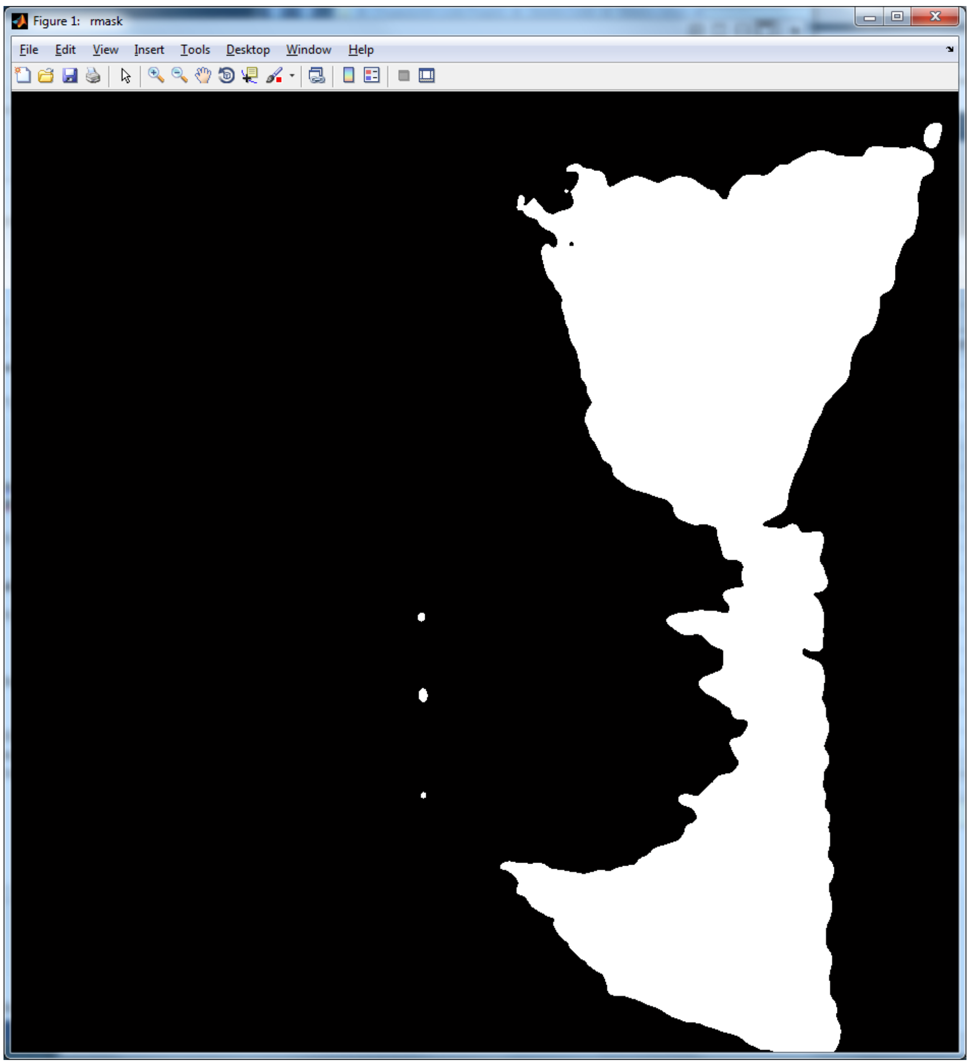
Here the image is zoomed in to allow for visual comparison of subsampled orientation vectors (blue lines) with ripple morphology.

7) Identify regions of the image with high variance perpendicular to the determined ripple orientation relative to variance parallel to ripple orientation (areas of high ‘reliability’) in order generate a logical mask of rippled vs non rippled areas.

```
reliability(reliability>0)=1;
rmask=logical(reliability);
```

8) Visually evaluate if the mask is accurately delineating rippled vs. non-rippled areas:

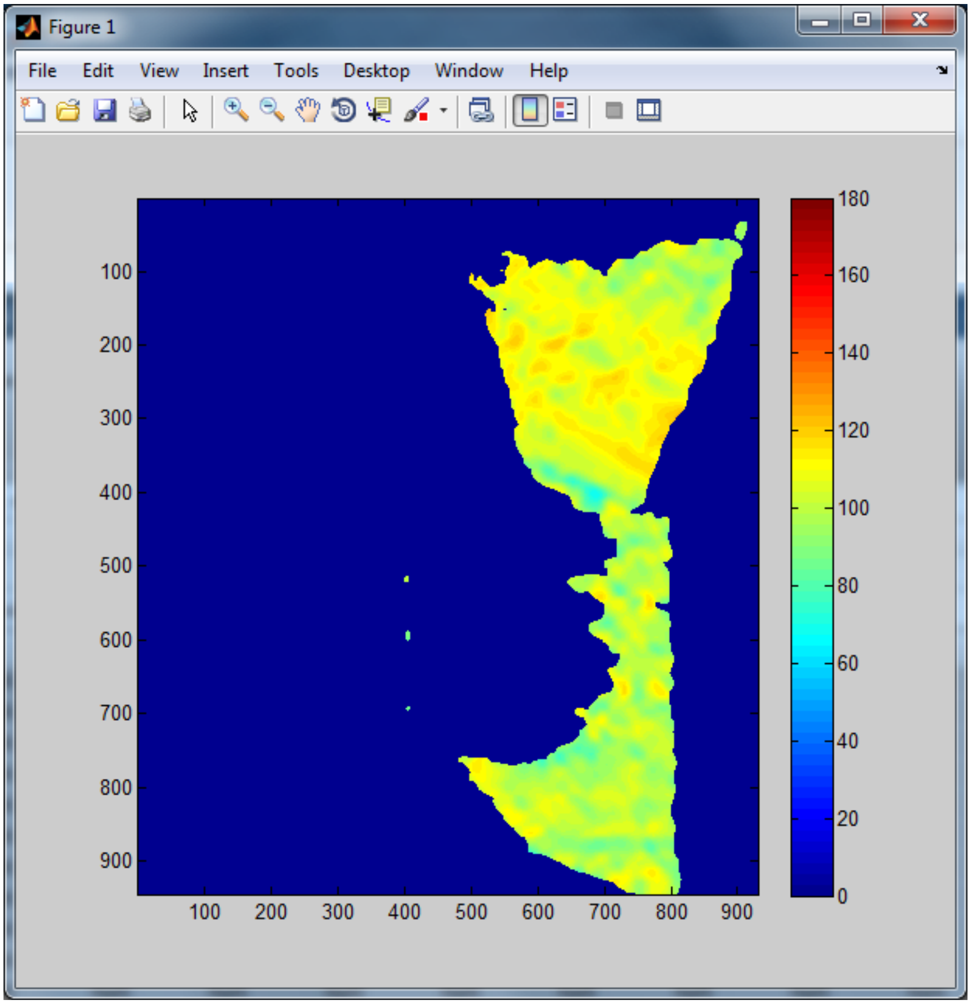
```
imshow(rmask);
```



With the exception of a few issues right at nadir, this anisotropic mask does a good job of isolating the areas with ripples in the image.

The calculated ripple orientation and "reliability" mask allow for the plotting of ripple orientation:

imagesc((orientim.*180/pi).*rmask);colormap jet;caxis([0 180]);colorbar



Note: Orientation results for this example are in Matlab convention ("up"=90°). Orientation results for real data must be modified to reflect the real world georeference.

To calculate mean orientation of ripples:

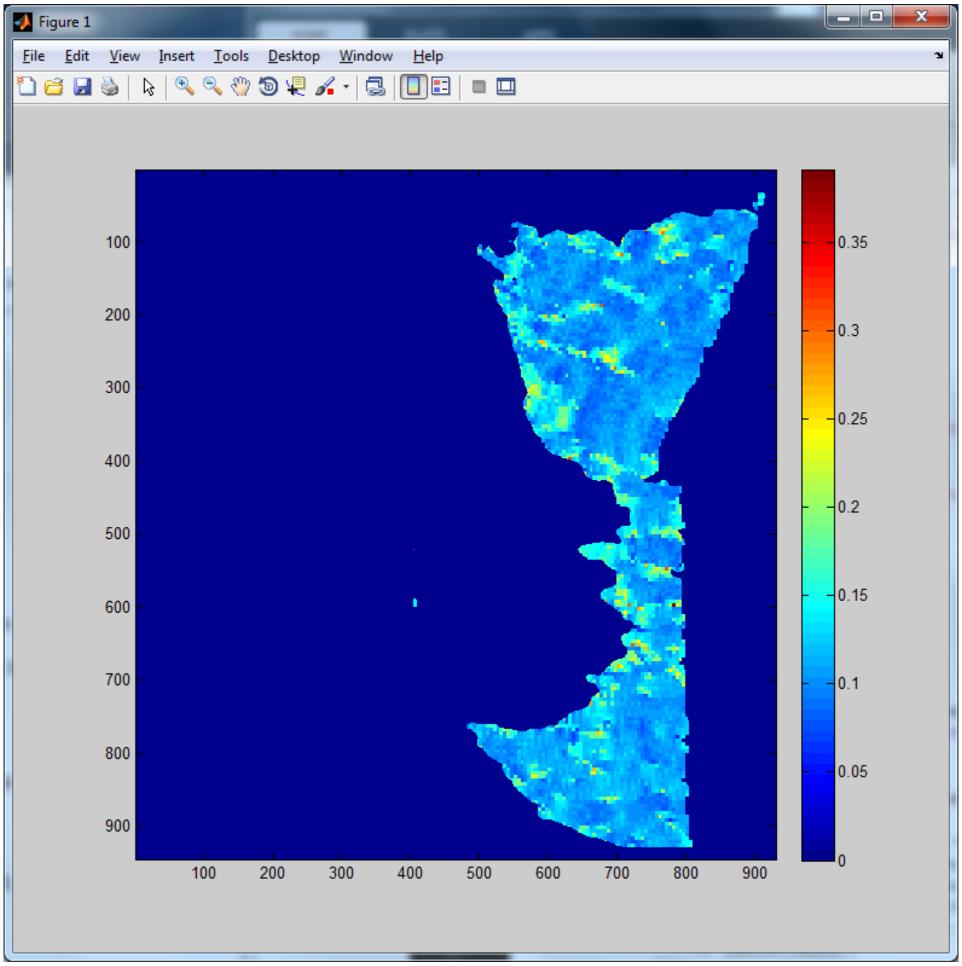
meanorient=(mean(orientim(find(rmask==1)))).*180/pi;

Mean, median, and standard deviation of ripple orientation are often very useful but the specific statistical tools you choose will of course, depend upon the nature of your research problem.

9) Calculate the frequency of ripples:

[freq] = rf3(normim, rmask, orientim, 33, 10, 0, 1000);

imagesc(freq.*rmask);colormap jet;colorbar;



I have incorporated blockwise spatial filtering here. The results are consistent with a visual estimation of ripple wavelength in the initial image (5-10 pixels). Note: frequency results are in units of pixels. For real word data this can be converted to wavelength in units of meters using the native image resolution.

To calculate mean frequency of ripples:

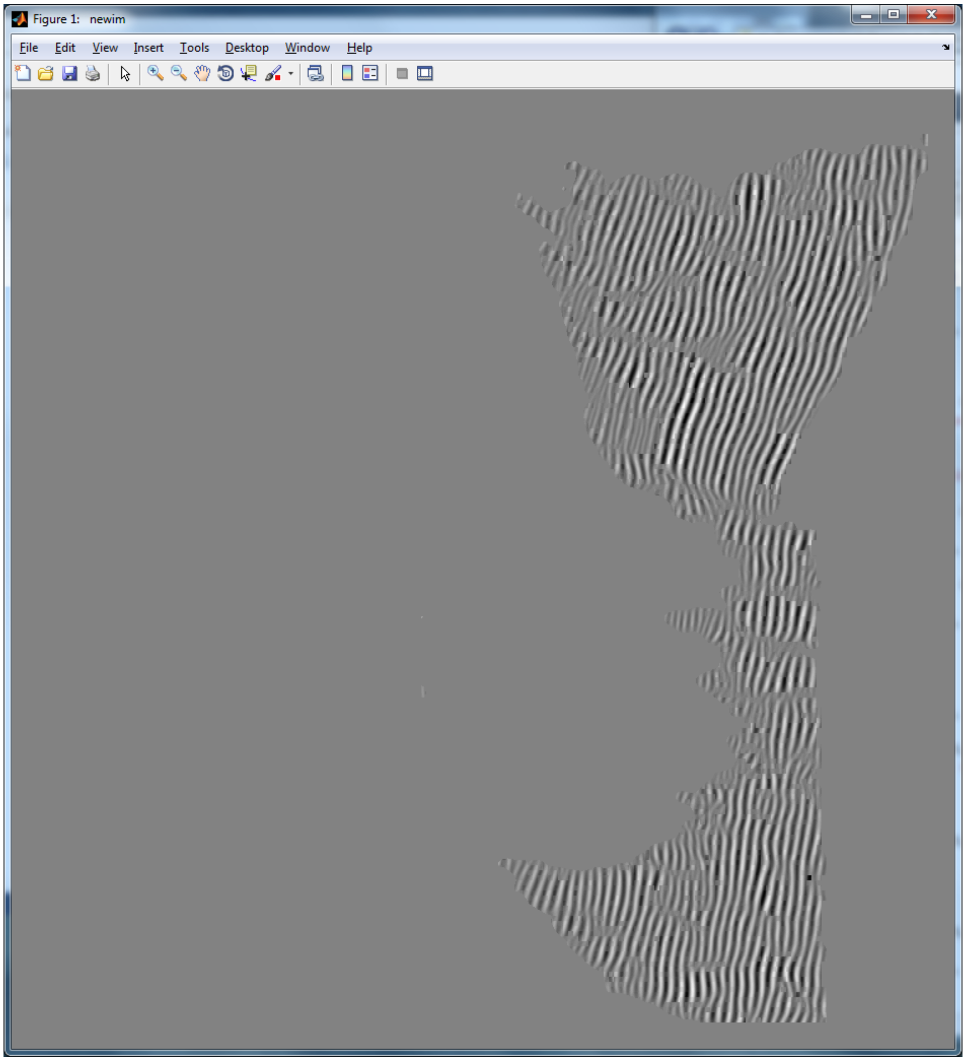
meanfreq=mean(freq(find(rmask==1)));

Mean, median, and standard deviation of ripple frequency are often very useful but the specific statistical tools you choose will of course, depend upon the nature of your research problem.

10) Filter image for noise:

newim=ridgefilter(normim, orientim, freq, 0.25, 0.25, 0);

show(newim);

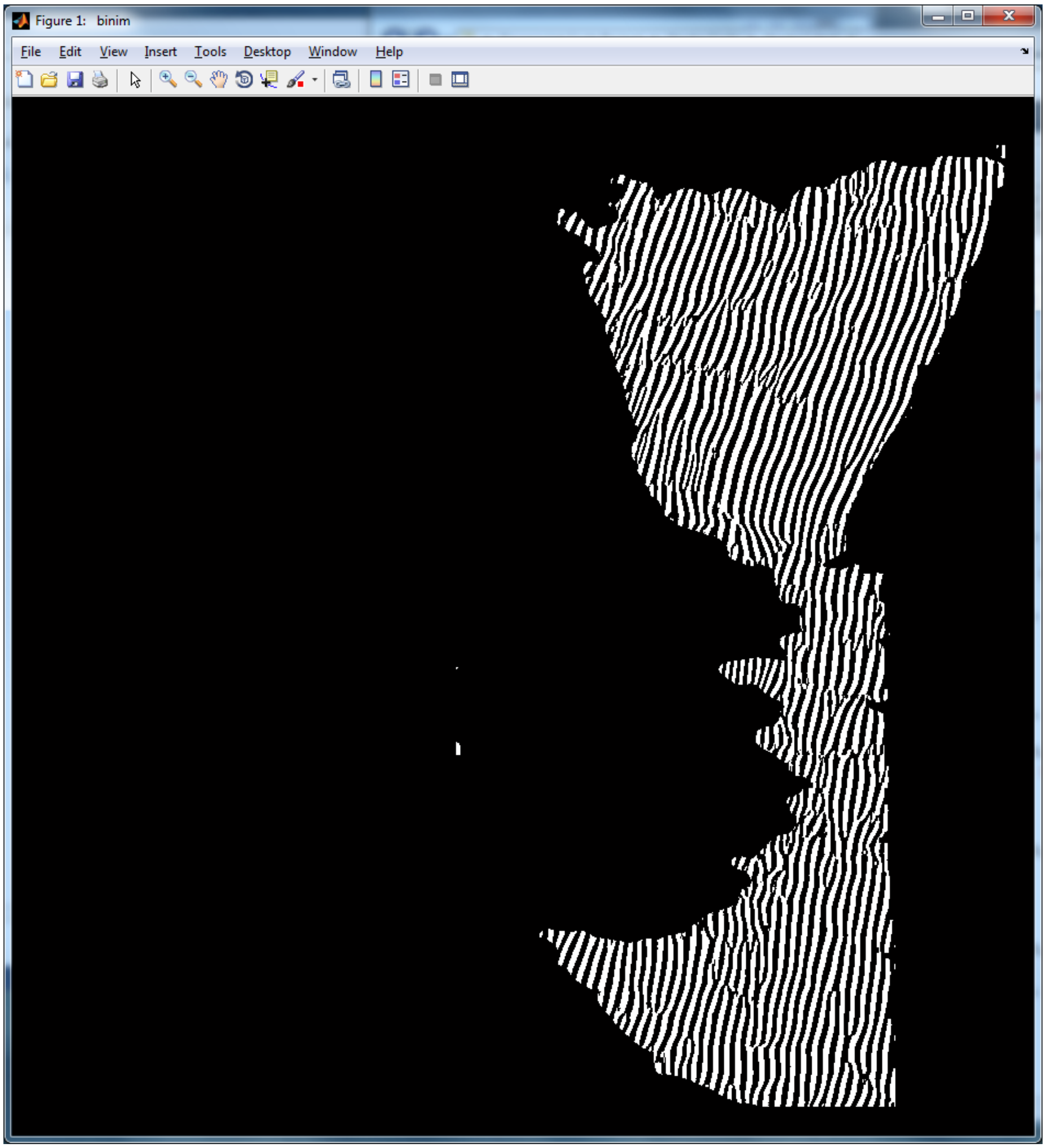


The quality of the filtered image will largely depend upon the determination of ripple frequency and the settings of tunable filter parameters. Refer to commented sections in associated code for documentation of variables that control filter parameters.

11) Binarize the filtered image:

binim=newim>0;

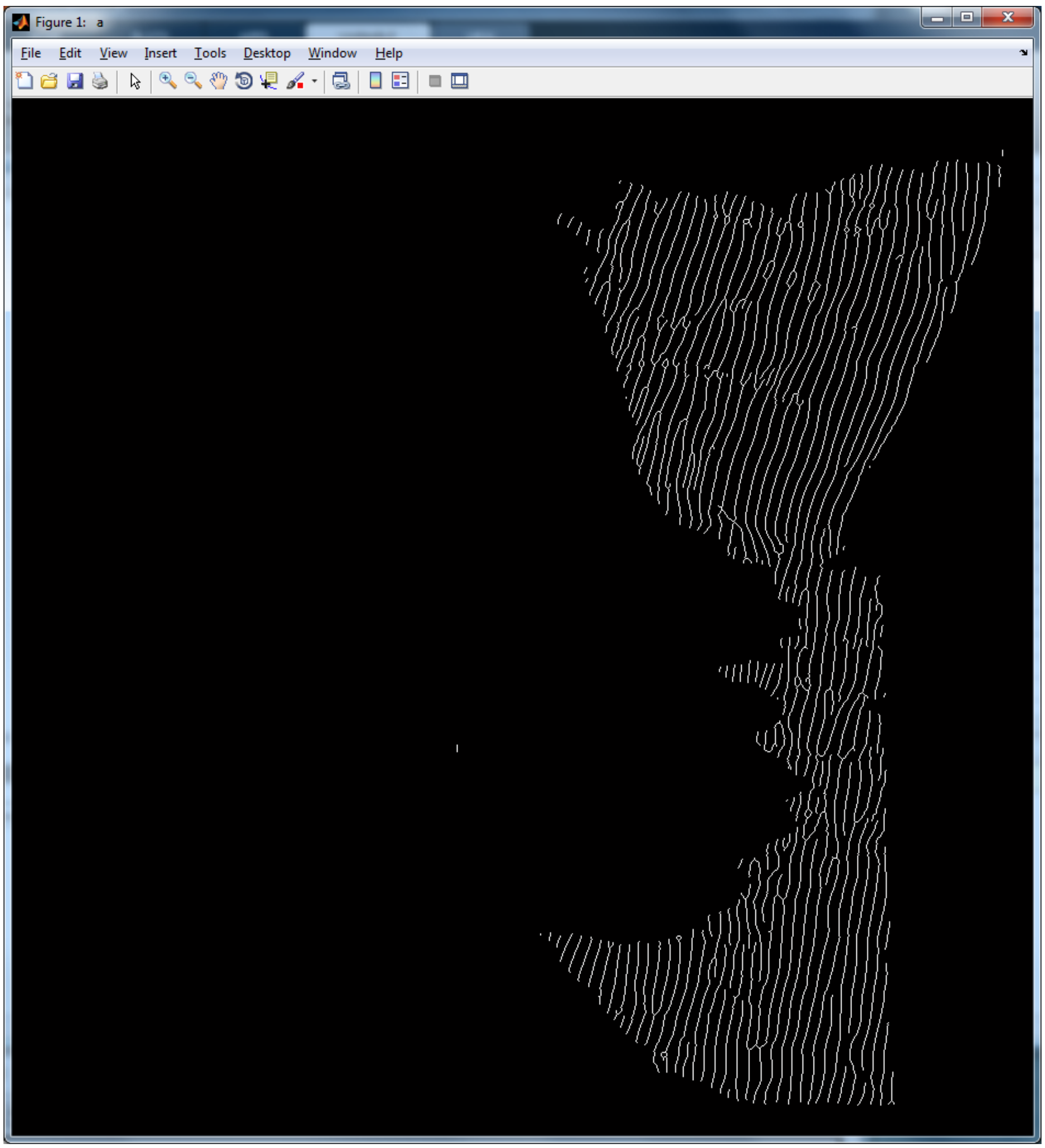
imshow(binim);



This is a good point to evaluate the resulting image in terms of how accurately it preserves ripple structure observed in the initial image.

12) Thin binarized lines to create a map of ripple crest with a single pixel wide line:

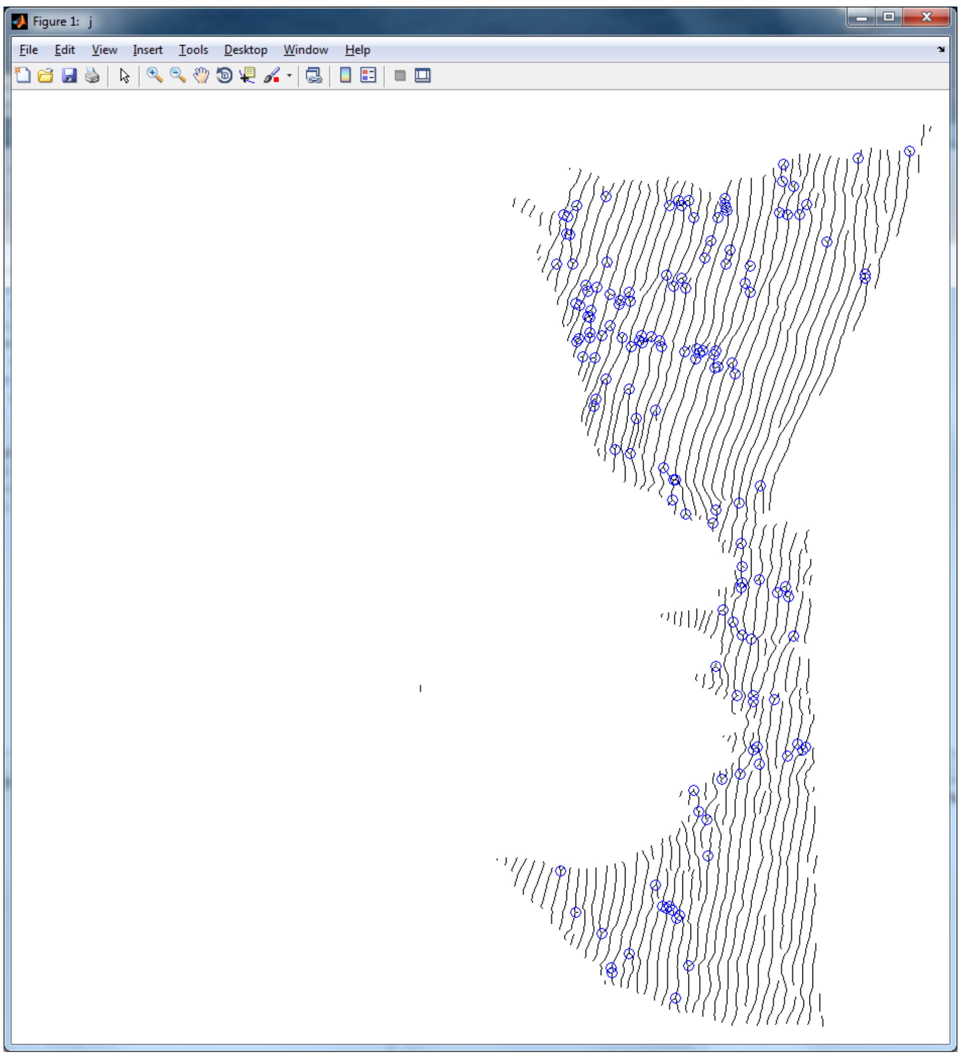
```
a=bwmorph(binim, 'thin', Inf);
a=bwmorph(a, 'fill');
a=bwmorph(a, 'thin', Inf);
a=bwmorph(a, 'spur');
a=bwmorph(a, 'thin', Inf);
a=bwmorph(a, 'spur');
a=bwmorph(a, 'clean');

imshow(a);
```



This seems to compare well to the original image as far as isolating and preserving ripple structure.

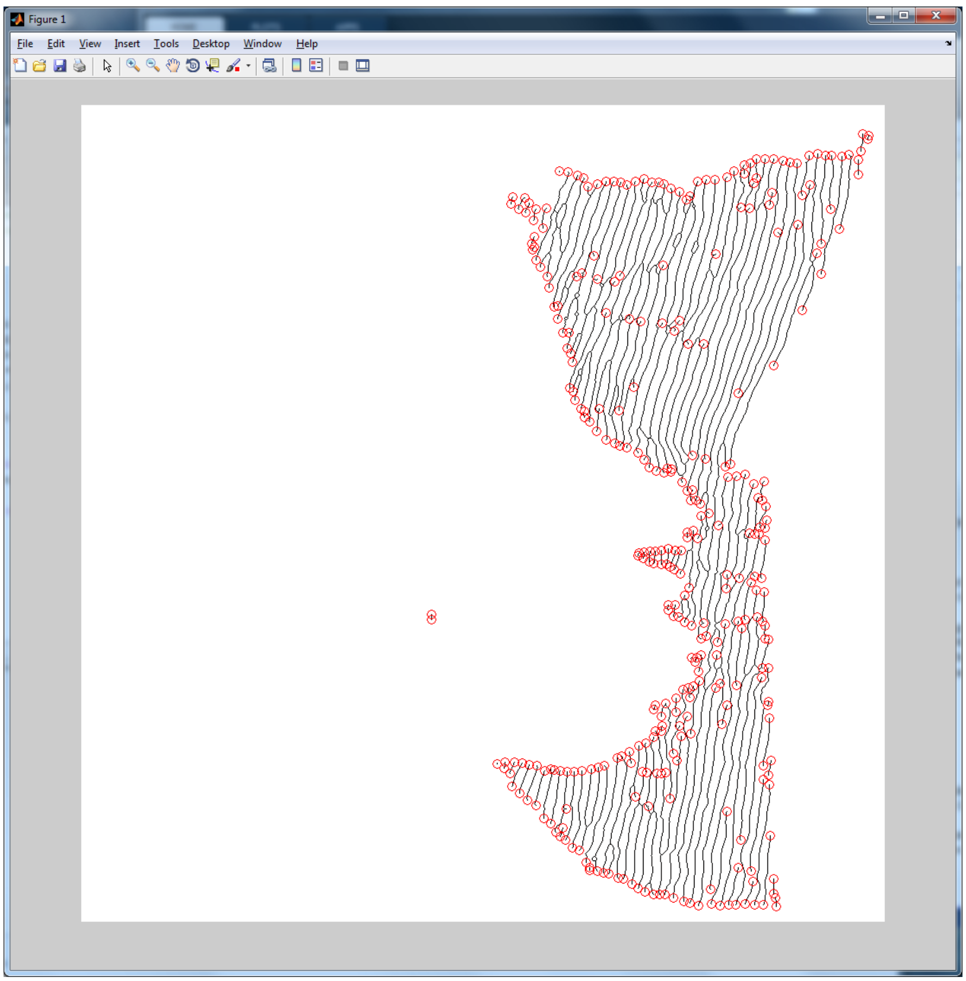
13) Isolate bifurcations:

```
c=a;
m=find(a==1);
for g=1:length(m);
  [X, Y] = ind2sub([size(a)],m(g));
  b=a((X-1):(X+1),(Y-1):(Y+1));
  if sum(b(:))==4;
    c(X,Y)=1;
  else
    c(X,Y)=0;
  end
end
c=bwmorph(c,'shrink', Inf);
[Xb, Yb] = ind2sub([size(c)],find(c==1));
j=abs(a-1);
show(j); hold on; plot(Yb,Xb,'o','MarkerSize',8);
```



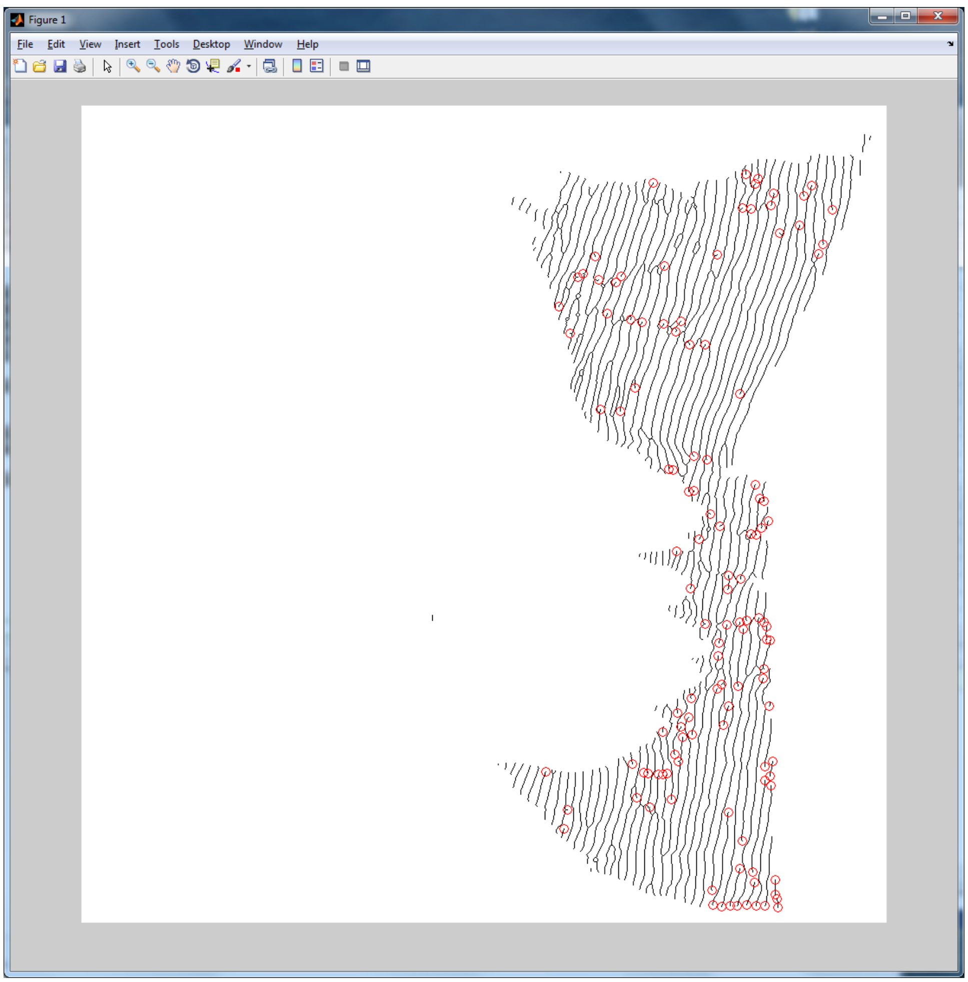
14) Isolate terminations:

```
d=a;
m=find(a==1);
for g=1:length(m);
   [X, Y] = ind2sub([size(a)],m(g));
   b=a((X-1):(X+1),(Y-1):(Y+1));
   if sum(b(:))==2;
      d(X,Y)=1;
   else
      d(X,Y)=0;
   end
end
d=bwmorph(d,'shrink', Inf);
[Xt, Yt] = ind2sub([size(d)],find(d==1));
j=abs(a-1);
imshow(j); hold on; plot(Yt,Xt,'o','MarkerSize',8,'MarkerEdgeColor','r');
```



Because this example image contains a sorted bedform (i.e. rippled scour depression) there are numerous terminations along the edge of the feature. It may be desirable for certain analyses to differentiate ripple terminations at the edge of a rippled area from those within it. A method for doing this is presented below. Data specific approaches to this problem may be needed.

```
emask=abs(rmask-1);
fmask=bwmorph(emask,'dilate',3);
h=logical(a+fmask);
d=a;
m=find(a==1);
for g=1:length(m);
  [X, Y] = ind2sub([size(a)],m(g));
  b=h((X-1):(X+1),(Y-1):(Y+1));
  if sum(b(:))==2;
    d(X,Y)=1;
  else
    d(X,Y)=0;
  end
end
d=bwmorph(d,'shrink', Inf);
[Xt, Yt] = ind2sub([size(d)],find(d==1));
j=abs(a-1);
imshow(j); hold on; plot(Yt,Xt,'o','MarkerSize',8,'MarkerEdgeColor','r');
```



After desired bifurcation and termination locations have been identified, producing estimates of ripple defect density is straight forward, using the image pixel resolution.

References:

Crawford, A., and A. Skarke, 2014. Automatic detection of sand ripple features in sidescan sonar imagery. *Proceedings of MTS/IEEE Oceans 2014,* Saint Johns, Newfoundland, Canada, Sept 14-19. doi:10.1109/oceans.2014.7003117

Felzenberg, J., 2009. Detecting bedform migration from high-resolution multibeam bathymetry in Portsmouth Harbor, New Hampshire, USA. M.S. Thesis, University of New Hampshire, Durham, NH.

Hong, L., Wan, Y., and Jain, A. K., 1998. Fingerprint image enhancement: Algorithm and performance evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 20 (8): 777-789.

Skarke, A., and A.C. Trembanis, 2011. Parameterization of bedform morphology and defect density with fingerprint analysis techniques. *Continental Shelf Research,* 31: 1688-1700.

Thai Thesis:
http://www.peterkovesi.com/studentprojects/raymondthai/RaymondThai.pdf

Kovesi Research:
http://www.peterkovesi.com/matlabfns/FingerPrints/Docs/index.html