ASSIGNMENT ON WITHIN-IMAGE AND BETWEEN-IMAGE ERROR

DUE DECEMBER 11, 2017 BY MIDNIGHT

1. Study the error distribution of Zhang's quantitative detector

The purpose of this assignment is to study the statistical properties of the within-image and between-image error of quantitative detectors. In particular, you will work with the Zhang's quantitative detector.

1.1. **Between-image error.** Use the supplied 'Jsteg_det.m' routine. For a JPEG image on its input (or a matrix of quantized DCT coefficients), this routine outputs an estimate of the change rate β due to Jsteg embedding (the relative payload α would be $\alpha = 2\beta$). Apply it to all 1000 JPEG images from archive '1000JPEGimages.zip' from Blackboard. All 1000 images are all 85% JPEG quality and they are all *cover* images. As part of your submission, plot the histogram of the change-rate estimates β returned by 'Jsteg_det.m.' The histogram should be centered (peak) around zero. The estimates of β from cover images are the between-image errors. Choose the number of bins and their width for the histogram wisely so that the chart has a good information value (see the example below).

Create the log-log empirical plot (see Appendix A.10 or the PowerPoint slides 'quant-steg-errors.pptx') for the *right tail* of the change-rate estimates distribution only (the right tail is formed by all positive estimates β). You can either use the 'loglog.m' routine in Matlab or simply use 'plot.m' for the logarithm of both x and y coordinates of the plotted data. Notice the range in which the right tail in the log-log plot becomes approximately a straight line. Fit a line through the data points laying only in this range (Matlab command 'regression.m') and include the line fit in your log-log plot. Report the value of the line's slope – this is the number of degrees of freedom, ν, for the Student's t-distribution model of the right tail.

1.2. **Within-image error.** Select one image from the 1000 JPEG images used in Task 1.1. Simulate embedding in this image by flipping the LSBs of randomly-selected $\beta_0 = 0.2$ non-zero non-one DCT coefficients. You can use the supplied routine 'Jsteg_simulator.m' for this task. Run 'Jsteg_det.m' on the modified image and estimate the change rate (it should be close to β_0). Repeat this 10,000-times, each time starting with the same cover image and β_0 but using a different seed for the PRNG driving the location of flipped pixels (the seed is an input to 'Jsteg_simulator.m'), and each time run the 'Jsteg_det.m' estimator of β. Denoting your estimates β_i, $i = 1, \ldots, 10,000$,

- Compute and report the average change-rate estimate:

$$\overline{\beta} = \frac{1}{10,000} \sum_{i=1}^{10,000} \beta_i.$$

- Compute and report the bias $\overline{\beta}-\beta_0$. This is the value of the between-image error for the image you selected.
- Plot the distribution of the within-image error $\beta_i - \overline{\beta}$ for all 10,000 images *and* its Gaussian fit. You should see a good agreement with the Gaussian fit. Compute its variance σ^2 and report it.

Remark 1: Notice that 'Jsteg_simulator' outputs the matrix of modified quantized DCT coefficients rather than a JPEG file. Use the outputted matrix of DCTs directly as an input into 'Jsteg_det' to obtain the 10,000 estimates β_i. This way you do not have to save the modified JPEG file to the disk using 'jpeg_write.m' and read it back, saving thus a lot of processing and keeping the computational time down.

Remark 2: Here is a code fraction that allows you to plot a sample distribution of a random variable together with its Gaussian fit. The samples are in vector x, while $\hat{\sigma}$ and $\hat{\mu}$ are the sample standard deviation and mean.

```
<BEGIN code fraction>
d = σ̂/10;
Range = −4σ̂ : d : 4σ̂;
plot(Range, Gaussian_density(Range,0,σ̂));
hold on
h = hist(x, Range);
h = h/sum(h);
plot(Range, h/d, 'r+')
hold off
<END code fraction>
```

Yes, you will need to write your own function for a Gaussian density with variance σ^2 and mean μ that evaluates the p.d.f. for a *range* of values (a vector). This function should not need more than a single line of code written in vector form.

Remark 3: For your batch processing, you will need to extract the file names of all files of a given extension from a specific folder. For this purpose, use the supplied 'filenames.m' or simply implement your own routine.

In order to give you some idea how your outputs should roughly look like, below I provide a sample output for 1000 JPEG images *from a different source than the one with which you will be working.* I also provide an example output across 10,000 embeddings for one selected image with the Gaussian fit.

I know this sounds like a lot of work but, in fact, it is not since I basically give you almost all the code. The simulations took a few minutes on my ancient seven-year old office desktop. Even if you do them on a less powerful laptop, they should not take an excessive amount of time. Just let your computer do the work. Good luck and do not hesitate to ask for help should you need it.

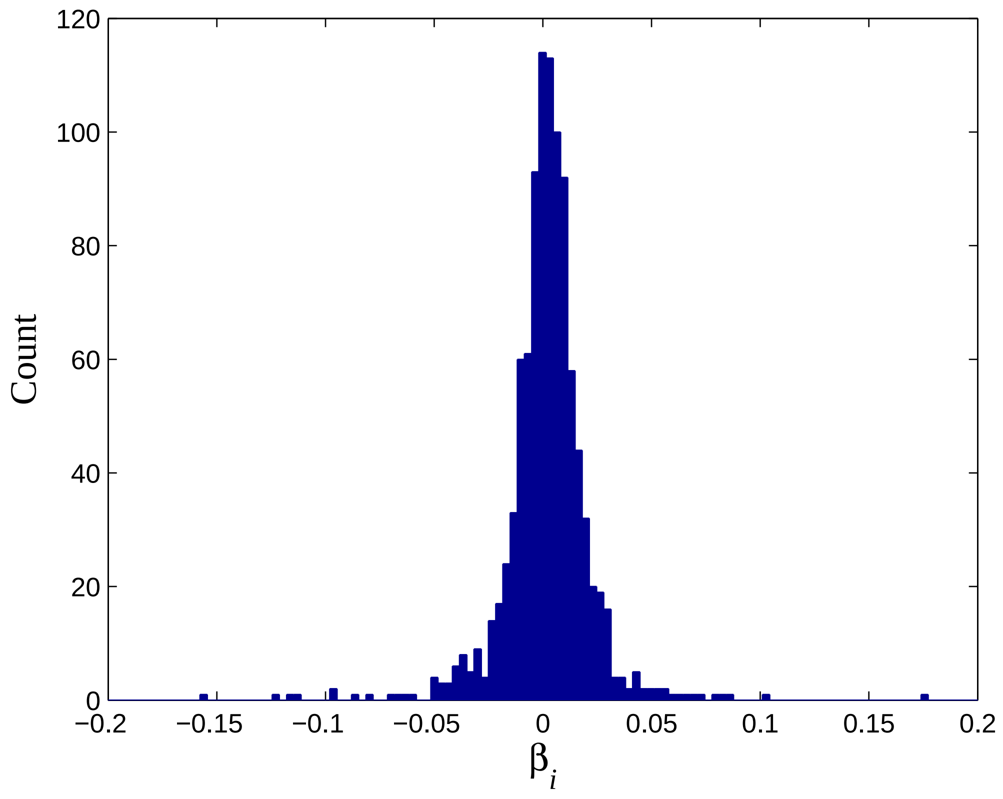

FIGURE 1.1. **Sample output Part 1.1**: Distribution (histogram) of the estimated change rate β for 1000 cover JPEG images.

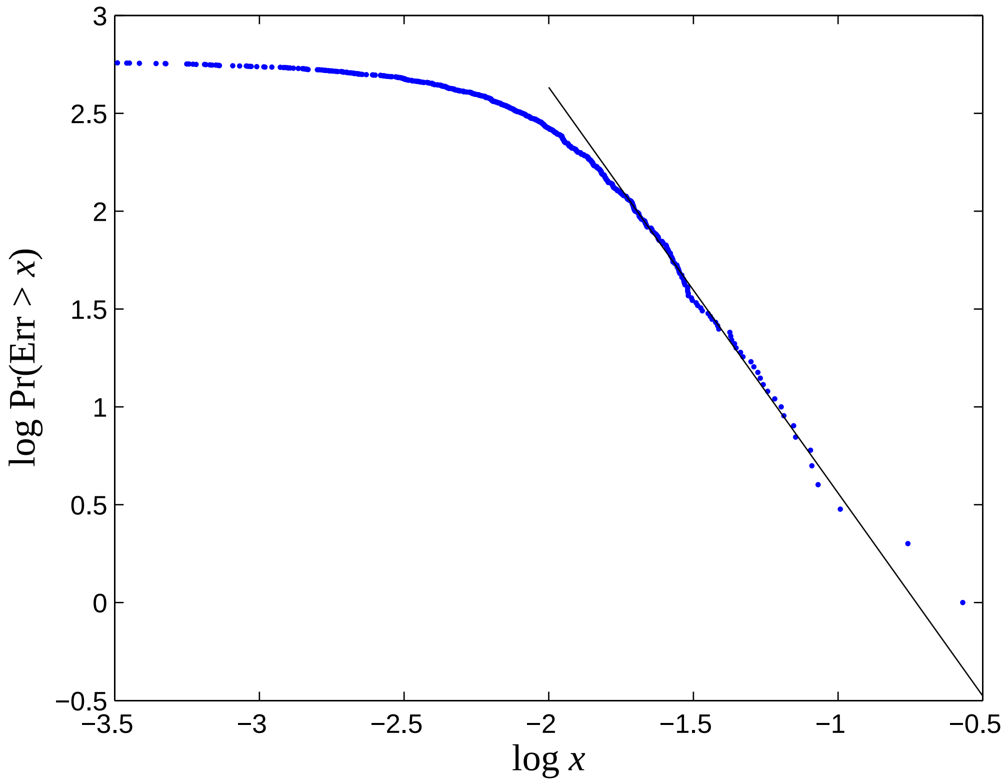

Figure 1.2. **Sample output Part 1.1:** Log-log plot of the right tail of β estimated from 1000 cover JPEG images. The slope of the line fit to the tail was obtained from points whose x values were between -2 and -1, where the log-log plot appeared linear. The slope value (the negative of the number of degrees of freedom for the Student's t-distribution) is $\nu = -2.073$. Notice that in this case, I used the 'plot.m' Matlab routine, hence the $\log x$ and $\log \Pr(\text{Err} > x)$ on the x and y axes.

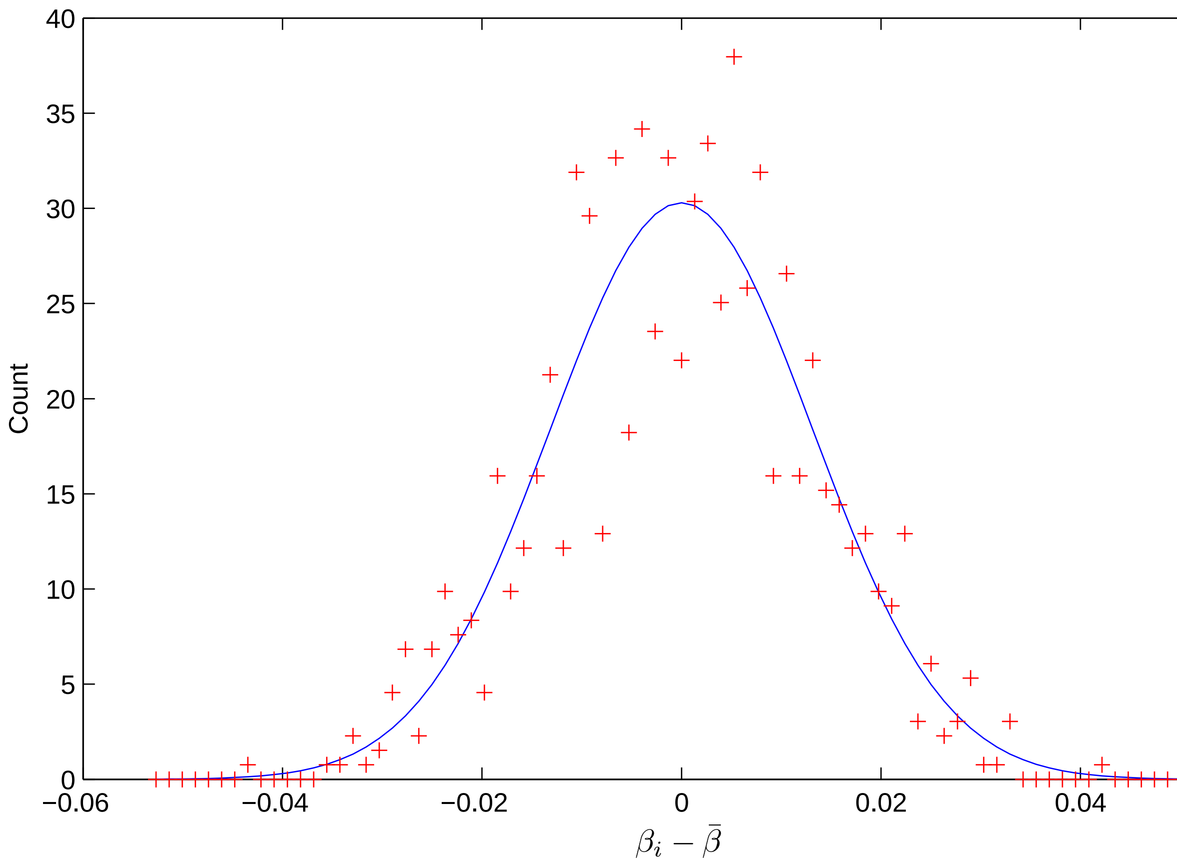

FIGURE 1.3. **Sample output Part 1.2:** Distribution of the within-image error (red crosses) for one image and its Gaussian fit (blue line) estimated from 10,000 embeddings of a fixed change rate $\beta = 0.2$ with different seeds. The bias is $\hat{\mu} = \overline{\beta} - \beta_0 = -0.00361296$, and the variance is $\hat{\sigma}^2 = 2.7 \times 10^{-5}$.