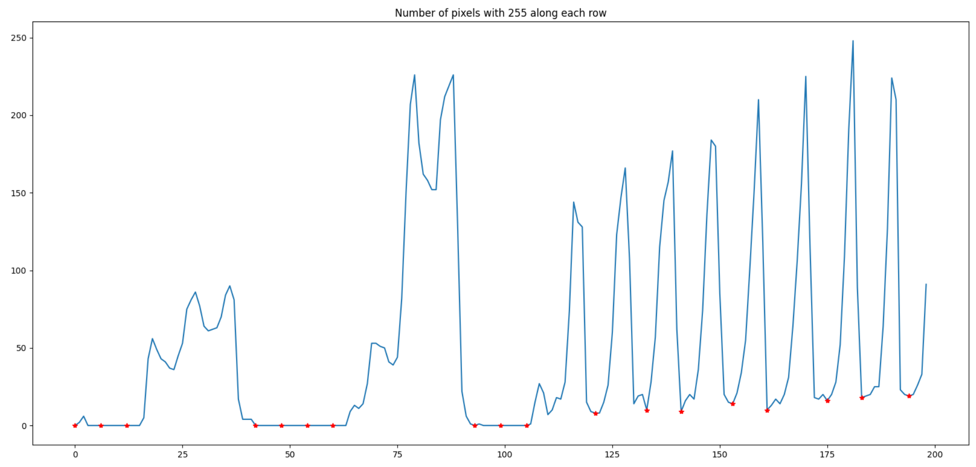
8/23/23 CAP 6415 - Professor Sarkar Victor Ejiasi

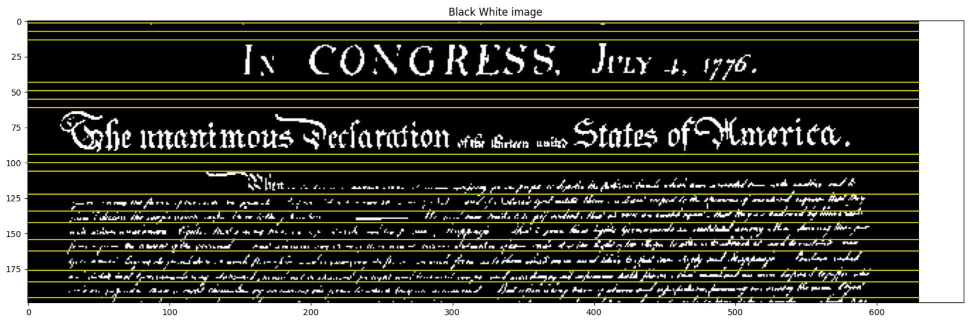
Line segmentation in documents

(a) the updated code:

```
num_255_row = np.sum(bw_doi, axis=1)

minima_locations = np.zeros(num_255_row.shape)
static_threshold = 25
window_size = 5
min_gap = 5
last_minima = -min_gap

fig, axs = plt.subplots(2, 1)
fig.set_size_inches(20, 20)

axs[0].plot(num_255_row[1:200])
axs[0].set_title('Number of pixels with 255 along each row')
axs[1].imshow(bw_doi[1:200,:], 'gray')
axs[1].set_title('Black White image')

# Iterating through rows 1 to 200
for i in range(1, 200):
    # Determining the window within which to look for local minima
    window_start = max(0, i - window_size)
    window_end = min(200, i + window_size)
    # Finding the minimum value within the window
    window_min = np.min(num_255_row[window_start:window_end])
    # Check if the current row has the minimum value within the window and if it is below the
static threshold
    if num_255_row[i] == window_min and num_255_row[i] < static_threshold:
        # Ensure that detected minima are spaced out by at least min_gap
        if (i - last_minima > min_gap):
            # Mark the minima on the graph of white pixel counts
            axs[0].plot(i-1, num_255_row[i], 'r*')  # The plot starts at horizontal axis = 1 not 0.
            axs[1].plot(np.array([0, bw_doi.shape[1]]), np.array([i, i]), 'y')
            minima_locations[i] = 1
            last_minima = i

plt.show()
fig.savefig('plot_output.png')
```

(b) the output generated:





(c) explanation of why you modified the logic in the manner that you did:

The primary objective was to detect the local minima in the histogram of the number of white pixels across rows. These minima indicative the gaps between lines of text in the thresholded image. The code provided used the binary thresholded image (bw_doi), where text is represented by white pixels and the background by black pixels. In the sum across rows (num_255_row), high values represent lines of text. The initial logic did have a few areas for potential improvement. Such as;

Introduction of Variables for Flexibility - I introduced static_threshold, window_size, and min_gap as variables. This makes the code more readable but also allows for easy adjustments in the future. I can easily tweak these parameters in the future as needed.

Window-Based Minima Detection - The original code was looking for minima based on immediate neighbors, which can be susceptible to noise. To address this, I implemented a window-based approach. By defining a window_size, the code now checks for local minima within a specified range. Ensuring that the detected minima are less prone to minor fluctuations.

Spacing Between Detected Minima - To avoid detecting multiple minima that are very close to each other, I introduced a min_gap parameter. This ensures that once a minima is detected, the next one can only be detected if it's spaced out by at least min_gap. This helps in reducing false positives and ensures that only significant minima were detected.

Saving the Output - I added a line to save the final plot as an image. Ensuring that the results can be easily reviewed later without having to rerun the code.

(d) any observation about any output errors with your modified logic:

Upon analyzing the modified logic, I observed that the dynamic window and the minimum gap constraints improved the minima detection for the desired text, "In Congress July 4, 1976". However, there are a few points to consider:

Dynamic Window Size - The window size determines the vicinity in which the current value is checked to be a minimum. If the window size is too large, it might miss closely spaced minima. If it's too small, it might detect false minima in noisy regions.

Minimum Gap Between Minima - The min_gap ensures that detected minima are spaced apart. However, if set too high, it might miss closely spaced lines of text.

Static Threshold - The static threshold determines the minimum pixel count for a row to be considered a minima. If set too low, it might detect noise as minima, and if set too high, it might miss thin lines.

Improved Minima Detection - The window-based approach for detecting minima ensures that the detected points are more representative of actual minima in the data. This reduces the chances of false positives, especially in cases where there might be minor fluctuations in the data.

Potential for Parameter Tuning - While the introduced parameters (static_threshold, window_size, and min_gap) provide flexibility, they also introduce the need for tuning. Depending on the specific image or data, one might need to adjust these parameters for optimal results.

While the modified logic is a step in the right direction, it's would be essential to fine-tune the parameters based on the specific characteristics of the images being processed. I also found that it was not perfect and there were some excess minima found. In conclusion, the modifications made to the original code were aimed detecting the missing lines. While the new logic offers improved performance in this regard. It does have some room for improvement and potentially parameter adjustment to receive the best results for a given image.