Question 4

Write a program to implement change in dynamic range of an image from `[a, b]` to `[c, d]`. `a` and `c` are the minimum pixel values of input and output images respectively, while `b` and `d` are the maximum for the two. Comment on visual quality of the image after the operation.

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import math
```

Images to process

In [2]:

```
path_inp = '../../images/dat/'  # path for input files
path_out_orig = 'originals/'    # path for output files: originals
path_out_conv = 'converted/'    # path for output files: converted

filenames = [
    'l256',
    'o256',
    'p256',
    'z256'
]

ext_inp = '.dat'    # file extention for input
ext_out = '.bmp'    # file extention for output
```

Convert images to numpy array and store in a list of tuples as (filename, np.array)

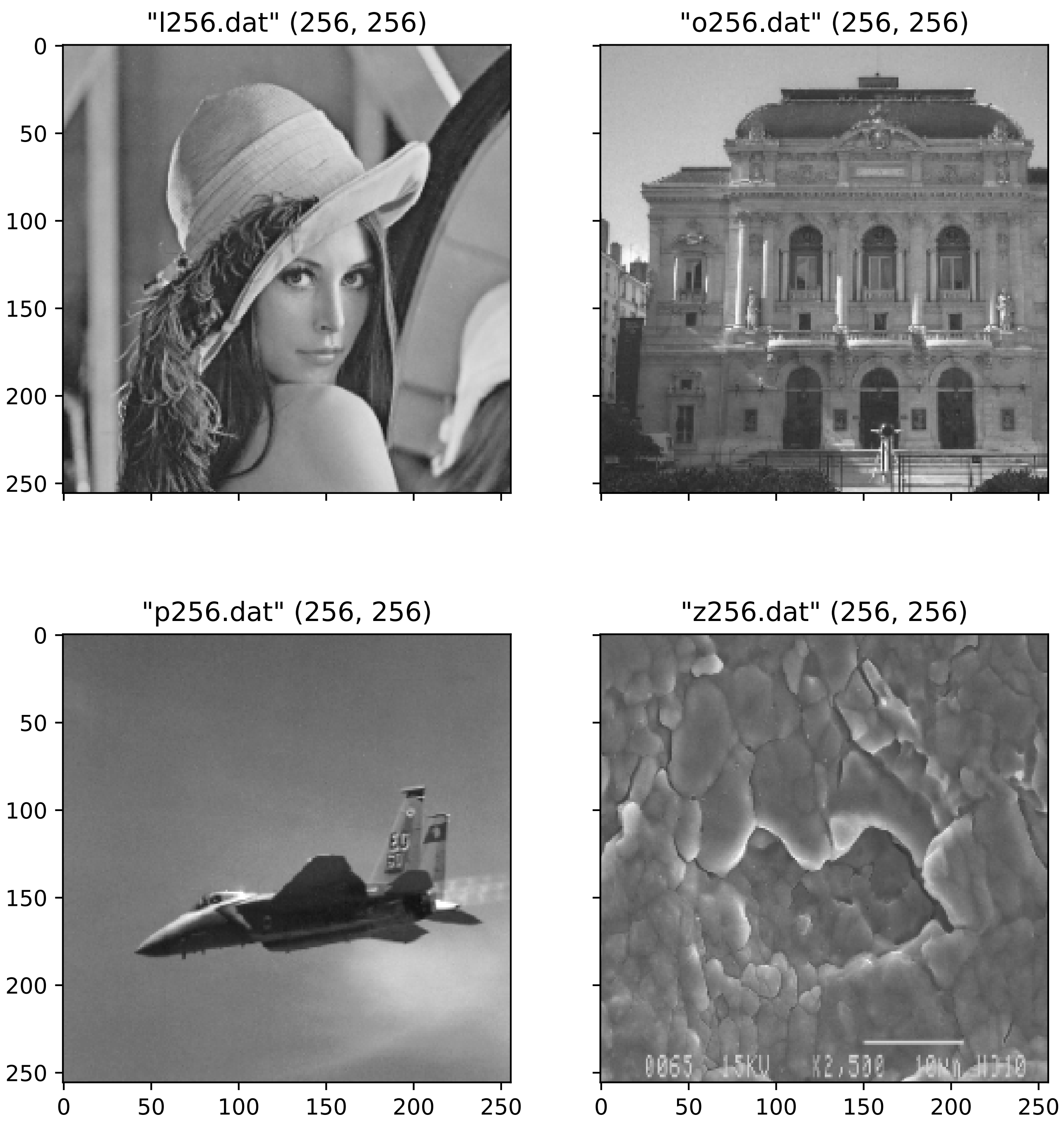
In [3]:

```
# Stores the list of dictionaries for the filename, original image, converted image/s
images = []

# Iterate for all filenames
for idx, filename in enumerate(filenames):
    # Store image pixels as uint8 2D array
    image = np.array(
        [i.strip().split() for i in open(path_inp + filename + ext_inp).readlines()],
        dtype='uint8'
    )

    # Add (filename, numpy array of image) into images list
    images.append({
        'filename': filename,
        'orig': image
    })

    # Save original image as .dat file
    np.savetxt(
        path_out_orig + ext_inp[1:] + '/' + filename + ext_inp,
        image,
        fmt=' %d',
        newline=' \n'
    )
```

Display input images

In [4]:

```
# Matrix dimensions
cols = 2
rows = -(-len(filenames) // cols)

# Create figure with rows × cols subplots
fig, axs = plt.subplots(rows, cols, dpi=80, sharex=True, sharey=True)
fig.set_size_inches(4 * cols, 4.5 * rows)

# Iterate for all images
for idx, image_dict in enumerate(images):
    filename = image_dict['filename']
    image = image_dict['orig']

    # Set subplot title as '"filename" (rows, cols)'
    axs[int(idx // cols), idx % cols].set_title('"{}" {}'.format(
        filename + ext_inp,
        image.shape
    ))
    # Add subplot to figure plot buffer
    axs[int(idx // cols), idx % cols].imshow(
        image,
        cmap='gray',
        vmin=0,
        vmax=255
    )

    # Save original image as .bmp file
    plt.imsave(
        path_out_orig + ext_out[1:] + '/' + filename + ext_out,
        image,
        cmap='gray',
        vmin=0,
        vmax=255
    )

# Hide x labels and tick labels for top plots and y ticks for right plots
for ax in axs.flat:
    ax.label_outer()

# Display the figure
plt.show()
```



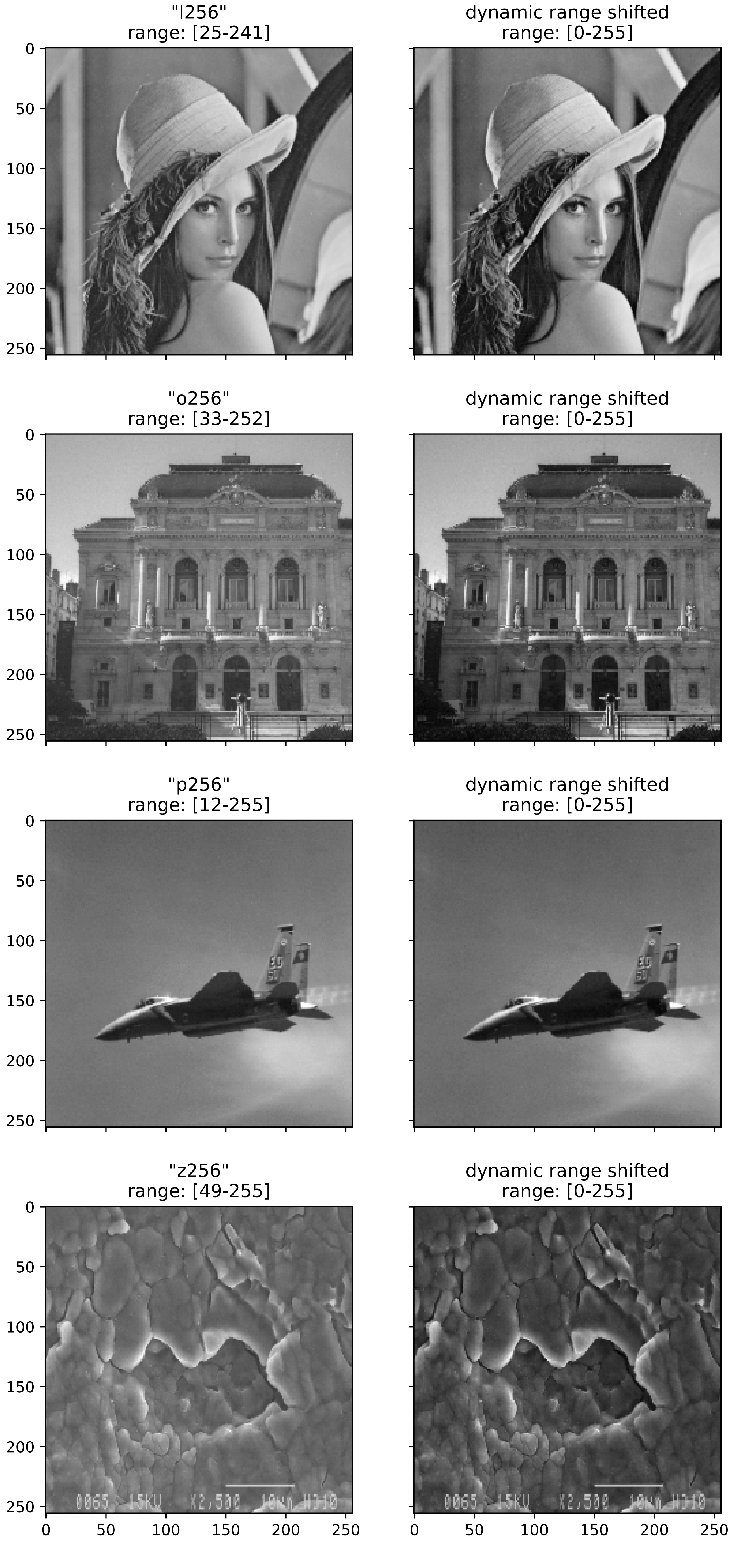
Dynamic Range Shift

In [5]:

```
def shift_dyn_range(image, range):
    min_pixel = min([min(i) for i in image])
    max_pixel = max([max(i) for i in image])

    min_range, max_range = range

    image = \
        min_range + \
        ((image - min_pixel) / (max_pixel - min_pixel)) * (max_range - min_range)

    return np.array(image, dtype='uint8')
```

In [6]:

```
rows, cols = len(images), 2

# Create figure with rows × cols subplots
fig, axs = plt.subplots(rows, cols, dpi=80, sharex=True, sharey=True)
fig.set_size_inches(4 * cols, 4.5 * rows)

# Iterate for all images
for idx, image_dict in enumerate(images):
    filename = image_dict['filename']
    orig = image_dict['orig']
    shift_img = shift_dyn_range(orig, [0, 255])

    min_orig = min([min(i) for i in orig])
    max_orig = max([max(i) for i in orig])

    axs[idx, 0].set_title('"{}"\nrange: [{}-{}]'.format(
        filename,
        min_orig,
        max_orig
    ))
    axs[idx, 0].imshow(orig, cmap='gray', vmin=0, vmax=255)
    min_shift = min([min(i) for i in shift_img])
    max_shift = max([max(i) for i in shift_img])

    axs[idx, 1].set_title('dynamic range shifted\nrange: [{}-{}]'.format(
        min_shift,
        max_shift
    ))
    axs[idx, 1].imshow(shift_img, cmap='gray', vmin=0, vmax=255)

    # Save threshold image as .bmp file
    plt.imsave(
        path_out_conv + ext_out[1:] + '/' + filename + '_dyn_shift' + ext_out,
        shift_img,
        cmap='gray',
        vmin=0,
        vmax=255
    )

    # Save pixel values of threshold image as a 2D matrix in a .dat file
    np.savetxt(
        path_out_conv + ext_inp[1:] + '/' + filename + '_dyn_shift' + ext_inp,
        shift_img,
        fmt=' %d',
        newline=' \n'
    )

# Hide x labels and tick labels for top plots and y ticks for right plots
for ax in axs.flat:
    ax.label_outer()

# Save and display the figure
plt.savefig('dynamic_range_shift_comp.jpg')
plt.show()
```



Resource

GitHub repository: `Image Processing and Pattern Recognition - Anindya Kundu (meganindya)`