Computational Physics III (PHYS-332) 2019–2020
O. Yazyev, K. Cernevics, Y. Guan, J. Colbois, J. Felisaz

Exercise sessions 4–5: Fast Fourier transform. Image filtering.

Exercise 1: Radix-2 algorithm [10 pts]

Files: `test.m`

The radix-2 recursive algorithm performs the discrete Fourier transform in an efficient way, hence belonging to the family of Fast Fourier transform (FFT) algorithms. The price paid for computational efficiency is a more sophisticated implementation involving recursion and workarounds for odd-sized arrays. The results of a discrete Fourier transform (DFT) and an FFT are **exactly** the same.

1. Write a function that implements the radix-2 variant of FFT algorithm as explained in the lecture. Consider only the simplest case of $N = 2^r$ sampling points, where r is a natural number.
2. Verify your FFT implementation by transforming simple model functions and comparing your result with MatLab's function `fft`. Take a look into the supplied test script `test.m` for some examples. Make sure that your implementation handles correctly every possible input array. Check whether your implementation of the FFT is indeed faster than a simpler straightforward implementation of DFT.

Listing 1: 'myfft.m'

```
function result=myfft(input_array)

% myfft computes discrete Fourier transform with a radix-2.
%
% Arguments:
%
%     input_array (1D array complex, size 2^N, N>0): data to
%     transform;
%
% Returns: 1D array complex, transformed data of the same shape as
% an input array.

% Following is a stub. This should be replaced with your own code.
result = fft(input_array);

end
```

Submit: file `myfft.m` with your implementation of the radix-2 FFT algorithm.

Exercise 2: The Berry phase [20 pts]

Files: `stm.png`

The Berry phase plays an important role in many important properties and phenomena observed in condensed matter systems, e.g. the quantum Hall effect. However, direct observations of this non-evident quantum mechanical effect are challenging. It was recently reported that the Berry phase of electrons in graphene can be detected by performing scanning tunneling microscopy (STM) measurements around a point defect (e.g. a missing atom in the lattice). The Berry phase can be detected by analyzing the spatial pattern of electron density obtained in such measurements (Fig. 1a). Observing dislocations (structures that can be viewed as abrupt terminations of one or more elements in a periodic arrangement) in such images indicates the presence of the Berry phase (Fig. 1b). However, this task can easily be accomplished only with the help of image filtering techniques. The goal of this exercise is to design and apply an appropriate filter to the image in `stm.png` that will help us revealing the Berry phase of electrons in graphene.

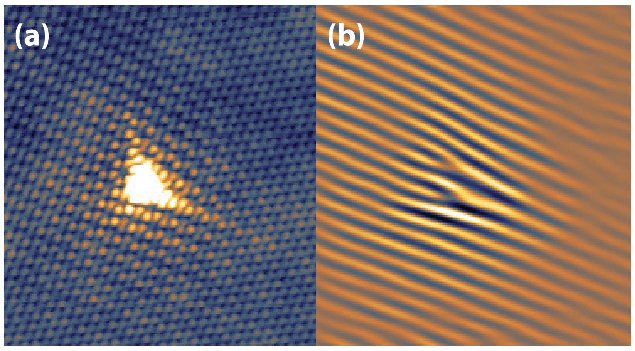

Figure 1: (a) STM image of a point defect in graphene. The underlying honeycomb lattice of carbon atoms can be seen, while the point defect produces a bright triangular feature. (b) Same image after Fourier filtering reveals the dislocation structure. Reproduced from C. Dutreix *et al.*, Nature **574**, 219 (2019).

1. Import the image from `stm.png` into Matlab. In order to do that:
   - load the image as an RGB three-channel array using `imread`;
   - convert the data to a grayscale array with `rgb2gray` and `double`;
   - plot the image using `imshow`.
2. Calculate the Fourier transform of the STM image using the `fft2` and `fftshift` functions of Matlab. Visualize the result using `imshow`. Make sure the resulting image clearly shows the hexagonal pattern formed by the ensemble of Bragg peaks.
   Hint: In order to see clearly the Bragg peaks in the Fourier transformed image, you will need to tune the contrast. To do so you can either apply a linear and/or logarithmic transformation to the pixel intensities.
3. Give a brief explanation to the result of Fourier transform. Specifically, comment on the order of the Bragg peaks and on the relation of the graphene lattice to the hexagonal pattern of the Fourier-transformed image.
4. Design a band-pass Fourier filter to distinguish the intensity component of interest with respect to other contributions.
   Hint: Only one pair of peaks is needed. The filter should retain only the first Bragg peaks and not the low-frequency part (center of the Fourier-transformed image). Figure 2 is an example of a color-coding filter.

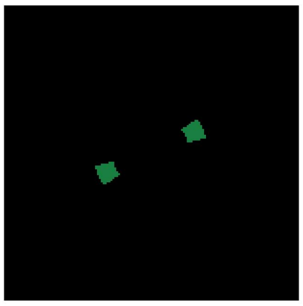

Figure 2: An example of a color-coding band-pass filter.

The following listing is a good starting point for your filter.

```
N = 525;

% Cartesian coordinates - x,y
[x,y] = meshgrid(1:N,1:N);

% Polar coordinate - phi
phi = atan2(y-N/2,x-N/2);

% Prepare colors in hue-saturation-value (HSV) model
hsv = zeros(N,N,3);
hsv(:,:,1) = mod(phi/2/pi,1); hsv(:,:,2) = 1; hsv(:,:,3) = 1;

% Convert to red-green-blue (RGB) model
rgb = hsv2rgb(hsv);

imshow(uint8(rgb*255));
```

5. Perform the inverse Fourier transform on the selected pair of Bragg peaks and plot the final image. The results should resemble Fig. 1b.

6. Use the Fourier-filtered image that you obtained to find the index of dislocation. When calculating the dislocation index consider only the central part of the image that corresponds to the bright spot of the original image.

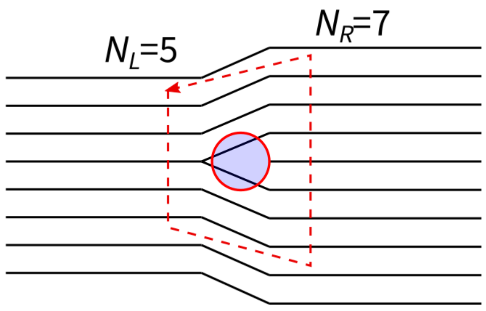

Figure 3: Evaluation of the index of a dislocation $N = |N_L - N_R| = 2$

Submit: your report as a PDF file as well as your scripts. The points are distributed in the following way:

- [12 pts] Filtered image with dislocations, the filter and the Fourier space images (before and after filtering);
- [4 pts] answer to the question of point 3;
- [4 pts] index of the dislocation;