Course reader: *2D Fourier transform*

- The 2D FFT is an extension of the 1D FFT that you've been learning about so far in this course. The 2D FFT is used for images and pictures.
- The digital representation of a picture is either as a 2D or as a 3D matrix. The two dimensions are the width and height of the image, and the third dimension would have three elements for the red, green, and blue channels.
- To compute the 2D FFT, first compute the normal FFT on each column of the matrix, then compute the FFT on each row of that result. This produces a 2D matrix of Fourier coefficients, from which you can extract amplitude and phase just like with the 1D FFT.
- Natively, the low frequencies are at the edges of the matrix and the high frequencies are in the center of the matrix. However, most people swap the quadrants in order to make the low frequencies in the center and the high frequencies be at the edges.
- Interpreting a 2D FFT is more difficult than interpreting the 1D FFT. That's why I recommend starting simple and trying to understand the effects of sine gradients on the resulting amplitude and phase maps.
- I hope the videos that are made in MATLAB and python are helpful for you to build some intuition about interpreting the 2D fft.

Exercises

Note about the exercises: I give my answers in MATLAB code but you should feel free to use whatever program (Python, C++, Julia, html5) you feel most comfortable with.

1. The two MATLAB examples of side gradients used square-shaped gradients. Adjust the code to make the sine gradient a circle instead of a square. Does that impact the resulting amplitude spectrum?

2. I showed an example in MATLAB of a solid ball moving around on a plane. The amplitude spectrum was a dot in the center. Integrate the sine gradient and ball examples by replacing the ball with moving gradients. Then try moving the ball around in different regions of the plane to see the effects on the amplitude spectrum.

Answers

1. There are two modifications to the code for the sine gradient videos.

1. Before the loop over phases or frequencies:

```
width  = 100;
gaus2d = exp(-(x.^2 + y.^2) ./ (2*width^2));
```

2. And then inside the loop in the line after variable `img` is defined but before its 2D FFT:

```
img(gaus2d<.7) = 0;
```

You can leave the width as-is and try changing the threshold, e.g., from .7 to .5 to .99.

2. Starting from the code to generate the moving ball, add the following code:

1. Before the loop over locations:

```
sinefreq = .1;
sinegrad = sin( 2*pi*sinefreq*x );
```

2. And then inside the loop, replace the two lines after `gaus2d = ...` with

```
img = sinegrad;
img(gaus2d<.9) = 0;
```

3. You might also want to set the color limit for better interpretation. In the code that sets up subplot(121):

```
set(gca,'clim',[-1 1])
```

It's interesting to compare different spatial frequencies. For example, try .01 or .05. For .01, the amplitude at low frequencies (center of the amplitude spectrum plane) goes from red to blue (decreased amplitude). What's happening in the image that causes this decrease in amplitude?