question 2 views

Daily Challenge 25.5

(**Due: Thursday 3/28 at 12:00 noon Eastern**)

Today you will implement the discrete Fourier transform in Python.

Up to an overall constant, and the choice of whether you put the 2π in the exponent, the Fourier transform of a function $f(x)$ is roughly

$$\tilde{f}(k) = \int_{-\infty}^{\infty} e^{-ikx} f(x)\, dx.$$

For realistic situations, we assume $f(x)$ is some finite signal or wave train which is *supported* on an interval $[-S, S]$. That is, we assume that $f(x) = 0$ if $|x| > S$ for some big number S called the support. Then we can cut off the integral as

$$\tilde{f}(k) = \int_{-S}^{S} e^{-ikx} f(x)\, dx.$$

To put this on a computer, we need to replace the integral with a Riemann sum. Let's chop up the interval $[-S, S]$ into N points x_j, using a partition

$$x_0 = -S \leq x_1 \leq \cdots \leq x_N = S,$$

and assume for simplicity that the spacing Δx between adjacent points is constant. Then

$$\tilde{f}(k) \approx \sum_{j=0}^{N-1} e^{-ikx_j} f(x_j)\, \Delta x_j.$$

This is now in a form ready to be implemented in Python.

(**Part a**). Write a function `fourier_transform(f, S=100)` that takes in a Python function and returns another function which gives the Fourier transform of the original function, using the conventions above.

That is, complete the following code skeleton:

```
def fourier_transform(f, S=100):
    """
    Computes the Fourier transform of f
    Inputs:
       f: a Python function mapping floats to floats
       S: the support such that f(x) = 0 for |x| > S
    Outputs:
       a Python function mapping floats to floats, which is the Fourier transform of f
    """

    ## Chop up the interval [-S, S] into N pieces

    def f_tilde(k):
       ## Implement the sum described above
       return output

    return f_tilde
```

I should emphasize that you are writing a function which takes in a function and returns another function.

Hint: you can use `np.exp` with a complex argument. Python knows about Euler's formula; to enter the imaginary unit i, you type `1j` in Python. For example, the statement $e^{\pi i} = -1$ in Python is

```
import numpy as np

print(round(np.exp(np.pi*1j)))
```

```
(-1+0j)
```

(**Part b**) Define a function which gives a finite cosine wave train. That is, implement a Python function that returns

$$f(x) = \begin{cases} \cos(x) & \text{if } -100 \leq x \leq 100 \\ 0 & \text{otherwise} \end{cases}.$$

Compute the Fourier transform $\tilde{f}(k)$. Note that the Fourier transform you get will be purely real, rather than complex, since the input signal is even.

Plot it, by which I mean plot the real part of $\tilde{f}(k)$ using matplotlib. Does it look like what you expect from the last meeting, namely a sharp peak at a particular frequency and then some oscillatory fall-off?

daily_challenge

Updated 17 days ago by Christian Ferko

the students' answer, *where students collectively construct a single answer*

soon

Updated 17 days ago by Logan Pachulski

the instructors' answer, *where instructors collectively construct a single answer*

Click to start off the wiki answer

followup discussions *for lingering questions and comments*