18.330 Pset 8

Kathleen Brandes

April 28, 2019

1 Problem 1

1.1 Part A

Since the function g_n is even and real, then the fourier transform will also be even and real. Therefore, we can get the transform for the section of $n = 0, \ldots, N$ with $g_n = f_n$ has DFT coefficients of $\hat{g}^1_\nu = \hat{f}_\nu = \frac{1}{2N}\sum_{n=0}^{N-1} f_n e^{-2\pi i n\nu/2N}$ for $2N$ samples. And similarly, for $k = 0, \ldots, N$ with $g_{N+k} = f_{N-1-k}$, the DFT coefficients will be $\hat{g}^2_\nu = \hat{f}_\nu = \frac{1}{2N}\sum_{n=0}^{N-1} f_{N-1-n} e^{-2\pi i n\nu/2N}$. Then, to get the overall coefficients of function g, we have to sum the two:

$$\begin{gathered}\hat{g}_\nu = \hat{g}^1_\nu + \hat{g}^2_\nu \\ = \tfrac{1}{2N}[\textstyle\sum_{n=0}^{N-1} f_{N-1-n} e^{-2\pi i n\nu/2N} + \sum_{n=0}^{N-1} f_n e^{-2\pi i n\nu/2N}] \\ = \tfrac{1}{2N}\textstyle\sum_{n=0}^{N-1}(f_n + f_{N-1-n}) e^{-\pi i n\nu/N} \\ = \tfrac{1}{2N}\textstyle\sum_{n=0}^{N-1}(f_n + f_{N-1-n}) cos(\pi n\nu/N)\end{gathered}$$

We can reduce the exponential down to an expression in only cosines since the function g is even and real. The DCT has $N/2$ independent coefficients $\hat{g}_\nu$ since the function is even and the summand $(f_n + f_{N-1-n}$ will partition the total number of values in half since it can be the same for high values of n as it is for a low value of n and the periodicity of cosine allows this to hold.

1.2 Part B

To derive the inverse DCT:

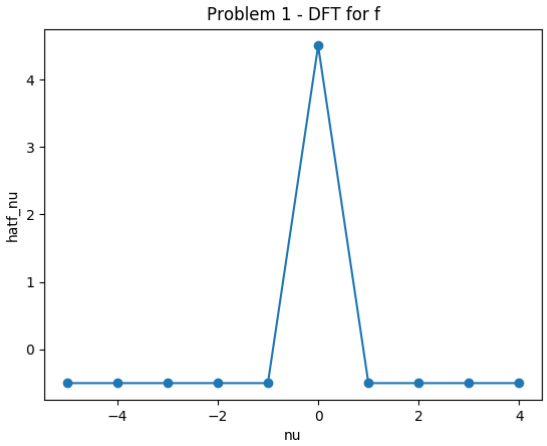
$$\begin{gathered}g_n = \tfrac{1}{N}\textstyle\sum_{k=0}^{N-1} \hat{g}_\nu e^{2\pi i n k/N} \\ = \tfrac{1}{N}\textstyle\sum_{k=0}^{N-1} \tfrac{1}{2N}\sum_{j=0}^{N-1}(f_j + f_{N-1-j}) cos(\pi j k/N) e^{2\pi i n k/N} \\ = \tfrac{1}{2N^2}\textstyle\sum_{k=0}^{N-1}\sum_{j=0}^{N-1}(f_j + f_{N-1-j}) cos(2\pi n(k-j)/N)\end{gathered}$$

This summed cosine $\sum_{j=0}^{N-1}(f_j + f_{N-1-j}) cos(2\pi n(k-j)/N)$ will evaluate to $N(f_j + f_{N-1-j})$ if $k = j$ since $cos(0) = 1$ and will be 0 otherwise due the periodicity. Therefore, we can get the expression for IDCT:

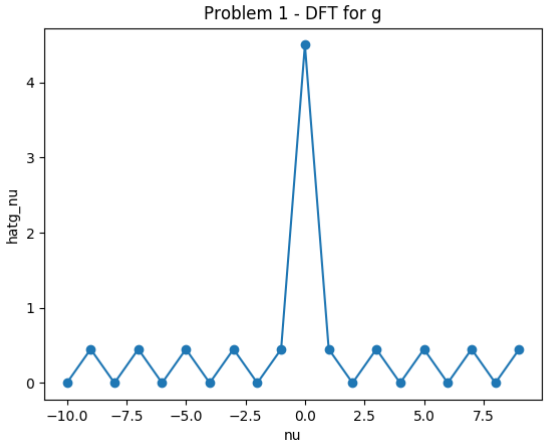
$$g_n = \tfrac{1}{2N}\textstyle\sum_{k=0}^{N-1}(f_k + f_{N-1-k}) cos(2\pi n k/N)$$

1.3 Part C

My implementation of the DCT formula is attached on steller in problem1.jl. The results of function can mimic the plots on the pset for the test function $f_n = n$ for $n = 0, \ldots, 9$. Here is the result for the DCT coefficients of regular $\hat{f}_\nu$:



And here is the result for the DCT coefficients $\hat{g}_\nu$ for the function g_n as described:



1.4 Part D

The discrete cosine transformation is incredibly useful for lossy compressions in spectral space because it eliminates small, high frequency components of the original signal. It identifies and eliminates high frequency boundary conditions, and in turn helps prevent Gibbs phenomenon which could occur if you tried to

just improve DFT with many more samples. This is useful for lossy compressions since these high frequency edge cases are not specific to the image and are mostly noise, so therefore they should be removed to help compress the image.

2 Problem 2

2.1 Part A

For function $f(x,y,z) = e^{-\sigma_x x^2/2 - \sigma_y y^2/2 - \sigma_z z^2/2}$, the multidimensional Fourier transform will be:

$$F(\vec{k}) = \frac{1}{(2\pi)^3}\int_{-\infty}^{\infty} e^{-\sigma_x x^2/2 - \sigma_y y^2/2 - \sigma_z z^2/2} e^{-\vec{k}\cdot\vec{x}} d\vec{x}$$
$$= \frac{1}{(2\pi)^3}\int_{-\infty}^{\infty} e^{-\sigma_x x^2/2 - ik_1 x} dx \int_{-\infty}^{\infty} e^{-\sigma_y y^2/2 - ik_2 y} dy \int_{-\infty}^{\infty} e^{-\sigma_z z^2/2 - ik_3 z} dz$$

Since these three integrals are independent in their use of variables, I will solve only one explicitly:

$$\int_{-\infty}^{\infty} e^{-\sigma_x x^2/2 - ik_1 x} dx$$
$$= \int_{-\infty}^{\infty} e^{\frac{-\sigma_x}{2}[x^2 + 2ik_1 x/\sigma_x]} dx$$
$$= \int_{-\infty}^{\infty} e^{\frac{-\sigma_x}{2}[x^2 - 2(-ik_1/\sigma_x)x + (-ik_1/\sigma_x)^2 - (-ik_1/\sigma_x)^2]} dx$$
$$= e^{\frac{-k_1^2}{2\sigma_x}} \frac{\sqrt{pi}}{\sqrt{\frac{\sigma_x}{2}}}$$
$$= \frac{\sqrt{2\pi}}{\sqrt{\sigma_x}} e^{\frac{-k_1^2}{2\sigma_x}}$$

Therefore, we can get the result of the entire fourier transform using this result:

$$F(\vec{k}) = \frac{1}{(2\pi)^3}\int_{-\infty}^{\infty} e^{-\sigma_x x^2/2 - ik_1 x} dx \int_{-\infty}^{\infty} e^{-\sigma_y y^2/2 - ik_2 y} dy \int_{-\infty}^{\infty} e^{-\sigma_z z^2/2 - ik_3 z} dz$$
$$= \frac{1}{(2\pi)^3} \frac{\sqrt{2\pi}}{\sqrt{\sigma_x}} e^{\frac{-k_1^2}{2\sigma_x}} \frac{\sqrt{2\pi}}{\sqrt{\sigma_y}} e^{\frac{-k_2^2}{2\sigma_y}} \frac{\sqrt{2\pi}}{\sqrt{\sigma_z}} e^{\frac{-k_3^2}{2\sigma_z}}$$
$$= \frac{1}{(2\pi)^{3/2}\sqrt{\sigma_x}\sqrt{\sigma_y}\sqrt{\sigma_z}} e^{\frac{-k_1^2}{2\sigma_x} + \frac{-k_2^2}{2\sigma_y} + \frac{-k_3^2}{2\sigma_z}}$$

2.2 Part B

For function $f(x,y,z) = \frac{e^{-\kappa||r||}}{||r||}$ where $||r|| = \sqrt{x^2+y^2+z^2}, \kappa > 0$, the n-D fourier transform will be:

$$F(\vec{k}) = \frac{1}{(2\pi)^3}\int e^{-i\vec{k}\cdot\vec{x}} \frac{e^{-\kappa||r||}}{||r||} d\vec{x}$$

The transform this to spherical coordinates, so $\vec{k}\cdot\vec{x} = |k|rcos(\theta)$ and $d\vec{x} = r^2 sin(\theta) dr d\theta d\phi$ to get:

$$F(\vec{k}) = \frac{1}{(2\pi)^3}\int e^{-ircos(\theta)} \frac{e^{-\kappa r}}{r} r^2 sin(\theta) dr d\theta d\phi$$
$$= \frac{1}{(2\pi)^3}\int_0^{2\pi} d\phi \int_0^{\pi} d\theta \int_0^{\infty} dr \frac{e^{-ircos(\theta)-\kappa r}}{r} r^2 sin(\theta)$$
$$= \frac{1}{(2\pi)^2}\int_0^{\infty} dr \int_0^{\pi} d\theta r \frac{d}{d\theta}\left[\frac{e^{-ikrcos(\theta)-\kappa r}}{ikr}\right]$$

$$= \frac{1}{(2\pi)^2}\int_0^\infty drr\frac{e^{ikr-\kappa r}-e^{-ikr-\kappa r}}{ikr}$$
$$= \frac{1}{(2\pi)^2}\int_0^\infty dre^{-\kappa r}\frac{e^{ikr}-e^{-ikr}}{ik}$$
$$= \frac{1}{(2\pi)^2}\int_0^\infty dre^{-\kappa r}\frac{2sin(kr)}{k}$$
$$= \frac{1}{(k\pi)^2}[-e^{-\kappa r}\frac{\kappa sin(kr)+kcos(kr)}{k^2+\kappa^2}]|_0^\infty$$
$$= \frac{1}{(k\pi)^2}[-1\frac{\kappa sin(0)+kcos(0)}{k^2+\kappa^2}-0]$$
$$= \frac{1}{(k\pi)^2}[-1\frac{k}{k^2+\kappa^2}]$$
$$= \frac{-1}{(\pi)^2(k^2+\kappa^2)}$$

2.3 Part C

For function $f(\vec{x}) = e^{-x^TAx}$, where A is a symmetric, positive definite matrix, the n-D fourier transform will be:

$$F(\vec{k}) = \frac{1}{(2\pi)^n}\int_{R^n} e^{-i\vec{k}\cdot\vec{x}}e^{-x^TAx}d\vec{x}$$

I will diagonalize A in an orthonormal basis (using Gram Schmidt procedure) such that $Q^TAQ = D$, which can also be expressed as $A = Q^TDQ$ because Q is an orthogonal matrix (so $Q^T = Q^-1$) and D is diagonal matrix with eigenvalues $\lambda_i > 0$ since A is positive definite. Additionally, we can rewrite $\vec{k}\cdot\vec{x} = Q\vec{k}\cdot Q\vec{x}$ since it is orthonormal. And then, we can rewrite the transform: $F(\vec{k}) = \frac{1}{(2\pi)^n}\int_{R^n} e^{-i\vec{k}\cdot\vec{x}}e^{-x^TAx}d\vec{x} = \frac{1}{(2\pi)^n}\int_{R^n} e^{Q\vec{x}DQ\vec{x}-iQ\vec{k}\cdot Q\vec{x}}|det(Q)|d\vec{x}$. Then, doing a u-substitution with $\vec{u} = Q\vec{x}$, we get:

$$F(\vec{k}) = \frac{1}{(2\pi)^n}\int_{R^n} e^{\vec{u}D\vec{u}-iQ\vec{k}\cdot\vec{u}}|det(Q)|d\vec{u}$$
$$F(Q\vec{k}) = \frac{1}{(2\pi)^n}\int_{R^n}\prod_{i=1}^n e^{\lambda_i u_i^2-ik_iu_i}1 * d\vec{u}$$
$$= \prod_{i=1}^n \frac{1}{2\pi}\int_R e^{\lambda_i u_i^2-ik_iu_i}du_i$$

The result is that we get a product of a bunch of fourier transforms for Gaussians:

$$\prod_{i=1}^n \frac{1}{2\pi}\int_R e^{\lambda_i u_i^2-ik_iu_i}du_i = \prod_{i=1}^n \frac{1}{2\pi}\int_R e^{2\pi\lambda_i u_i^2-2\pi ik_iu_i}du_i$$
$$= \prod_{i=1}^n \sqrt{\frac{\pi}{-2\pi\lambda_i}}e^{\frac{-\pi^2k^2}{-2\pi\lambda_i}}$$

3 Problem 3

3.1 Part A

For the system of equations of the circulant matrix:

$$\begin{bmatrix} c_0 & c_{n-1} & \dots & c_1 \\ c_1 & c_0 & \dots & c_2 \\ \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & c_{n-2} & \dots & c_0 \end{bmatrix}\begin{bmatrix} x_0 \\ \vdots \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ \vdots \\ b_{n-1} \end{bmatrix}$$

We get a system of equations, for example $c_0x_0 + c_{n-1}x_1 + \dots + c_1x_{n-1} = b_0$. This can be written as a convolution of the form: $b_k = \sum_{j=0}^{n-1} c_jx_{k-j}$.

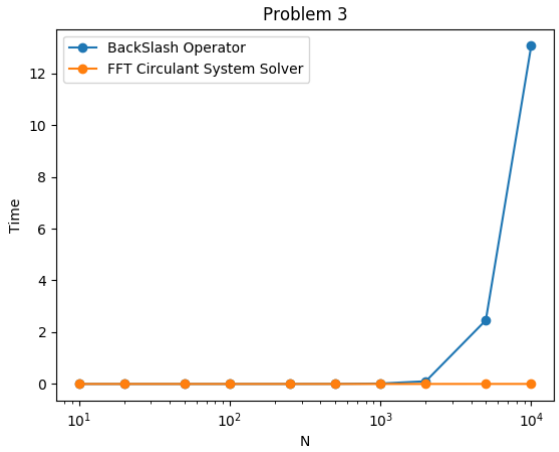
3.2 Part B

We can write the system as $x = C^{-1}b$, and then we can see that you can write $C = F^{-1}DF$ where D is a diagonal matrix of the eigenvalues since this is the diagonalization of C since F, the DFT coefficient's matrix, is a unitary operation. Therefore, to get an expression for x easily solvable using FFT, we can take the inverse(which, for a diagonal matrix, is an efficient operation requiring only inverting the values along the diagonal) and multiply by b on the right. We get $x = FD^{-1}F^{-1}b$, which can be solved using fast Fourier transforms by: `fft( inv(diag(`λ_i`)) ifft(b) )` where λ_i represent the eigenvalues of C.

3.3 Part C

My algorithm for solving the system $C\mathbf{x} = \mathbf{b}$ is on stellar as problem3.jl. First, I tested it on a problem of size $n = 4$. The FFT operation produced: $[0.301043 + 0.0im, 0.217268 + 0.0im, 1.27749 + 0.0im, 0.658571 + 0.0im]$. As expected, these are identical results to that of the backslash operation to solve $C\mathbf{x} = \mathbf{b}$ for random b, c.The backslash operation produced: $[0.301043, 0.217268, 1.27749, 0.658571]$.

I compared the solve-times for the FFT-based solver and the backslash operator in julia when evaluating systems with $n \in [10, 10^4]$:



The FFT-based solver was significantly faster than the backslash operator as n increased, particularly for $n > 10^3$.

3.4 Part D

Considering $n = 10^7$ for a circulant system, the backslash operator completely fails, throwing an OutOfMemory exception, because it does not have enough memory to explicitly store the entire matrix c since it will be 10^7x10^7. In contrast, the FFT-based algorithm is still capable to solve a system of equations

of this size since it is much more efficient in its operations on the vector and requires much less physical information to be stored.

4 Problem 4

4.1 Part A

For v_k be the k-th eigenvector, eigenvalue pair for a matrix C, then we know $Cv_k = \lambda_k v_k$. Let $b_l = (Cv_k)_l$ be the l-th element of the vector from this product. Then we know using the convolution formula that $b_l = \sum_{j=0}^{n-1} c_j(\zeta_k)^{l-j} = e^{l\frac{2\pi ik}{n}} \sum_{j=0}^{n-1} c_j(\zeta_k)^{-j} = \zeta_k^l \sum_{j=0}^{n-1} c_j(\zeta_k)^{-j} = \lambda_k \zeta_k^l$. Therefore we can see that $v_k = (1, \zeta_k^1, ..., \zeta_k^{n-1})^T$ is an eigenvector for matrix C and will have eigenvalue $\lambda_k = \sum_{j=0}^{n-1} c_j e^{\frac{-2\pi ijk}{n}}$.

4.2 Part B

We know that $CV = V\texttt{diag}(\lambda)$ since this is the matrix version of the standard eigenpair relationship $Cv = \lambda_i v$, so then we can get $C = V\texttt{diag}(\lambda)V^{-1}$. This is identical to the matrix expression for C for part 3a where the eigenvector matrix V is identical to the DFT coefficient matrix.

4.3 Part C

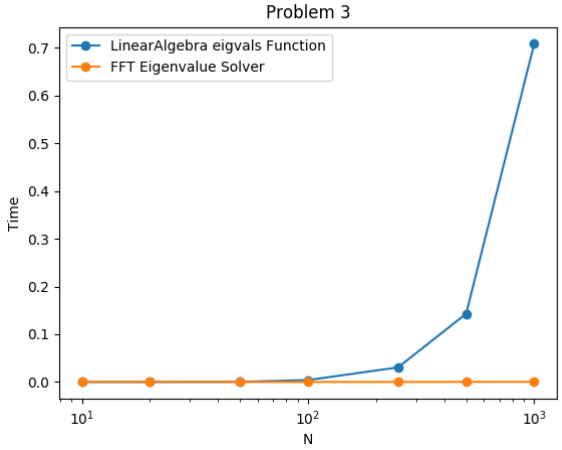
I found that the eigenvalues will be $\lambda_k = \sum_{j=0}^{n-1} c_j e^{\frac{-2\pi ijk}{n}}$. This is exactly the DFT of the vector representing the entries of the circulant matrix, c, so $\lambda_k = F_k$ so we can use the FFT to determine the eigenvalues using an algorithm:

```
c← entries of a column of C
λ = fft(c)
```

where λ will be an nx1 vector consisting of the eigenvalues of matrix C if C is a circulant matrix. This algorithm is also implemented in julia, attached on stellar under the title problem4.jl.

4.4 Part D

Results for finding eigenvalues of a random vector of length $n = 3$:
From the LinearAlgebra package's eigvals: [1.7152e19+0.0im, 1.23265e18+5.3879e18im, 1.23265e18-5.3879e18im]
From solving using FFT: [1.7152e19+0.0im, 1.23265e18-5.3879e18im, 1.23265e18+5.3879e18im]
The two methods produce identical results for the eigenvalues of the random vector. Analyzing the solve times of the two methods for vectors of length $n \in [10, 1000]$:



The FFT method of obtaining eigenvalues is significantly faster than the LinearAlgebra package's QR decomposition method for increasing values of n.