(1) Solve the Poisson's equation

$$\Delta u = f$$

on the square $(x, y) : 0 \le x, y \le 1$ with homogeneous Dirichlet boundary conditions. Assume that the function f is well approximated by

$$f(x, y) = \sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_{mn} \sin(m\pi x)\sin(n\pi x).$$

Choose an appropriate discretization for $f(x, y)$ and organize your computations to use the Fast Fourier Transform (FFT). Verify your results on several examples.

First, let's assume that we can approximate u by

$$u(x, y) = \sum_{m=1}^{N} \sum_{n=1}^{N} \beta_{mn} \sin(m\pi x)\sin(n\pi x).$$

Then,

$$\Delta u = \sum_{m=1}^{N} \sum_{n=1}^{N} -\pi^2(m^2 + n^2)\beta_{mn} \sin(m\pi x)\sin(n\pi x).$$

Plugging this approximation for the Laplacian into our PDE yields

$$\sum_{m=1}^{N} \sum_{n=1}^{N} -\pi^2(m^2 + n^2)\beta_{mn} \sin(m\pi x)\sin(n\pi x) = \sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_{mn} \sin(m\pi x)\sin(n\pi x)$$

which implies we can compute β_{mn} as

$$\beta_{mn} = -\frac{\alpha_{mn}}{\pi^2(m^2 + n^2)}$$

where α_{mn} can be found using the Discrete Fourier Sine Transform (DST) which relies on the FFT. Then, we can find β_{mn} using the found α_{mn} and then apply the inverse DST to recover our solution to high orders of accuracy.

My code for the above procedure can be found at the end of the document. My code uses FFTW to perform the DST and inverse DST of which I had to implement a normalization when applying the inverse.

A visual comparison between my code and Mathematica's numerical solution can be found on the next page.

(a) $f(x,y) = -xy(x-1)(y-1)$

(b) $f(x,y) = \sin(6\pi x)\sin(2\pi y)$

(c) $f(x,y) = 10000xy(x-1)(y-1) \cdot J_0\left(20\left(x^2+y^2\right)\right)$
where J_0 is the Bessel function of the first kind of order zero.

(d) $10^4 \, \triangle(6\max(|\sqrt{x}-0.5|\,,|y^2-0.5|))$
where $\triangle(x)$ is the triangle wave of period 1.

(2) Let $\partial\Omega$ be the ellipse $x^2/a^2 + y^2/b^2 = 1$. Consider the boundary value problem

$$\begin{cases} \Delta u = 1, & (x,y) \in \Omega \\ u = x^4 + y^4, & (x,y) \in \partial\Omega \end{cases}.$$

(a) Reduce the problem to that with homogeneous boundary conditions.

To impose homogeneous boundary conditions on our problem, let's make the change of variables

$$u(x,y) = v(x,y) + (x^4 + y^4)$$

with the constraint that $v(x,y) = 0$ for $(x,y) \in \partial\Omega$. Then, on the boundary $\partial\Omega$, we have

$$u|_{(x,y)\in\partial\Omega} = v|_{(x,y)\in\partial\Omega} + x^4 + y^4 = x^4 + y^4$$

which implies

$$v|_{(x,y)\in\partial\Omega} = 0.$$

Furthermore, our PDE becomes

$$\Delta u = \Delta v + 12(x^2 + y^2) = 1$$

or in other words,

$$\Delta v = 1 - 12(x^2 + y^2).$$

So we can repose our problem as

$$\boxed{\begin{cases} \Delta v = 1 - 12(x^2 + y^2), & (x,y) \in \Omega \\ v = 0, & (x,y) \in \partial\Omega \end{cases}}$$

with the original solution defined as $u(x,y) = v + (x^4 + y^4)$.

(b) Reduce the problem to the Dirichlet problem for the Laplace equation.

Our goal for this problem is to make a change of variables such that the Laplacian of our change of variables will kill the forcing term on the RHS of our PDE. There are many possibilities we could choose but I will pick the semi-symmetric change of variables

$$u = v + \frac{x^2 + y^2}{4}.$$

In this case, our PDE becomes

$$\Delta u = \Delta v + 1 = 1$$

which implies

$$\Delta = 0.$$

Furthermore, for $(x,y) \in \partial\Omega$, we have

$$u = v + \frac{x^2 + y^2}{4} = x^4 + y^4$$

which implies

$$v = x^4 - \frac{x^2 + y^2}{4} + y^4$$

on the boundary. So, we can repose our BVP as

$$\boxed{\begin{cases} \Delta v = 0, & (x,y) \in \Omega \\ v = x^4 - \dfrac{x^2 + y^2}{4} + y^4, & (x,y) \in \partial\Omega \end{cases}}$$

with the original solution defined as $u(x,y) = v + \frac{x^2+y^2}{4}$.

Code Used

Note: some of the symbols are missing in my code snippet because LaTeX does not support all unicode characters.

```julia
#=
# FFT Based Solver for Poisson's equation
#   Δu = f
#
# Author: Caleb Jacobs
# DLM: 17-04-2022
=#

using FFTW
using Plots


"""
    getGrid(Nx, Ny)

Generate 2D node grid with `Nx` x-coordinates and `Ny` y-coordinates.
"""
function getGrid(Nx, Ny)
    x = range(0, 1, length = Nx)
    y = range(0, 1, length = Ny)

    X = repeat([x...]', Ny, 1)
    Y = repeat(y, 1, Nx)

    return (x, y, X, Y)
end

"""
    poissonSolve(f, Nx, Ny)

Solve Poissons equation over the unit square with homogeneous
boundary Dirichlet boundary condtions and forcing term `f`(x,y).
"""
function poissonSolve(f, Nx, Ny)
    (x, y, X, Y) = getGrid(Nx, Ny)  # Get computation nodes

    fx = map(f, X, Y)               # Function values at each node

    b = FFTW.r2r(fx, FFTW.RODFT00)  # Perform FFT based sine transform

    a = [-b[j, i] /  (^2 * (i^2 + j^2))
        for j   1:Ny, i   1:Nx]      # Compute FFT of solution

    U = FFTW.r2r(a, FFTW.RODFT00) /
        (4 * (Nx + 1) * (Ny + 1))   # Get solution via inverse sine FFT

    display(contour(x, y, U, st = :contour,
                         fill = true,
                         aspect_ratio = :equal))

    return U
end
```