Задание

1. Выбрать входную функцию и число , исходя из варианта. Построить график . Здесь и далее для каждого графика следует строить отдельно графики/изображения амплитуды и фазы. Входную область ограничить радиусом = 5.
2. Восстановить изображение в двумерный массив и построить это изображение.
3. Реализовать преобразование Ханкеля методом численного интегрирования (например, методом левых прямоугольников). Размеры входной и выходной областей должны совпадать. Применить преобразование ко входной функции и получить выходную . Построить её график, а также восстановить двумерную функцию и построить её изображение.
4. Реализовать двумерное преобразование Фурье через БПФ. Применить его ко входной двумерной функции . Построить изображение выходной функции, сравнить его с результатом, полученным для преобразования Ханкеля. Если изображения амплитуд сильно отличаются, попытаться увеличить число точек дискретизации.
5. Исследовать скорость выполнения двумерного БПФ и преобразования Ханкеля, варьируя число точек дискретизации. Сделать выводы.

Таблица 1. Варианты выполнения задания.

|  |  |  |  |
| --- | --- | --- | --- |
| Вариант № | Название функции | Входная функция | Число |
| 19 | Тонкое кольцо |  |  |

Результаты выполнения

Исходный код программы приведён в приложении А.

На рисунке 1 представлены графики амплитуды и фазы входной функции.

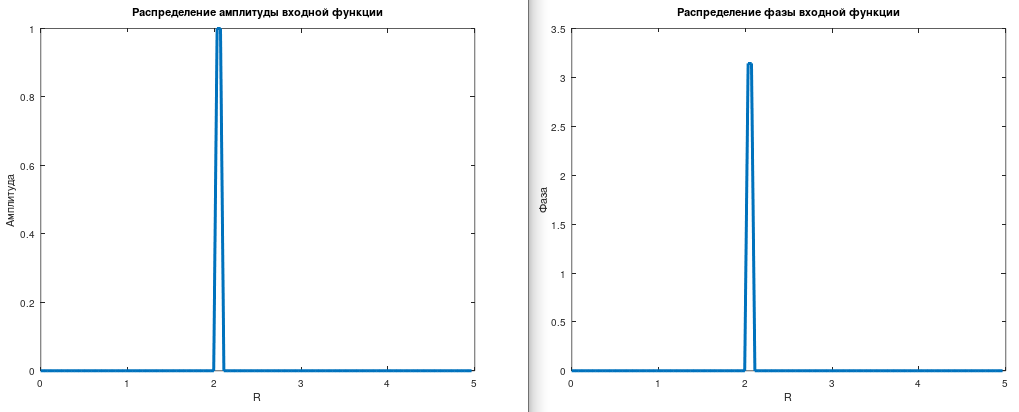


Рисунок 1 – Графики амплитуды и фазы входной функции

Восстановленное изображение входной функции представлено на рисунке 2.

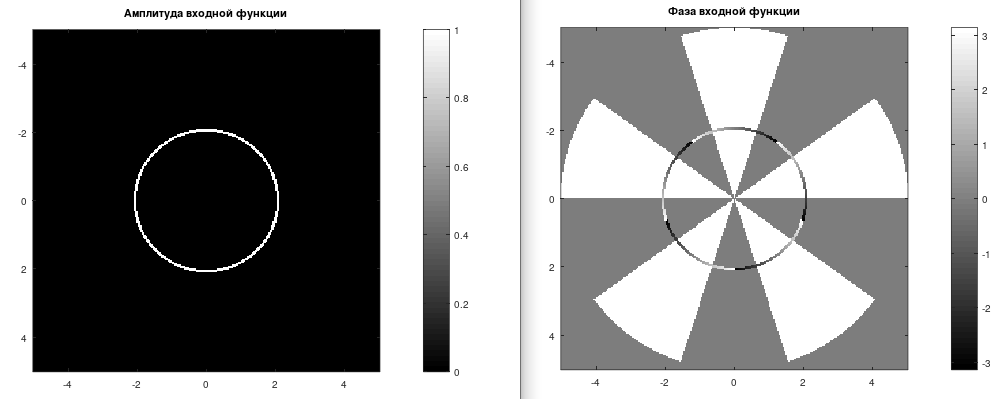


Рисунок 2 – Графики восстановленного изображения

На рисунках 3, 4 представлены изображения после применения преобразования Ханкеля.

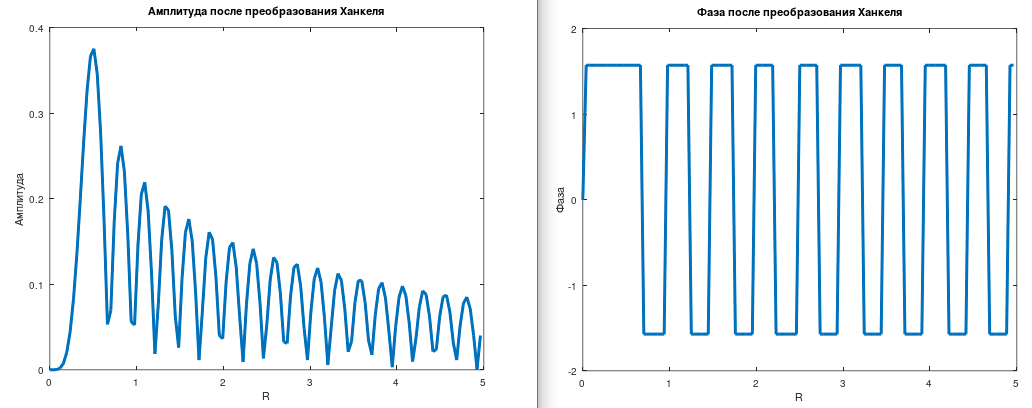


Рисунок 3 – Графики распределений амплитуды и фазы входной функции после преобразования Ханкеля

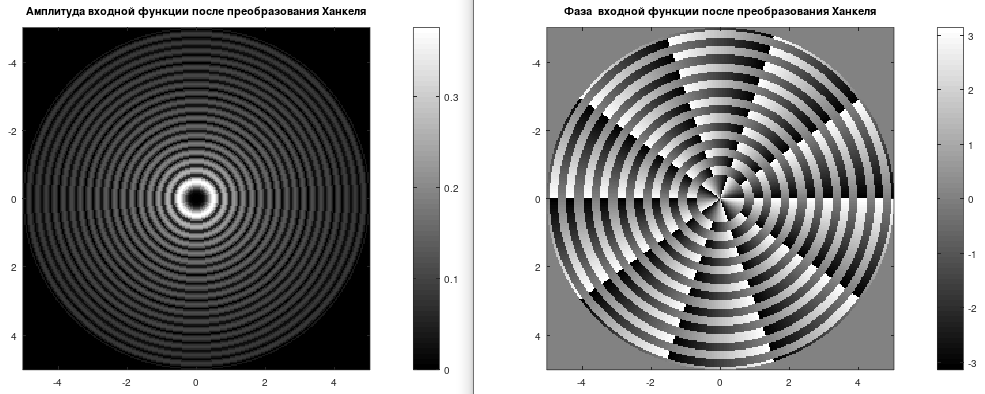


Рисунок 4 – Графики восстановленного изображения входной функции после преобразования Ханкеля

На 4 рисунке видно, что чем больше радиус кольца, тем он тусклее, это же подтверждает рисунок 3, так как по данному графику видно, что интенсивность уменьшается с увеличением радиуса. Деление картины распределения фазы соответствует значению порядка оптического вихря. Как как m = 5, то каждый полукруг делится на 5 конусов.

На рисунке 5 представлены графики амплитуды и фазы двумерного преобразования Фурье для восстановленной функции.

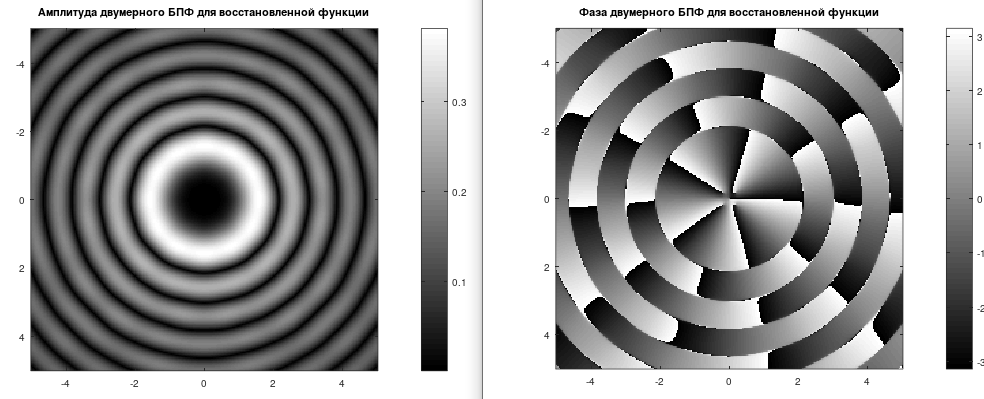


Рисунок 5 – Графики для двумерного преобразования Фурье для восстановленной функции

Результат преобразования БПФ имеет похожий график распределения на результат Ханкеля в центре пучка, но на краях видны некоторые дефекты.

На рисунке 6 показано, как время работы программы преобразования Ханкеля и БПФ зависит от количества точек дискретизации N, M.

Рисунок 6 – График зависимости времени работы преобразований от количества точек дискретизации

Результаты выполнения программы представлены в таблице 2.

Таблица 2 – Результаты выполнения программы

|  |  |  |
| --- | --- | --- |
| N, M | Преобразование Ханкеля, c | Преобразование Фурье, c |
| 127, 256 | 0,008262 | 0,124204 |
| 127, 512 | 0,008583 | 0,163437 |
| 127, 1024 | 0,008960 | 0,179274 |
| 255, 512 | 0,031002 | 0,249264 |
| 255, 1024 | 0,032623 | 0,254037 |
| 255, 2048 | 0,033839 | 0,306293 |

Сравнив результаты измерений, можно сделать вывод, что преобразование Фурье работает более, чем в 11 раз медленнее преобразования Ханкеля.

ПРИЛОЖЕНИЕ А  
Исходный код программы

m = 5;

R = 5;

n = 128;

h\_r = R/n;

r = 0:h\_r:(R-h\_r/2);

N = 2\*n-1;

M = 2048;

function result = circ(r)

if r <= 1

result = 1;

else

result = 0;

end

endfunction

function plot1d(range, fv, title\_abs, title\_arg, xlabel\_s)

figure;

plot(range, abs(fv), "linewidth", 2);

title(title\_abs);

xlabel(xlabel\_s);

ylabel("Амплитуда");

figure;

plot(range, angle(fv), "linewidth", 2);

xlabel(xlabel\_s);

ylabel("Фаза");

title(title\_arg);

endfunction

function plot2d(range1, range2, fv, title\_abs, title\_arg)

figure();

imagesc(range1, range2, abs(fv));

title(title\_abs);

colormap gray;

colorbar;

figure();

imagesc(range1, range2, arg(fv));

title(title\_arg);

colormap gray;

colorbar;

endfunction

function restored\_func = restore\_func(func, n, m, N)

restored\_func = zeros(N);

for j = 1:N

for k = 1:N

alpha = round(sqrt((j-n)^2 + (k-n)^2)) + 1;

if alpha <= n

restored\_func(j, k) = func(alpha) \* exp(1i\*m\*atan2(k-n, j-n));

endif

endfor

endfor

endfunction

function hankel\_result = hankel(func, r, h\_r, n, m)

hankel\_result = zeros(1, n);

for i = 1:n

hankel\_result(i) = sum(func.\*besselj(m, 2.\*pi.\*r.\*r(i)).\*r.\*h\_r);

endfor

hankel\_result = hankel\_result.\* ((2.\*pi)./(1i^m));

endfunction

function F = fourier\_bpf(fv, m, n, h\_r)

F = zeros(1, m);

start\_idx = floor((m - n) / 2) + 1;

end\_idx = start\_idx + n - 1;

F(start\_idx:end\_idx) = fv;

F = fftshift(F);

F = fft(F) \* h\_r;

F = fftshift(F);

F = F(length(F) / 2 - n / 2 + 1: length(F) / 2 + n / 2);

endfunction

function F\_bpf\_2d = fourier\_bpf\_2d(F\_bpf\_2d, h\_r, N, M)

for i = 1:N

F\_bpf\_2d(i, :) = fourier\_bpf(F\_bpf\_2d(i, :), M, N, h\_r);

endfor

for i = 1:N

F\_bpf\_2d(:, i) = fourier\_bpf(F\_bpf\_2d(:, i), M, N, h\_r);

endfor

endfunction

result = arrayfun(@circ, r./2) - arrayfun(@circ, 0.95.\*r./2);

plot1d(r, result, "Распределение амплитуды входной функции", "Распределение фазы входной функции", "R");

restored\_func = restore\_func(result, n, m, N);

plot2d([-R, R], [-R, R], restored\_func, "Амплитуда входной функции", "Фаза входной функции")

tic

hankel\_result = hankel(result, r, h\_r, n, m);

toc

plot1d(r, hankel\_result, "Амплитуда после преобразования Ханкеля", "Фаза после преобразования Ханкеля", "R");

restored\_hankel = restore\_func(hankel\_result, n, m, N);

plot2d([-R, R], [-R, R], restored\_hankel, "Амплитуда входной функции после преобразования Ханкеля", "Фаза входной функции после преобразования Ханкеля");

tic

fourier\_result = fourier\_bpf\_2d(restored\_func, h\_r, N, M);

toc

plot2d([-R, R], [-R, R], fourier\_result, "Амплитуда двумерного БПФ для восстановленной функции", "Фаза двумерного БПФ для восстановленной функции");