Final Project, Results and figures

Andrew Carr

April 13, 2016

(a) See gures (1-6)

```
81x41
Maximum error:4.117667e-01
Max error occurs on ray:1 and ring:8
Which is the point:2.423717e+00 1.344738e-06
Average Maximum Norm error for p_sc

avgerr = 0.0431

*****************Errors at Artificial Boundary r=r(N)********************

L2 Error for u_sc on r = R: 7.107130e-02

Relative L2 Error for u_sc on r = R: 5.602214e-02

101x51

Maximum error:3.170593e-01
Max error occurs on ray:1 and ring:9
Which is the point:2.384315e+00 1.185841e-06
Average Maximum Norm error for p_sc

avgerr = 0.0330

*****************Errors at Artificial Boundary r=r(N)*******************
```

```
L2 Error for u_sc on r = R: 4.681081e-02

Relative L2 Error for u_sc on r = R: 3.713212e-02

201x101

Maximum error:2.110997e-01
Max error occurs on ray:1 and ring:21
Which is the point:2.492788e+00 1.934809e-06
Average Maximum Norm error for p_sc

avgerr = 0.0193

*****************Errors at Artificial Boundary r=r(N)*******************

L2 Error for u_sc on r = R: 2.933733e-02

Relative L2 Error for u_sc on r = R: 2.357255e-02
```

Notice that the errors are decreasing when the grid is refined, as expected.

(b) When the grid is refined, the approximations become more smooth and there are fewer irregularities and less noise around the obstacle. This is because increasing the number of points, when the grid is well designed, allows for a better approximation.
See figures (7 - 9, 12)

(c) See figures (9-12)

(d) When Δt is changed to 0.05 we observe that our solution never converges. There is a severe instability in the approximations. This is because, as was proved in class, the approximation is only stable for small values of Δt

Figures

Figure 1: 81x41

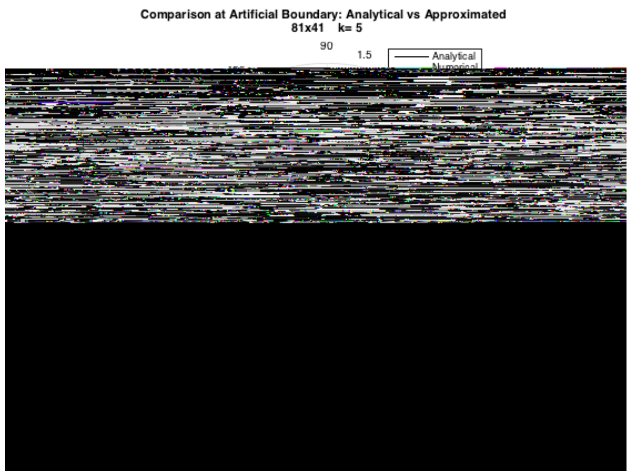


Figure 2: 81x41

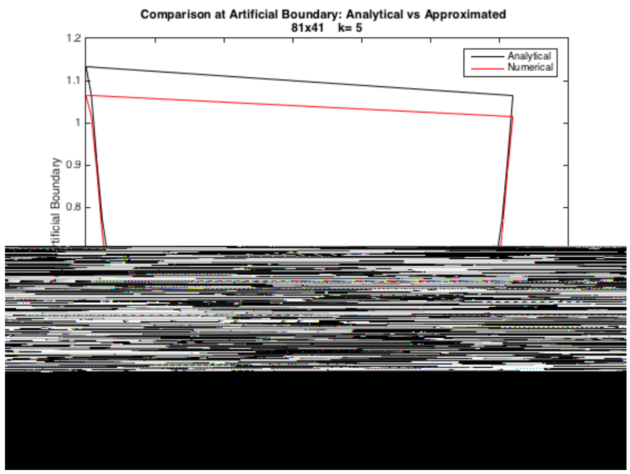


Figure 3: 101x51

Figure 4: 101x51

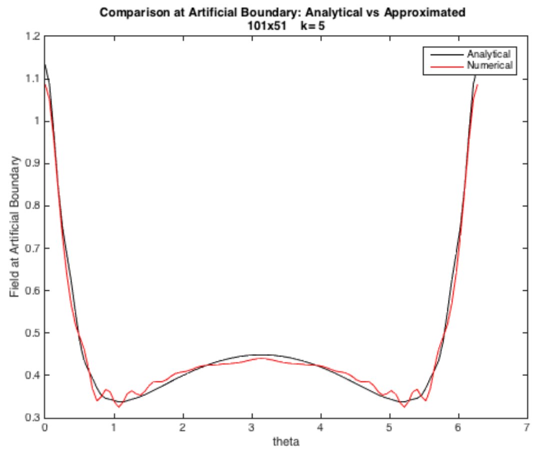


Figure 5: 201x101

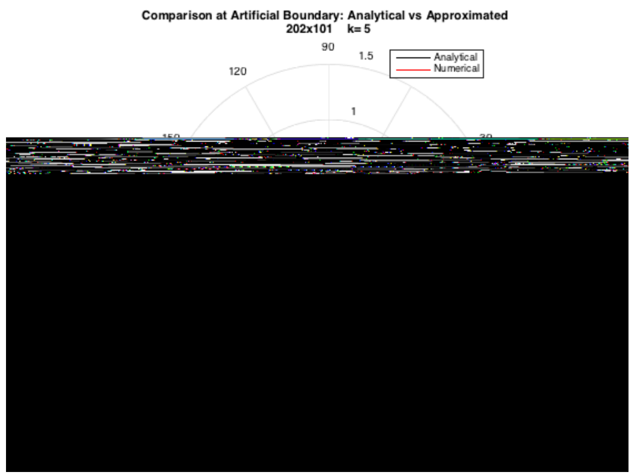


Figure 6: 201x101

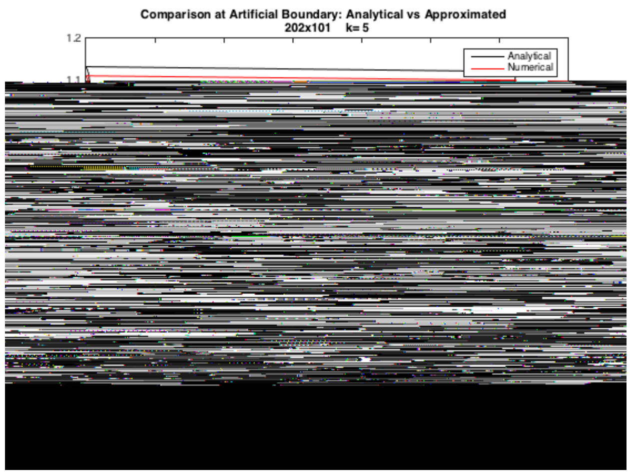


Figure 7: 81x41

Figure 8: 101x51

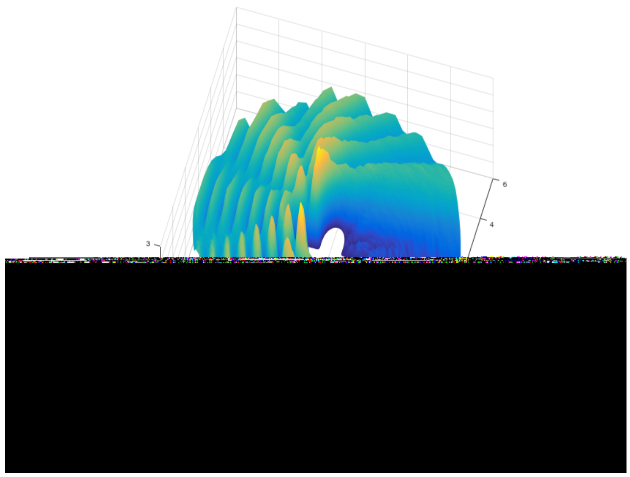


Figure 9: 201x101

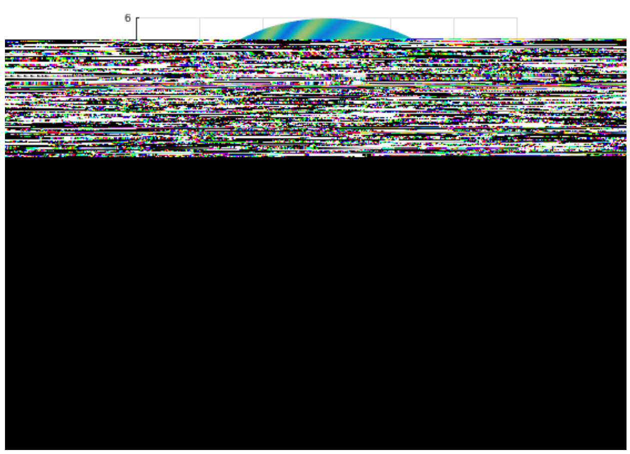


Figure 10: 81x41

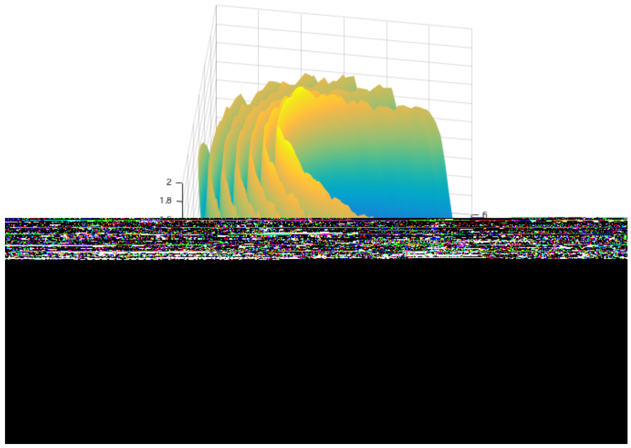


Figure 11: 101x51

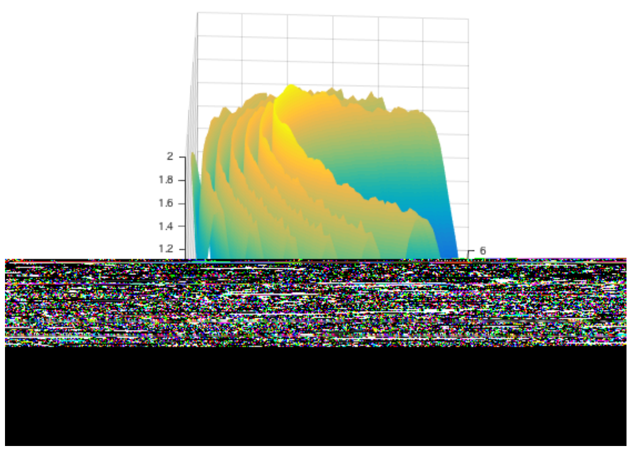


Figure 12: 201x101

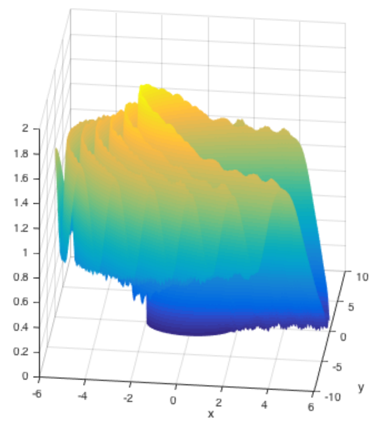
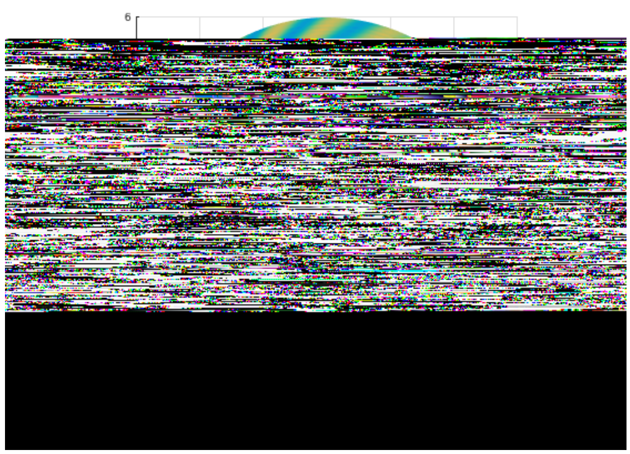


Figure 13: 201x101



Code:

```
%Main
close all;
clear all;

N1 = 201;
N2 = 101;

% choice = 'rose';
choice = 'circle';

w = 1.7912;
tol = 10^-6;
n = 8000;
k = 5;
omega = 5;
deltat = 10^-2;


[x,y] = definegrid(N1,N2,w,tol,n,choice);
[pfinal, p2] = final(x,y,N1,N2,n,k,omega,deltat);
ExactvsNumerical(x,y,p2,k,2,0,N1,N2);
orient landscape;
figure
title('final graph: abs')
surf(x,y,abs(pfinal),'EdgeColor','black');
shading interp;
grid on;
xlabel('x')
ylabel('y')
axis square;
```

```
function [ptotal, p2] = final(x,y,N1,N2,n,k,omega,deltat)
image = sqrt(-1);

r = 6;

firsttol = 10^-7;
secondtol = 10^-2;

%Matrix Initialization
p0 = zeros(N1,N2);
p1 = zeros(N1,N2);
p2 = zeros(N1,N2);

%initial condition for t = 0
time = 0;
for i = 1:N1-1
    p1(i,1) = -(exp(image*k*x(i,1)));

end
    %wrap up
    p1(N1,1) = p1(1,1);

%Time step
for time = 1:n

    %Computation at the obstacle boundary
    for i = 1:N1-1
        p2(i,1) = -(exp(image*k*x(i,1))*exp(-image*omega*time*deltat));
    end
    %wrap up
    p2(N1,1) = p2(1,1);

    %Computation of interior points
    for i = 1:N1-1
        for j = 2:N2-1
            xeta = (x(i,j+1) - x(i,j-1))/2;
            yeta = (y(i,j+1) - y(i,j-1))/2;
```

```
        c = 1;

        if i == 1
            ypsi = (y(i+1,j)-y(N1-1,j))/2;
            xpsi = (x(i+1,j)-x(N1-1,j))/2;

            jacobian = xpsi*yeta-xeta*ypsi;

            alpha = xeta^2 + yeta^2;
            beta = xpsi*xeta + ypsi*yeta;
            gamma = xpsi^2 + ypsi^2;

            g = ((c*deltat)/jacobian)^2;

            p2(i,j) = p1(N1-1,j)*(alpha*g) + p1(i+1,j)*(alpha*g)+p1(i,j-1)*(g
            - 2*g*beta*(p1(i+1,j+1)-p1(i+1,j-1)-p1(N1-1,j+1)+p1(N1-1,j-1))/4
        else
            ypsi = (y(i+1,j)-y(i-1,j))/2;
            xpsi = (x(i+1,j)-x(i-1,j))/2;

            jacobian = xpsi*yeta-xeta*ypsi;

            alpha = xeta^2 + yeta^2;
            beta = xpsi*xeta + ypsi*yeta;
            gamma = xpsi^2 + ypsi^2;

            g = ((c*deltat)/jacobian)^2;

            p2(i,j) = p1(i-1,j)*(alpha*g) + p1(i+1,j)*(alpha*g)+p1(i,j-1)*(ga
            - 2*g*beta*(p1(i+1,j+1)-p1(i+1,j-1)-p1(i-1,j+1)+p1(i-1,j-1))/4 -
        end
    end
end
%wrap up
for j = 2:N2-1
    p2(N1,j) = p2(1,j);
end
```

```
%Computation at artificial boundary
for i = 1:N1-1

    %Backwards scheme to eliminate ghost points
    xeta = (3/2)*x(i,N2) - 2*x(i,N2-1) + (1/2)*x(i,N2-2);
    yeta = (3/2)*y(i,N2) - 2*y(i,N2-1) + (1/2)*y(i,N2-2);

    c = 1;

    if i == 1
        ypsi = (y(i+1,N2)-y(N1-1,N2))/2;
        xpsi = (x(i+1,N2)-x(N1-1,N2))/2;

        jacobian = xpsi*yeta-xeta*ypsi;
        g = ((c*deltat)/jacobian)^2;

        alpha = xeta^2 + yeta^2;
        beta = xpsi*xeta + ypsi*yeta;
        gamma = xpsi^2 + ypsi^2;

        %Variables used to ease calculation
        delt = c/(r*jacobian);
        lambda = y(i,N2)*xpsi-x(i,N2)*ypsi;
        kay = x(i,N2)*yeta - y(i,N2)*xeta;
        D = (1+(gamma*g)/(deltat*delt*lambda));
        capC = (-2*alpha*g-2*gamma*g+2);
        B1 = ((1/4)*(3*p1(i+1,N2) - 4*p1(i+1,N2-1) + p1(i+1,N2-2) - 3*p1(N1-1
        R1 = p0(i,N2)/(deltat*delt*lambda);
        R2 = 2*lambda*kay*(p1(i+1,N2) - p1(N1-1,N2))/2;
        R3 = c*p1(i,N2)/(delt*lambda*r);
        R4 = p1(i,N2-1);

        %Actual boundary equation
        p2(i,N2) = (g*alpha/D)*p1(i+1,N2) + (g*alpha/D)*p1(N1-1,N2)-(2*g*beta
            + (gamma*g/D)*R4+(gamma*g/D)*p1(i,N2-1)-p0(i,N2)/D+(capC/D)*p1(i,
    else
     ypsi = (y(i+1,N2)-y(i-1,N2))/2;
        xpsi = (x(i+1,N2)-x(i-1,N2))/2;
```

```
        jacobian = xpsi*yeta-xeta*ypsi;
        g = ((c*deltat)/jacobian)^2;

        alpha = xeta^2 + yeta^2;
        beta = xpsi*xeta + ypsi*yeta;
        gamma = xpsi^2 + ypsi^2;

        %Variables used to ease calculation
        delt = c/(r*jacobian);
        lambda = y(i,N2)*xpsi-x(i,N2)*ypsi;
        kay = x(i,N2)*yeta - y(i,N2)*xeta;
        D = (1+(gamma*g)/(deltat*delt*lambda));
        capC = (-2*alpha*g-2*gamma*g+2);
        B1 = ((1/4)*(3*p1(i+1,N2) - 4*p1(i+1,N2-1) + p1(i+1,N2-2) - 3*p1(i-1,
        R1 = p0(i,N2)/(deltat*delt*lambda);
        R2 = 2*lambda*kay*(p1(i+1,N2) - p1(i-1,N2))/2;
        R3 = c*p1(i,N2)/(delt*lambda*r);
        R4 = p1(i,N2-1);

        %Actual boundary equation
        p2(i,N2) = (g*alpha/D)*p1(i+1,N2) + (g*alpha/D)*p1(i-1,N2)-((2*g*beta
            + (gamma*g/D)*R4+(gamma*g/D)*p1(i,N2-1)-p0(i,N2)/D+(capC/D)*p1(i,
     end
end
    %wrapping up
    p2(N1,N2) = p2(1,N2);


%Stop criteria

fullvalue = 0;

for i = 1:N1
    for j = 1:N2
        diff = abs(abs(p2(i,j))-abs(p1(i,j)));
        fullvalue = fullvalue + diff;
    end
```

```
    end

    average = fullvalue / (N1*N2);

    if time <=2000
        if(average < firsttol)
            disp(time)
            break
        end
    elseif time > 2000
        if(average < secondtol)
            disp(time)
            break
        end
    end

    %Update pressure matrix
    p0 = p1;
    p1 = p2;


    if mod(time,100) == 0
        disp(time)
    end
end

%creation of incident wave for entire domain, to graph
pinc = zeros(N1,N2);
for new = 1:time
    for i = 1:N1
        for j = 1:N2
            pinc(i,j) = exp(sqrt(-1)*k*x(i,j))*exp(-sqrt(-1)*omega*new*deltat);
        end
    end
end

ptotal = p2 + pinc;
```