SA1 Wing Analysis
Interim Report 1

Cindy Wu and Yitak Lee

May 19, 2022

1 Introduction

This week we explore basic 1D panel numerical methods for a vortex sheet, and approximate a cylinder as a set of small panels, each which act as vortex sheets.

2 Exercise 1

Code for psiv.m:

```
function psixy = psipv(xc,yc,Gamma,x,y)
    psixy = -Gamma*log((x-xc)^2+(y-yc)^2)/(4*pi);
end
```

Code for script:

```
Gamma = 3;
xmin = -2.5;
xmax = 2.5;
ymin = -2;
ymax = 2;
nx = 51;
ny = 41;
psi = zeros(nx,ny);
xm = zeros(nx,ny);
ym = zeros(nx,ny);
xc = 0.5;
yc = 0.25;

for i=1:nx
    for j=1:ny
        xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
        ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
        psi(i,j)= psipv(xc,yc,Gamma,xm(i,j),ym(i,j));
    end
end


c = -0.4:0.05:1.2;
contour(xm,ym,psi,c);
```

Contour plot is shown in Figure 1.

3 Exercise 2

Code for refpaninf.m:

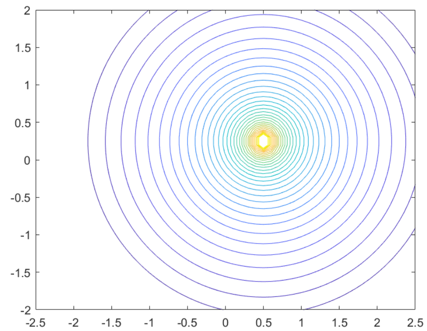

Figure 1: Contour plot for streamlines for singular vortex.

```
function [infa infb] = refpaninf(Delta,X,Y)
    if abs(Y)<10^(-7)
        Y = 10^(-7);
    end
    I0vals = -((X*log(X^2+Y^2)-(X-Delta)*log((X-Delta)^2+Y^2)-2*Delta+2*Y*(atan(X/Y)-atan((X
    -Delta)/Y))))/(4*pi);
    I1vals = 1/(8*pi)*((X^2+Y^2)*log(X^2+Y^2)-((X-Delta)^2+Y^2)*log((X-Delta)^2+Y^2)-2*X*
    Delta+Delta^2);
    infa = (1-X/Delta)*I0vals-I1vals/Delta;
    infb = X/Delta*I0vals+I1vals/Delta;
end
```

Code for script:

```
Gamma = 3;
xmin = -2.5;
xmax = 2.5;
ymin = -2;
ymax = 2;
nv = 100;
nx = nv;
ny = nv;
psi = zeros(nx,ny);
xm = zeros(nx,ny);
ym = zeros(nx,ny);
Delta = 1.5;
fa = zeros(nx,ny);
fb = zeros(nx,ny);

for i=1:nx
    for j=1:ny
        xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
        ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
        [fa(i,j),fb(i,j)]= refpaninf(Delta,xm(i,j),ym(i,j));
    end
end

c = -0.15:0.05:0.15;
figure
contour(xm,ym,fa,c)

figure
contour(xm,ym,fb,c)

dl = Delta/nv;
gamma = zeros(nv,1);
yc = 0;
```

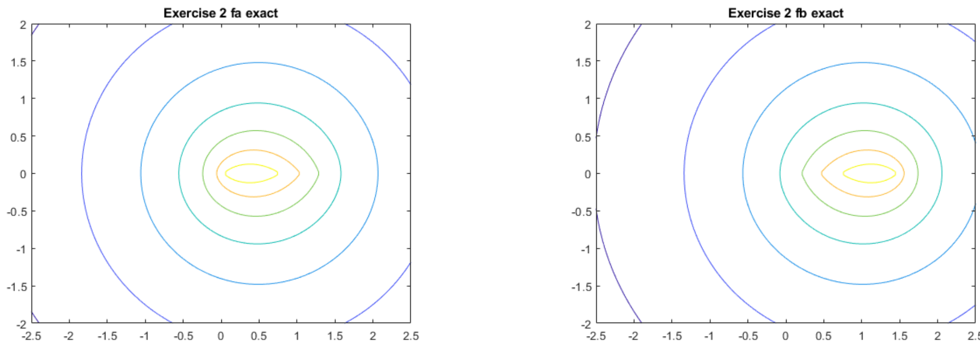

Figure 2: Plots showing contour plots of exact f_a (left) and f_b (right) values.

```
discrete_fa = zeros(nv); %psi at one location due to many point vortices
discrete_fb = zeros(nv);


for i=1:nx

    for j=1:ny
        xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
        ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
        for k=1:nv
            xc = dl*(2*k-1)/2;
            gamma_a = dl*(nv+0.5-k)/nv;
            gamma_b = dl*(k-0.5)/nv;

            discrete_fa(i,j) = discrete_fa(i,j) + psipv(xc,yc,gamma_a,xm(i,j),ym(i,j));
            discrete_fb(i,j) = discrete_fb(i,j) + psipv(xc,yc,gamma_b,xm(i,j),ym(i,j));
        end
    end
end

figure
c = -0.15:0.05:1.15;
contour(xm,ym,discrete_fa,c);

figure
contour(xm,ym,discrete_fb,c);
```

To explain some of the code script, consider that the value of gamma at the ends is a linear combination of every discretised gamma, proportional to the distance from that discrete vortex to the end. For γ_a, expect 99.5% of contribution from Γ_1 to go towards it, and 0.05% of Γ_1 to contribute towards γ_b.

4 Exercise 3

Code for panelinf.m:

```
function [infa infb] = panelinf(xa,ya,xb,yb,x,y)
    tangential = [xb yb] - [xa ya];
    t = tangential/norm(tangential);
    normal = [(ya-yb) (xb-xa)];
    n = normal/norm(normal);
    r = [x y] - [xa ya];
    X = dot(r,t);
    Y = dot(r,n);
    if abs(Y)<10^(-7);
```

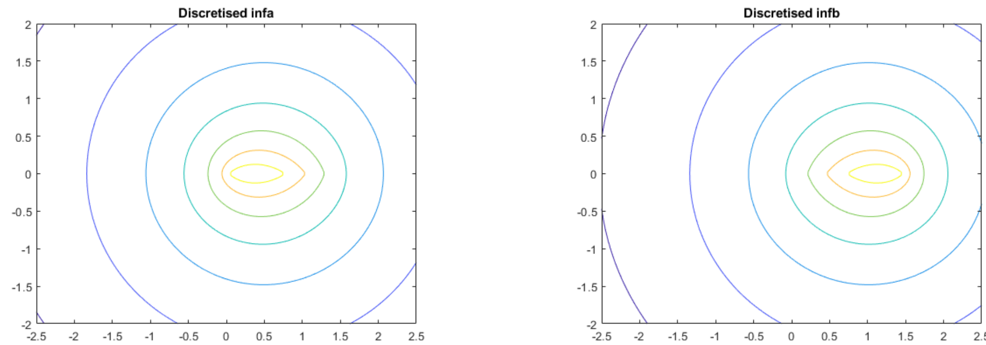

Figure 3: Plots showing contour plots of approximate f_a (left) and f_b (right) values.

```
        Y = 10^(-7);
    end
    [infa infb] = refpaninf(norm(tangential),X,Y);
end
```

Code for script:

```
Gamma = 3;
xa=3.5;
ya=2.5;
xb=1.6;
yb=1.1;
xmin = 0;
xmax = 5;
ymin = 0;
ymax = 4;
nv = 100;
nx = 51;
ny = 41;
psi = zeros(nx,ny);
xm = zeros(nx,ny);
ym = zeros(nx,ny);
Delta = 1.5;
fa = zeros(nx,ny);
fb = zeros(nx,ny);

for i=1:nx
    for j=1:ny
        xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
        ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
        [fa(i,j),fb(i,j)]= panelinf(xa,ya,xb,yb,xm(i,j),ym(i,j));
    end
end

c = -0.15:0.05:0.15
figure
contour(xm,ym,fa,c)

figure
contour(xm,ym,fb,c)

%start of discretised
dl = Delta/nv;
gamma = zeros(nv,1);
discrete_fa = zeros(nx,ny); %psi at one location due to many point vortices
discrete_fb = zeros(nx,ny);

```

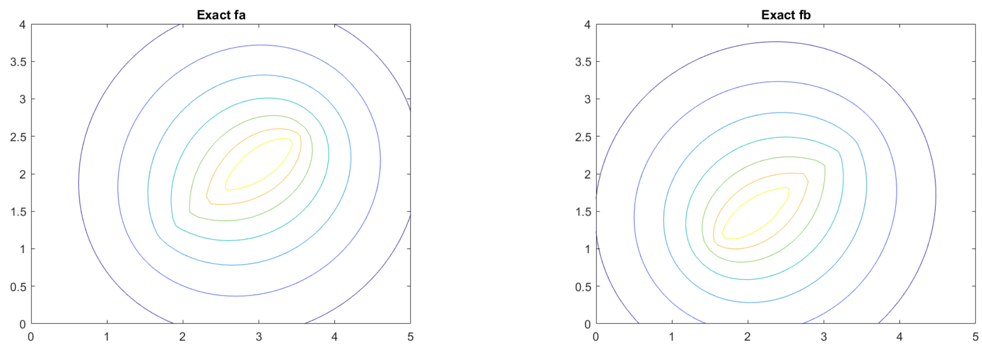

Figure 4: Exercise 3 plots showing contour plots of exact f_a (left) and f_b (right) values.

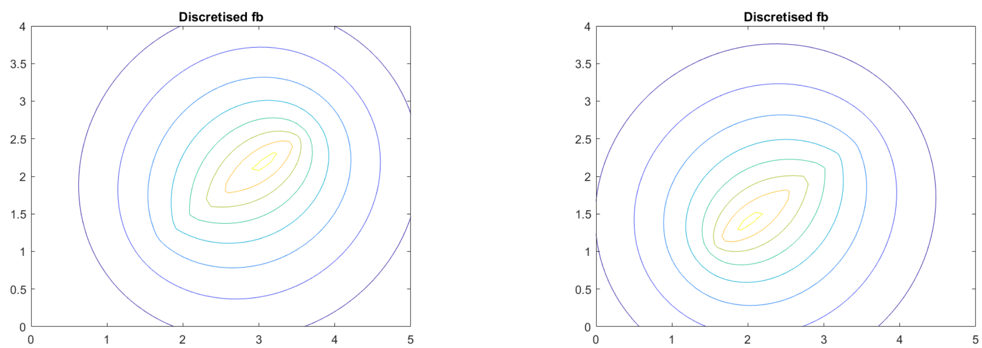

Figure 5: Exercise 3 plots showing contour plots of approximate f_a (left) and f_b (right) values.

```
for i=1:nx
    for j=1:ny
        xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
        ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
        for k=1:nv
            xc = xa+(xb-xa)*(2*k-1)/(2*nv);
            yc = ya+(yb-ya)*(2*k-1)/(2*nv);
            r = sqrt((yb-ya)^2+(xb-xa)^2);
            gamma_a = r*(nv+0.5-k)/(nv^2);
            gamma_b = r*(k-0.5)/(nv^2);
            discrete_fa(i,j) = discrete_fa(i,j) + psipv(xc,yc,gamma_a,xm(i,j),ym(i,j));
            discrete_fb(i,j) = discrete_fb(i,j) + psipv(xc,yc,gamma_b,xm(i,j),ym(i,j));
        end
    end
end

figure
c = -0.15:0.05:1.15;
contour(xm,ym,discrete_fa,c);

figure
contour(xm,ym,discrete_fb,c);
```

5 Exercise 4

Code for script:

```
np = 100;
xmin = 0;
xmax = 5;
ymin = 0;
ymax = 4;
nx=51;
ny=41;

% xm and ym are the grid points to evaluate psi at
xm = zeros(nx,ny);
ym = zeros(nx,ny);
psi = zeros(nx,ny);

% 101 values of theta
theta = (0:np)*2*pi/np;

% xs and ys are the panel intersection points
xs = zeros(np+1,1);
ys = zeros(np+1,1);

% gammas are evaluated on the surface of the cylinder at panel
% intersections
gamma = zeros(np+1,1);

for i=1:nx
    for j=1:ny
        xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
        ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
        for k=1:np
           % k is loop over all panels
           xs(k) = cos(theta(k));
           ys(k) = sin(theta(k));
           xs(k+1) = cos(theta(k+1));
           ys(k+1) = sin(theta(k+1));
           gamma_k = -2*sin(theta(k));
           gamma_k1 = -2*sin(theta(k+1));
           [fa,fb]= panelinf(xs(k),ys(k),xs(k+1),ys(k+1),xm(i,j),ym(i,j));
           psi(i,j) = psi(i,j)+gamma_k*fa+gamma_k1*fb;
        end
        % Now add contribution from free stream
        psi(i,j) = psi(i,j)+ym(i,j);
    end
end

figure
hold on
c = -1.75:0.25:1.75;
contour(xm,ym,psi,c);
plot(xs,ys);
hold off
```

Streamline plot is shown in Figure 6.

6 Exercise 5

Note that the index for i in A when it comes to the np-1 equations only goes up to np-1 as psi only has n rows, so cannot have a np+1 index for psi.

Each row in psi corresponds to one of the panel edges at which we are evaluating the streamfunction, which is why there are np rows. For each row there are np+1 columns because the circular format means

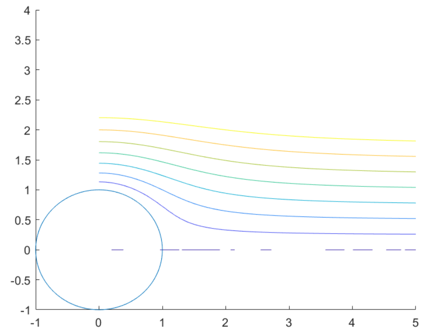

Figure 6: Streamline plot for exercise 4.

that the f_a and f_b values for the 1st point are split between the first and last columns. It should be noted there are actually np true psi values.

Code for build_lhs.m:

```
function lhsmat = build_lhs(xs,ys)
% xs is a vector of the edge of panel coordinates
    np = length(xs)-1;
    psip = zeros(np,np+1);
    lhsmat = zeros(np+1,np+1);

    % Build psi matrix
    % i loops over the panel edges at which psi is evaluated. There are np
    % such points

    % j loops over the end points of the panels which produce the
    % streamfunctions
    for i = 1:np
        for j = 1:np+1
            if j==1
                [fa fb] = panelinf(xs(j),ys(j),xs(j+1),ys(j+1),xs(i),ys(i));
                psip(i,j)=fa;
            elseif j==np+1
                [faprev fbprev] = panelinf(xs(j-1),ys(j-1),xs(j),ys(j),xs(i),ys(i));
                psip(i,j)=fbprev;
            else
                [fa fb] = panelinf(xs(j),ys(j),xs(j+1),ys(j+1),xs(i),ys(i));
                [faprev fbprev] = panelinf(xs(j-1),ys(j-1),xs(j),ys(j),xs(i),ys(i));
                psip(i,j)=fa+fbprev;
            end
        end
    end

    % Build A
    % A has a dimension of 101x101, i.e. np+1xnp+1
    lhsmat(1,1)=1;
    lhsmat(np+1,np+1)=1;

    for j=2:np+1
        lhsmat(1,j)=0;
    end

    for j=1:np
        lhsmat(np+1,j)=0;
    end

    for i=1:np-1
```

```
        lhsmat(i+1,:)=psip(i+1,:)-psip(i,:);
    end
end
```

Code for build_rhs.m:

```
function rhsvec = build_rhs(xs,ys,alpha)
    % xs is a vector of the edge of panel coordinates
    % This has np+1 entries as the first and last entries are repeated
    % There are np unique coordinates
    np = length(xs)-1;
    % rhsvec: vector that stores 101X1 values
    rhsvec = zeros(np+1,1);
    % Equations 7 and 8 require 0 on RHS
    % Equation 6 fills in the rest of the vector
    for i=2:np
        rhsvec(i) = ys(i)*cos(alpha)-xs(i)*sin(alpha)-ys(i+1)*cos(alpha)+xs(i+1)*sin(alpha);
    end
end
```

Code for script:

```
np = 100;
xs = zeros(np+1,1);
ys = zeros(np+1,1);
theta = (0:np)*2*pi/np;
alpha = pi/18;

% Fill xs,ys
for k=1:np+1
    xs(k) = cos(theta(k));
    ys(k) = sin(theta(k));
end

% Build A, b and find gamma using matrix inversion
A = build_lhs(xs,ys);
b = build_rhs(xs,ys,alpha);
gam = A\b;

% Plot of gamma against theta/pi
theta_plot = theta/pi;
figure
plot(theta_plot,gam);
axis([0 2 -2.5 2.5])

%======================================================
% Produce plot of streamlines as well

xmin = -5;
xmax = 5;
ymin = -4;
ymax = 4;
nx=101;
ny=81;

% xm and ym are the grid points to evaluate psi at
xm = zeros(nx,ny);
ym = zeros(nx,ny);
psi = zeros(nx,ny);

% gammas are evaluated on the surface of the cylinder at panel
% intersections

for i=1:nx
    for j=1:ny
        xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
        ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
```

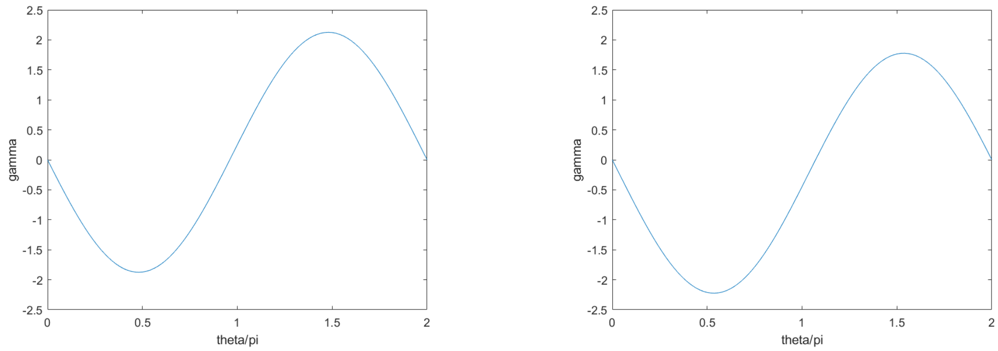

Figure 7: Exercise 5 plots showing gamma against theta/pi for alpha = 0 (left) and alpha = pi/18 (right).

```
        for k=1:np
           %k is loop over all panels
           xs(k) = cos(theta(k));
           ys(k) = sin(theta(k));
           xs(k+1) = cos(theta(k+1));
           ys(k+1) = sin(theta(k+1));
           [fa,fb]= panelinf(xs(k),ys(k),xs(k+1),ys(k+1),xm(i,j),ym(i,j));
           psi(i,j) = psi(i,j)+gam(k)*fa+gam(k+1)*fb;
        end
        %Now add contribution from free stream
        psi(i,j) = psi(i,j)+ym(i,j)*cos(alpha)-xm(i,j)*sin(alpha);
    end
end

figure
hold on
c = -1.75:0.25:1.75;
contour(xm,ym,psi,c);
plot(xs,ys);
hold off

%============================================================
% Total circulation = integral of gam over cylinder
% I will just multiply each gam value by the panel length and sum them up

% Panel length
r = sqrt((xm(1)-xm(2))^2+(ym(1)-ym(2))^2);

% Don't need to worry about np+1 index as it's 0 anyway
Gamma = sum(gam*r);

display(Gamma)
```

Figures 7 and 8 relate to surface gamma and streamline plots respectively. The total circulation was found to be 1.2564 for alpha = 0 and -2.2305 for alpha = pi/18.

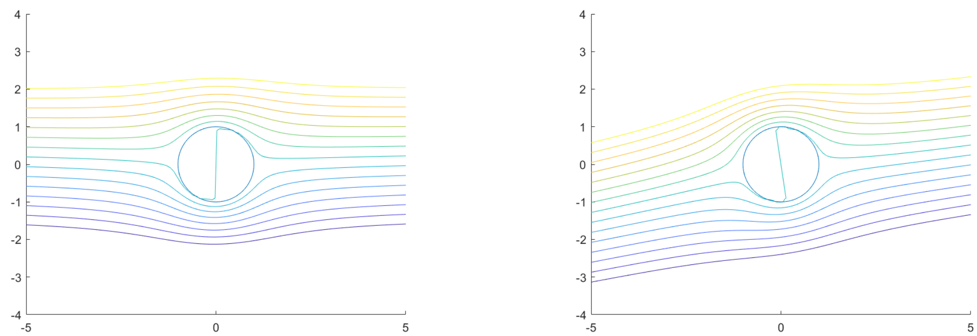

Figure 8: Exercise 5 plots showing streamlines for alpha = 0 (left) and alpha = pi/18 (right).