

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:

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

Code for script:

```
1 Gamma = 3;
2 xmin = -2.5;
3 xmax = 2.5;
4 ymin = -2;
5 ymax = 2;
6 nx = 51;
7 ny = 41;
8 psi = zeros(nx,ny);
9 xm = zeros(nx,ny);
10 ym = zeros(nx,ny);
11 xc = 0.5;
12 yc = 0.25;
13
14 for i=1:nx
15     for j=1:ny
16         xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
17         ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
18         psi(i,j)= psipv(xc,yc,Gamma,xm(i,j),ym(i,j));
19     end
20 end
21
22
23 c = -0.4:0.05:1.2;
24 contour(xm,ym,psi,c);
```

Contour plot is shown in Figure 1.

3 Exercise 2

Code for refpaninf.m:

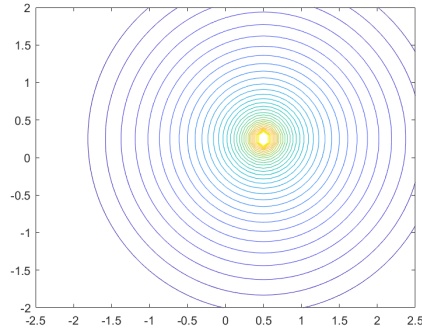


Figure 1: Contour plot for streamlines for singular vortex.

```

1 function [infa infb] = refpaninf(Delta,X,Y)
2     if abs(Y)<10^(-7)
3         Y = 10^(-7);
4     end
5     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);
6     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);
7     infa = (1-X/Delta)*I0vals-I1vals/Delta;
8     infb = X/Delta*I0vals+I1vals/Delta;
9 end

```

Code for script:

```

1 Gamma = 3;
2 xmin = -2.5;
3 xmax = 2.5;
4 ymin = -2;
5 ymax = 2;
6 nv = 100;
7 nx = nv;
8 ny = nv;
9 psi = zeros(nx,ny);
10 xm = zeros(nx,ny);
11 ym = zeros(nx,ny);
12 Delta = 1.5;
13 fa = zeros(nx,ny);
14 fb = zeros(nx,ny);
15
16 for i=1:nx
17     for j=1:ny
18         xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
19         ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
20         [fa(i,j),fb(i,j)]= refpaninf(Delta,xm(i,j),ym(i,j));
21     end
22 end
23
24 c = -0.15:0.05:0.15;
25 figure
26 contour(xm,ym,fa,c)
27
28 figure
29 contour(xm,ym,fb,c)
30
31 dl = Delta/nv;
32 gamma = zeros(nv,1);
33 yc = 0;

```

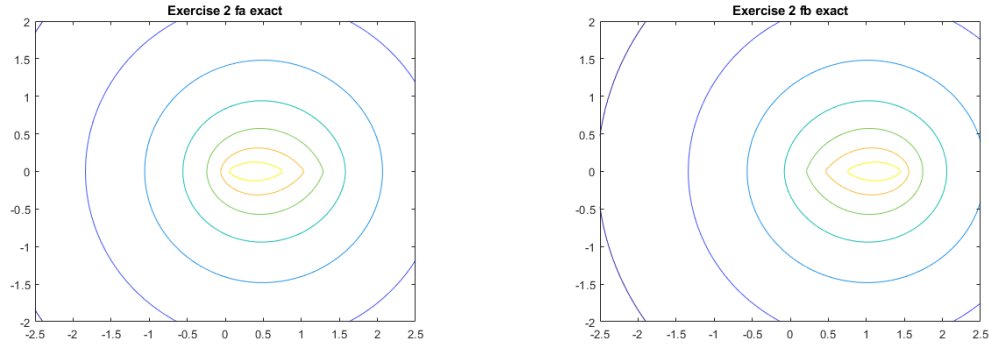


Figure 2: Plots showing contour plots of exact f_a (left) and f_b (right) values.

```

34 discrete_fa = zeros(nv); %psi at one location due to many point vortices
35 discrete_fb = zeros(nv);
36
37
38 for i=1:nx
39     for j=1:ny
40         xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
41         ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
42         for k=1:nv
43             xc = dl*(2*k-1)/2;
44             gamma_a = dl*(nv+0.5-k)/nv;
45             gamma_b = dl*(k-0.5)/nv;
46
47             discrete_fa(i,j) = discrete_fa(i,j) + psipv(xc,yc,gamma_a,xm(i,j),ym(i,j));
48             discrete_fb(i,j) = discrete_fb(i,j) + psipv(xc,yc,gamma_b,xm(i,j),ym(i,j));
49         end
50     end
51 end
52 end
53
54 figure
55 c = -0.15:0.05:1.15;
56 contour(xm,ym,discrete_fa,c);
57
58 figure
59 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:

```

1 function [infa infb] = panelinf(xa,ya,xb,yb,x,y)
2     tangential = [xb yb] - [xa ya];
3     t = tangential/norm(tangential);
4     normal = [(ya-yb) (xb-xa)];
5     n = normal/norm(normal);
6     r = [x y] - [xa ya];
7     X = dot(r,t);
8     Y = dot(r,n);
9     if abs(Y) < 10^(-7);

```

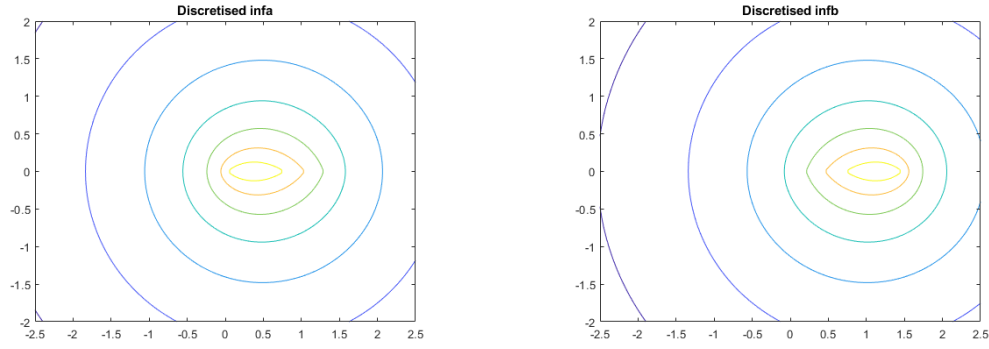


Figure 3: Plots showing contour plots of approximate f_a (left) and f_b (right) values.

```

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

```

Code for script:

```

1 Gamma = 3;
2 xa=3.5;
3 ya=2.5;
4 xb=1.6;
5 yb=1.1;
6 xmin = 0;
7 xmax = 5;
8 ymin = 0;
9 ymax = 4;
10 nv = 100;
11 nx = 51;
12 ny = 41;
13 psi = zeros(nx,ny);
14 xm = zeros(nx,ny);
15 ym = zeros(nx,ny);
16 Delta = 1.5;
17 fa = zeros(nx,ny);
18 fb = zeros(nx,ny);
19
20 for i=1:nx
21     for j=1:ny
22         xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
23         ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
24         [fa(i,j),fb(i,j)]= panelinf(xa,ya,xb,yb,xm(i,j),ym(i,j));
25     end
26 end
27
28 c = -0.15:0.05:0.15
29 figure
30 contour(xm,ym,fa,c)
31
32 figure
33 contour(xm,ym,fb,c)
34
35 %start of discretised
36 dl = Delta/nv;
37 gamma = zeros(nv,1);
38 discrete_fa = zeros(nx,ny); %psi at one location due to many point vortices
39 discrete_fb = zeros(nx,ny);
40

```

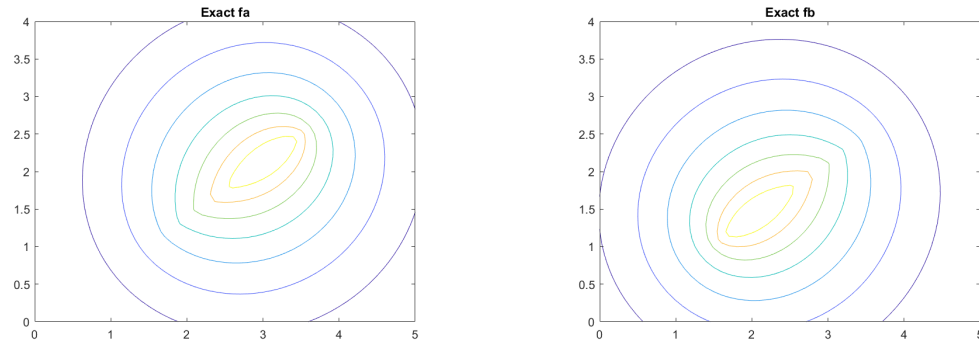


Figure 4: Exercise 3 plots showing contour plots of exact f_a (left) and f_b (right) values.

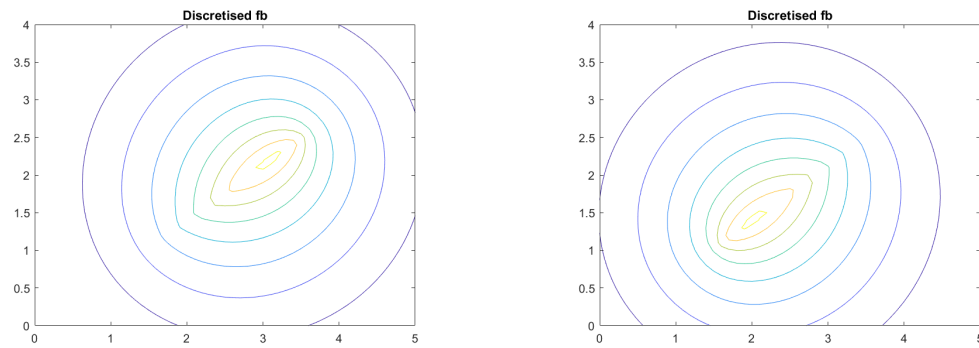


Figure 5: Exercise 3 plots showing contour plots of approximate f_a (left) and f_b (right) values.

```

41 for i=1:nx
42     for j=1:ny
43         xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
44         ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
45         for k=1:nv
46             xc = xa+(xb-xa)*(2*k-1)/(2*nv);
47             yc = ya+(yb-ya)*(2*k-1)/(2*nv);
48             r = sqrt((yb-ya)^2+(xb-xa)^2);
49             gamma_a = r*(nv+0.5-k)/(nv^2);
50             gamma_b = r*(k-0.5)/(nv^2);
51             discrete_fa(i,j) = discrete_fa(i,j) + psipv(xc,yc,gamma_a,xm(i,j),ym(i,j));
52             discrete_fb(i,j) = discrete_fb(i,j) + psipv(xc,yc,gamma_b,xm(i,j),ym(i,j));
53         end
54     end
55 end
56
57 figure
58 c = -0.15:0.05:1.15;
59 contour(xm,ym,discrete_fa,c);
60
61 figure
62 contour(xm,ym,discrete_fb,c);

```

5 Exercise 4

Code for script:

```
1 np = 100;
2 xmin = 0;
3 xmax = 5;
4 ymin = 0;
5 ymax = 4;
6 nx=51;
7 ny=41;
8
9 % xm and ym are the grid points to evaluate psi at
10 xm = zeros(nx,ny);
11 ym = zeros(nx,ny);
12 psi = zeros(nx,ny);
13
14 % 101 values of theta
15 theta = (0:np)*2*pi/np;
16
17 % xs and ys are the panel intersection points
18 xs = zeros(np+1,1);
19 ys = zeros(np+1,1);
20
21 % gammas are evaluated on the surface of the cylinder at panel
22 % intersections
23 gamma = zeros(np+1,1);
24
25 for i=1:nx
26     for j=1:ny
27         xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
28         ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);
29         for k=1:np
30             % k is loop over all panels
31             xs(k) = cos(theta(k));
32             ys(k) = sin(theta(k));
33             xs(k+1) = cos(theta(k+1));
34             ys(k+1) = sin(theta(k+1));
35             gamma_k = -2*sin(theta(k));
36             gamma_k1 = -2*sin(theta(k+1));
37             [fa,fb]= panelinf(xs(k),ys(k),xs(k+1),ys(k+1),xm(i,j),ym(i,j));
38             psi(i,j) = psi(i,j)+gamma_k*fa+gamma_k1*fb;
39         end
40         % Now add contribution from free stream
41         psi(i,j) = psi(i,j)+ym(i,j);
42     end
43 end
44
45 figure
46 hold on
47 c = -1.75:0.25:1.75;
48 contour(xm,ym,psi,c);
49 plot(xs,ys);
50 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 ψ only has n rows, so cannot have a $np+1$ index for ψ .

Each row in ψ 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

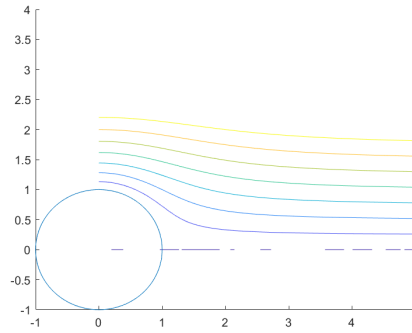


Figure 6: Streamline plot for exercise 4.

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:

```

1 function lhsmat = build_lhs(xs,ys)
2 % xs is a vector of the edge of panel coordinates
3     np = length(xs)-1;
4     psip = zeros(np,np+1);
5     lhsmat = zeros(np+1,np+1);
6
7     % Build psi matrix
8     % i loops over the panel edges at which psi is evaluated. There are np
9     % such points
10
11     % j loops over the end points of the panels which produce the
12     % streamfunctions
13     for i = 1:np
14         for j = 1:np+1
15             if j==1
16                 [fa fb] = panelinf(xs(j),ys(j),xs(j+1),ys(j+1),xs(i),ys(i));
17                 psip(i,j)=fa;
18             elseif j==np+1
19                 [faprev fbprev] = panelinf(xs(j-1),ys(j-1),xs(j),ys(j),xs(i),ys(i));
20                 psip(i,j)=fbprev;
21             else
22                 [fa fb] = panelinf(xs(j),ys(j),xs(j+1),ys(j+1),xs(i),ys(i));
23                 [faprev fbprev] = panelinf(xs(j-1),ys(j-1),xs(j),ys(j),xs(i),ys(i));
24                 psip(i,j)=fa+fbprev;
25             end
26         end
27     end
28
29     % Build A
30     % A has a dimension of 101x101, i.e. np+1xnp+1
31     lhsmat(1,1)=1;
32     lhsmat(np+1,np+1)=1;
33
34     for j=2:np+1
35         lhsmat(1,j)=0;
36     end
37
38     for j=1:np
39         lhsmat(np+1,j)=0;
40     end
41
42     for i=1:np-1

```

```

43     lhsmat(i+1,:)=psip(i+1,:)-psip(i,:);
44 end
45 end

```

Code for build_rhs.m:

```

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

```

Code for script:

```

1 np = 100;
2 xs = zeros(np+1,1);
3 ys = zeros(np+1,1);
4 theta = (0:np)*2*pi/np;
5 alpha = pi/18;
6
7 % Fill xs,ys
8 for k=1:np+1
9     xs(k) = cos(theta(k));
10    ys(k) = sin(theta(k));
11 end
12
13 % Build A, b and find gamma using matrix inversion
14 A = build_lhs(xs,ys);
15 b = build_rhs(xs,ys,alpha);
16 gam = A\b;
17
18 % Plot of gamma against theta/pi
19 theta_plot = theta/pi;
20 figure
21 plot(theta_plot,gam);
22 axis([0 2 -2.5 2.5])
23
24 %=====
25 % Produce plot of streamlines as well
26
27 xmin = -5;
28 xmax = 5;
29 ymin = -4;
30 ymax = 4;
31 nx=101;
32 ny=81;
33
34 % xm and ym are the grid points to evaluate psi at
35 xm = zeros(nx,ny);
36 ym = zeros(nx,ny);
37 psi = zeros(nx,ny);
38
39 % gammas are evaluated on the surface of the cylinder at panel
40 % intersections
41
42 for i=1:nx
43     for j=1:ny
44         xm(i,j)= xmin + (i-1)*(xmax-xmin)/(nx-1);
45         ym(i,j)= ymin + (j-1)*(ymax-ymin)/(ny-1);

```

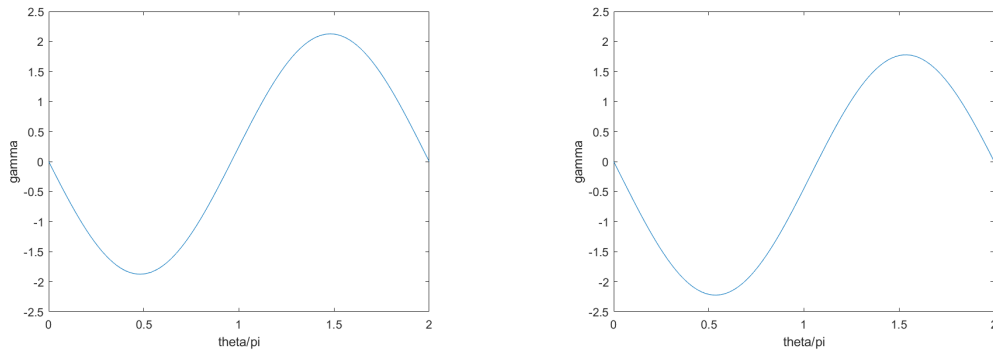



Figure 7: Exercise 5 plots showing gamma against theta/pi for $\alpha = 0$ (left) and $\alpha = \pi/18$ (right).

```

46     for k=1:np
47         %k is loop over all panels
48         xs(k) = cos(theta(k));
49         ys(k) = sin(theta(k));
50         xs(k+1) = cos(theta(k+1));
51         ys(k+1) = sin(theta(k+1));
52         [fa,fb]= panelinf(xs(k),ys(k),xs(k+1),ys(k+1),xm(i,j),ym(i,j));
53         psi(i,j) = psi(i,j)+gam(k)*fa+gam(k+1)*fb;
54     end
55     %Now add contribution from free stream
56     psi(i,j) = psi(i,j)+ym(i,j)*cos(alpha)-xm(i,j)*sin(alpha);
57 end
58 end
59
60 figure
61 hold on
62 c = -1.75:0.25:1.75;
63 contour(xm,ym,psi,c);
64 plot(xs,ys);
65 hold off
66
67 %=====
68 % Total circulation = integral of gam over cylinder
69 % I will just multiply each gam value by the panel length and sum them up
70
71 % Panel length
72 r = sqrt((xm(1)-xm(2))^2+(ym(1)-ym(2))^2);
73
74 % Don't need to worry about np+1 index as it's 0 anyway
75 Gamma = sum(gam*r);
76
77 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 $\alpha = \pi/18$.

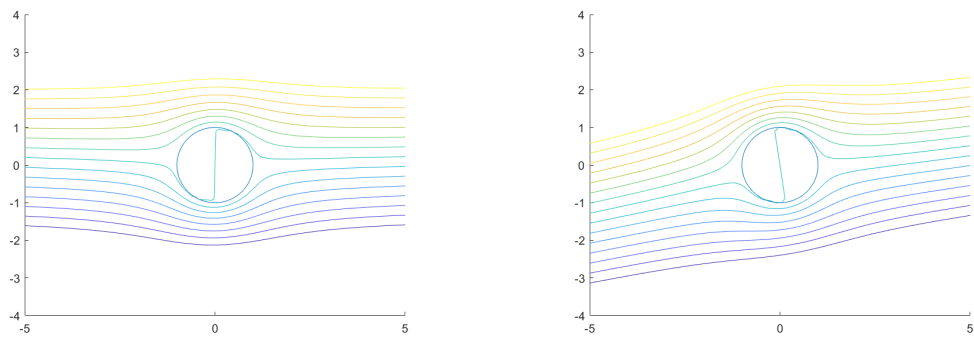


Figure 8: Exercise 5 plots showing streamlines for $\alpha = 0$ (left) and $\alpha = \pi/18$ (right).