SMA 5212/16.920J/2.096J Numerical Methods for Partial Differential Equations
Problem Set 4 - Integral Equation Methods

Issued: 8 May 2003 Due: 15 May 2003

1) As mentioned in class, second kind integral equations can sometimes be singular. In this problem you will analytically investigate a specific example of singularity in a second kind integral equation.

Please consider solving the following second kind integral equation

$$\lambda\sigma(x) - \int_0^1 (x + x')\sigma(x')dx' = c_1 x + c_2 \quad for\ all\ x \in [0, 1] \tag{1}$$

where λ is a real scalar, c_1, c_2 are constants, and σ is the unknown density function.

a) Determine a $\sigma(x)$ that satisfies (1) when $\lambda = 0.5$, $c_1 = 1$, and $c_2 = 0$ (Hint: try $\sigma(x) = constant$).

b) Find two values of λ for which (1) does NOT have a unique solution.

c) For each value of λ determined in (b), give two solutions which satisfy (1). You are free to pick c_1 and c_2 to be values which make the problem easiest.

2) In this problem you will examine using a Nystrom method to solve the integral equation

$$\sigma(x) + \int_0^1 (x - x')^2\sigma(x') = cos\frac{\pi}{2}x.$$

To help you get started, we have provided two matlab® files, legendrequad.m and evallegendrepoly.m. The file legendrequad.m generates test points and the weights for a quadrature scheme for evaluating integrals over the interval -1 to 1 (note that the interval for this problem is 0 to 1 so you will have to modify these files. Also, our implementation is limited to 25 points).

a) In the legendrequad.m code, you will see there is a question about forming the right-hand side for the system from which the quadrature weights are determined. Please answer the question.

b) Rescale the points and weights produced by the legendrequad so that you can use them to evaluate integrals on the interval $[0, 1]$. Test your modification by demonstrating that you can nearly exactly compute the integral

$$\int_0^1 x^p dx$$

for values of p equal to one less than the quadrature order.

c) Use the Nystrom method to solve the above integral equation, and determine experimentally how the error decays as the number of points in the Nystrom method increases. To examine convergence, consider looking at the difference between solutions computed with progressively more points. Be careful how you estimate those differences, the test point locations all change when you change the quadrature points. You will need to interpolate, with what order polynomial will you interpolate?

3) In this problem, you will examine using a Nystrom method to solve a problem which is very similar to the problem you solved above,

$$\sigma(x) + \int_{-1}^{1} |x - x'|\sigma(x') = cos\frac{\pi}{2}x.$$

a) Use the Nystom method to solve the above integral equation, and determine how the error decays as the number of points in the discretization increases.

b) Explain why the convergence is not as rapid in this case as it was in problem 2.

c) How would you try to improve the accuracy?

4) For this problem we developed some extensive matlab® code to help you understand boundary-element methods in three dimensions. In particular

you will modify our matlab® code to generate a method for computing the air velocity generated by a moving wing.

In this example we will be using the simplified potential flow model of the air. In this model the air velocity v at a point $\boldsymbol{x}$ is assumed to be irrotational. Therefore, the velocity can be expressed as the gradient of a scalar velocity potential u, as in

$$\boldsymbol{v}(\boldsymbol{x}) = \nabla u(\boldsymbol{x}).$$

If the air is assumed to be incompressible, the air velocity must have zero divergence as in

$$\nabla \cdot \boldsymbol{v}(\boldsymbol{x}) = 0.$$

Combining the above two equations yields Laplace's equation for the velocity potential u,

$$\nabla^2 u(\boldsymbol{x}) = 0.$$

Since the wing is moving with a velocity $\boldsymbol{v}_0(\boldsymbol{x})$, and air can not penetrate the wing, the air velocity in the direction normal to the wing must match the wing velocity. Nonpenetration therefore implies that

$$\frac{\partial u(\boldsymbol{x})}{\partial n_x} = \boldsymbol{v}_0(\boldsymbol{x}) \cdot \boldsymbol{n}_x$$

where n_x is the normal to the wing at point $\boldsymbol{x}$. Note that the potential flow problem is an exterior Neumann problem.

The wing surface values of the velocity potential can be used to determine the velocity tangent to the wing surface, and the wing drag can be determined from the tangent velocity. Your job in this problem will be to develop methods to solve the exterior Neumann potential flow problem for the wing surface values of the velocity potential u. You do not need to compute the tangent velocities or the drag.

We have developed a set of matlab® routines that use centroid collocation to solve the exterior Dirichlet problem, and expect you to modify the matlab® routines to solve the exterior Neumann problem. In particular, our code solves for the charge density on a conductor surface by solving the integral equation

$$u(\boldsymbol{x}) = \int_\Gamma \frac{\sigma(\boldsymbol{x}')}{\|\boldsymbol{x} - \boldsymbol{x}'\|} d\Gamma'$$

where $\boldsymbol{x}$ is a point in on the conductor surface, $u(\boldsymbol{x}) = 1$ for all points $\boldsymbol{x}$ on the conductor surface, and σ is the unknown surface charge density. Since

the potential is fixed at 1 over the conductor surface, our code is can be used to compute the capacitance of the conductor.

The matlab® code includes several files listed below.
calccap.m: This is the main routine which calculates capacitance.
calcp.m: This file contains the routine which analytically computes

$$\int_{panel} \frac{1}{\|x - x'\|} dS'$$

and can be used to compute panel integrals of other functions.
readpanels.m: This routines reads a set of panels describing the discretized geometry.
readpanel.m: This routine is called by readpanels and does the actual reading.
gencolloc.m: This routine computes panel centroids.
collocation.m: This routine sets up the collocation matrix.

We have also provided two examples, two wings (files **wing64.qif**, **wing120.qif**), with two successively refined discretizations, as the file names imply. Once you have downloaded the files from the web, you can run an example by typing
$[C, matrix]$ = calccap('wing64.qif')
and then you can also examine the matrix.

a) Derive an integral formulation for the exterior Neumann potential flow problem by differentiating the first kind equation for the Dirichlet problem. Please clearly indicate the boundary conditions assuming the wing is moving at unit speed along the x-axis.

b) Use Green's theorem to derive a different integral formulation for the exterior Neumann potential flow problem. Again, please clearly indicate the boundary conditions assuming the wing is moving at unit speed along the x-axis.

c) The routine in **calcp.m** can be used to compute the directional derivative of a panel potential at any point $\boldsymbol{x}$. What result does **calcp.m** produce when the point $\boldsymbol{x}$ is actually on the panel and the derivative is in the direction of the panel normal. What is the result when the point $\boldsymbol{x}$ is a little

below the panel or a little above the panel. Is **calcp.m** performing correctly?

d) Use the routines we provided, or ones of your own, to develop matlab® codes which apply centroid collocation to the two formulations you derived for the exterior Neumann problem. How would you compare the two methods?