Lecture Notes : Spectral Collocation Methods and Preconditioning

Byeong-Chun SHIN

Department of Mathematics, Chonnam National University, Korea

CNU Numerical Analysis Lab.

2008. 9. 22

1 Spectral Collocation Method and FDM Preconditioning

- Spectral Collocation Method
  - FDM for elliptic equation
  - FDM for elliptic equation

Spectral Collocation Method

Basic Program : on 1-dimension interval (a, b)

to find differentiation matrix, Gauss-Lobatto points and weights

```
[D, t, w] = leq_mat(N);  % N = poly order, [cheb_mat(N)]
t = a+(b-a)/2*(t+1);     w = (b-a)/2*w;
D = 2/(b-a)*D;
```

Two dimensional version : (c.f. sparse matrices)

```
[xg,yg] = meshgrid(t,t); % 2-D Gauss points
W = kron(diag(w), diag(w));  M = eye(N^2);
Sx = kron(D, eye(N));     Sy = kron(eye(N), D);
Lap = kron(D^2, eye(N)) + kron(eye(N), D^2);
```

Boundary Nodes :

```
b_l = 1:N;  b_b = N+1:N:N*(N-2)+1;
b_t = 2*N:N:N*(N-1); b_r = N*(N-1)+1:N:N^2];
bind = sort([b_l, b_b, b_t, b_r]); % boundary nodes
iind = setdiff([1:N^2], bind);     % interior nodes
```

Let $\Omega = (a, b)^2$. Consider the following equation

$$-e(x, y)\Delta u + \mathbf{b}(x, y)\nabla u + c(x, y)u = f(x, y)$$

or

$$-e(x, y)(u_{xx} + u_{yy}) + b_1(x, y)u_x + b_2(x, y)u_y + c(x, y)u = f(x, y).$$

Dirichlet Boundary condition :

$$u(x, y) = g(x, y) \quad \text{on } \partial\Omega.$$

Denote by

```
E = diag(e(xg, yg));     C = diag(c(xg,yg));
B1 = diag(b_1(xg,yg)); B2 = diag(b_2(xg,yg));
U = approximate vector for solution u(xg, yg).
```

The resulting linear system :

$$\mathbf{L}_N \mathbf{U} = \mathbf{F}$$

where

```
L_N = -E*Lap + B1*Sx + B2*Sy + C*M;
L_N(bind,:) = 0; L_N(bind,bind) = eye(length(bind));
F = f(xg,yg);
F(bind) = g(xg(bind), yg(bind));
```

Finite Difference Method

Let $h_j = t_{j+1} - t_j$ for $j = 1, \cdots, N-1$. i.e., `h = diff(t)`.

Then, the finite difference scheme is given by

$$
\begin{aligned}
&-\frac{2}{h_i + h_{i-1}}\left[\frac{u_{i-1,j}}{h_{i-1}} - \left(\frac{1}{h_i} + \frac{1}{h_{i-1}}\right)u_{i,j} + \frac{u_{i+1,j}}{h_i}\right]\\
&-\frac{2}{h_j + h_{j-1}}\left[\frac{u_{i,j-1}}{h_{j-1}} - \left(\frac{1}{h_j} + \frac{1}{h_{j-1}}\right)u_{i,j} + \frac{u_{i,j+1}}{h_j}\right]\\
&\qquad\qquad +b_1\frac{u_{i+1,j} - u_{i-1,j}}{h_i + h_{i-1}} + b_2\frac{u_{i,j+1} - u_{i,j-1}}{h_j + h_{j-1}}\\
&\qquad\qquad\qquad\qquad +c\, u_{i,j} = f_{i,j},
\end{aligned}
$$

for $i, j = 2, \cdots, N-1$.

Then we have the following five points stencil

$$\begin{bmatrix} 0 & \beta_{j+1} & 0 \\ \alpha_{i-1} & \alpha_i + \beta_j & \alpha_{i+1} \\ 0 & \beta_{j-1} & 0 \end{bmatrix}$$

$$\alpha_{i-1} = -\frac{2}{h_i + h_{i-1}} \cdot \frac{1}{h_{i-1}} - b_1 \frac{1}{h_i + h_{i-1}}$$

$$\alpha_i = \frac{2}{h_i + h_{i-1}} \cdot \left(\frac{1}{h_i} + \frac{1}{h_{i-1}} \right) = \frac{2}{h_i \, h_{i-1}}$$

$$\alpha_{i+1} = -\frac{2}{h_i + h_{i-1}} \cdot \frac{1}{h_i} + b_1 \frac{1}{h_i + h_{i-1}}$$

$$\beta_{j-1} = -\frac{2}{h_j + h_{j-1}} \cdot \frac{1}{h_{j-1}} - b_2 \frac{1}{h_j + h_{j-1}}$$

$$\beta_j = \frac{2}{h_j + h_{j-1}} \cdot \left(\frac{1}{h_j} + \frac{1}{h_{j-1}} \right) + c$$

$$\beta_{j+1} = -\frac{2}{h_j + h_{j-1}} \cdot \frac{1}{h_j} + b_2 \frac{1}{h_j + h_{j-1}}$$

Let

$$J_x = \mathrm{diag}\big(\alpha_{i-1},\, \alpha_i,\, \alpha_{i+1}\big) \quad \text{and} \quad J_y = \mathrm{diag}\big(\beta_{j-1},\, \beta_j,\, \beta_{j+1}\big).$$

```
b1 = 0;   b2 = 0;   c = 0;  % for example
h = diff(t);    % (N-1) vector
  h_s = 2./( h(2:N-1)+h(1:N-2) );
  h_p = 2./( h(2:N-1).*h(1:N-2) );
 alp1 = -h_s./h(1:N-2) - b1*h_s/2;  % (N-2)-vectors
 alp2 = h_p;
 alp3 = -h_s./h(2:N-1) + b1*h_s/2;
 bet1 = -h_s./h(1:N-2) - b2*h_s/2;
 bet2 = h_p + c;
 bet3 = -h_s./h(2:N-1) + b2*h_s/2;
```

In Matlab

```
Jx = diag([alp1,0],-1) + diag([1,alp2,1]) + diag([0,alp3],1
Jy = diag([bet1,0],-1) + diag([1,bet2,1]) + diag([0,bet3],1
B_N = kron(Jx, eye(N)) + kron(eye(N), Jy);
```

Preconditioning Spectral Collocation Method by FDM

< Du Fort-Frankel Method >

to solve the preconditioned system :

$$B_N^{-1}\, L_N\, U = B_N^{-1}\, F.$$

Let

$$\mu = \frac{\|\Psi\|_N}{\|\Phi\|_N} \quad \text{with} \quad \Phi = (1, -1, 1, -1, \cdots, 1)^t, \ \Psi = B_N^{-1}\, L_N\, \Phi.$$

With initial guesses $U_0 = U_1 = B_N^{-1}\, F$,

$$U^{k+1} = \alpha \Big[- B_N^{-1} \big(L_N\, U_k - F \big) + 2\sigma_2\, U_k \Big] + \beta\, U_{k-1}.$$

where

$$\alpha = \frac{2\sigma_1}{1 + 2\sigma_1\sigma_2}, \quad \beta = \frac{1 - 2\sigma_1\sigma_2}{1 + 2\sigma_1\sigma_2}, \quad \sigma_1 = \frac{1}{\sqrt{\mu}}, \quad \sigma_2 = \frac{\mu + 1}{4}.$$

Implementation of Du Fort-Frankel Method

```
mynorm = @(x) sqrt(x'*W*x);

[L,U] = lu(B_N);      % LU-decompostion of B_N

phi = ones(N,1);
phi(2:2:round(N/2)) = -phi(2:2:round(N/2)); % phi = (1,-1,1,-1, ...)
psi = U \ ( L \ (L_N*phi) );                % psi = inv(B_N)*L_N*phi

mu = mynorm(psi)/mynorm(phi);
sig1 = 1/sqrt(mu);  sig2 = (mu+1)/4;
alp = 2*sig1/(1+2*sig1*sig2); bet = (1-2*sig1*sig2)/(1+2*sig1*sig2);
```

```
X0 = U\(L\F); X1 = X0;
iter = 0;
R = Ln*X1-F;  % residual
R_err = mynorm( R );

while R_err > tol
   Xn = alp*( -U\(L\R) + 2*sig2*X1 ) + bet*X0;
   X1 = Xn; X0 = X1;
   R = L_N*X1-F;
   R_err = mynorm( R );
   iter = iter+1;
end
```