PHYSICS 410

2D RELAXATION

- Model problem

$$\nabla u(x, y) \equiv u_{xx} + u_{yy} = f(x, y)$$

on

$$\Omega : 0 \leq x \leq 1\,,\ 0 \leq y \leq 1$$

subject to

$$u(0, y) = u(1, y) = u(x, 0) = u(x, 1) = 0\,.$$

Script file: gs2d.m

```
% gs2d: Solves model elliptic problem using Gauss-Seidel iteration,.

% Discretization parameters ...
level = 6;
n = 2^level + 1;
h = 1.0 / (n - 1);

% Vectors of x , y coordinates for plotting ...
[x, y] = meshgrid(linspace(0.0, 1.0, n), linspace(0.0, 1.0, n));

% Frequencies for exact solution ...
omega_x = pi;
omega_y = pi;

[uxct f] = calc_exact(omega_x, omega_y, n, h);

% Sert boundary conditions and initialize solution to random values between
% 0 and 1 ...
u = zeros(n,n);
u(2:n-1,2:n-1) = rand(n-2,n-2);

% Number of relaxation sweeps ...
nsweep = 1000;
```

```
% Set to 1 to enable .avi creation
avienable = 1;
avisecs = 20;
aviframerate = 25;
aviframes = avisecs * aviframerate;
avifilename = ’gserr.avi’;

% AVI initialization and output ...
if avienable
   avifreq = round(nsweep / aviframes);
   aviobj = VideoWriter(avifilename);
   open(aviobj);

   uerr = uxct - u;
   surf(x, y, uerr);
   title(’Sweep 0’);

   framecount = 5 * aviframerate
   for iframe = 1 : framecount
      writeVideo(aviobj, getframe(gcf));
   end
end
```

```
% For each relaxation sweep ...
for isweep = 1 : nsweep
   rnorm = 0.0;
   for i = 2 : n - 1
      for j = 2 : n - 1
         % Update unknown and accumulate residual ...
         rij = (u(i+1,j) + u(i-1,j) + u(i,j+1) + u(i,j-1) - ...
                4*u(i,j)) / h^2 - f(i,j);
         rnorm = rnorm + rij^2;
         u(i,j) = u(i,j) + 0.25 * h^2 * rij;
      end
   end
   % Compute residual and error norms and output ...
   rnorm = sqrt(rnorm / n^2);
   uerr = uxct - u;
   fprintf(’sweep: %d  |res|: %g  |uerr|: %g\n’, isweep, rnorm, ...
           norm(uerr)/n);

   % AVI output ...
   if mod(isweep, avifreq) == 0
      surf(x, y, uerr);
      titlestr = sprintf(’Sweep %g\n’, isweep);
      title(titlestr);
      writeVideo(aviobj, getframe(gcf));
   end
```

```
end

% Close avi file ...
if avienable
   close(aviobj);
   fprintf('Created video file: %s\n', avifilename);
end
```

Function file: calc_exact.m

```
function [uxct f] = calc_exact(omega_x, omega_y, n, h)
%  calc_exact:  Calculates exact function values and corresponding r.h.s.
%  for model problem.
for i = 1 : n
   for j = 1 : n
      uxct(i,j) = sin(omega_x * (i - 1) * h) * sin(omega_y * (j - 1) * h);
      f(i,j) = -(omega_x^2 + omega_y^2) * uxct(i,j);
   end
end
```

Script file: sor2d.m

```
% sor2d: Solves model elliptic problem using SOR.

% Discretization parameters ...
level = 6;
n = 2^level + 1;
h = 1.0 / (n - 1);

% Define optimal SOR parameter ...
omega = 2.0 / (1.0 + sin(pi * h))

% Vectors of x , y coordinates for plotting ...
[x, y] = meshgrid(linspace(0.0, 1.0, n), linspace(0.0, 1.0, n));

% Frequencies for exact solution ...
omega_x = pi;
omega_y = pi;

[uxct f] = calc_exact(omega_x, omega_y, n, h);

% Sert boundary conditions and initialize solution to random values between
% 0 and 1 ...
u = zeros(n,n);
u(2:n-1,2:n-1) = rand(n-2,n-2);
```

```
% Number of relaxation sweeps ...
nsweep = 1000;

% Set to 1 to enable .avi creation
avienable = 1;
avisecs = 20;
aviframerate = 25;
aviframes = avisecs * aviframerate;
avifilename = ’sor.avi’;

% AVI initialization and output ...
if avienable
   avifreq = round(nsweep / aviframes);
   aviobj = VideoWriter(avifilename);
   open(aviobj);

   uerr = uxct - u;
   surf(x, y, u);
   title(’Sweep 0’);

   framecount = 5 * aviframerate
   for iframe = 1 : framecount
      writeVideo(aviobj, getframe(gcf));
   end
end
```

```
% For each relaxation sweep ...
for isweep = 1 : nsweep
   rnorm = 0.0;
   for i = 2 : n - 1
      for j = 2 : n - 1
         % Update unknown and accumulate residual ...
         rij = (u(i+1,j) + u(i-1,j) + u(i,j+1) + u(i,j-1) - ...
                4*u(i,j)) / h^2 - f(i,j);
         rnorm = rnorm + rij^2;
         % Compute GS value ...
         uij_gs = u(i,j) + 0.25 * h^2 * rij;
         % Overrelaxation step ...
         u(i,j) = omega * uij_gs + (1.0 - omega) * u(i,j);
      end
   end
   % Compute residual and error norms and output ...
   rnorm = sqrt(rnorm / n^2);
   uerr = uxct - u;
   fprintf(’sweep: %d  |res|: %g  |uerr|: %g\n’, isweep, rnorm, ...
           norm(uerr)/n);

   % AVI output ...
   if mod(isweep, avifreq) == 0
      surf(x, y, u);
      titlestr = sprintf(’Sweep %g\n’, isweep);
      title(titlestr);
      writeVideo(aviobj, getframe(gcf));
   end
end

% Close avi file ...
if avienable
   close(aviobj);
   fprintf(’Created video file: %s\n’, avifilename);
end
```