Numerical Analysis

10th ed

R L Burden, J D Faires, and A M Burden

Beamer Presentation Slides
Prepared by
Dr. Annette M. Burden
Youngstown State University

September 16, 2015

Selecting a Grid

1. Choose integers n and m to define step sizes $h = (b - a)/n$ and $k = (d - c)/m$.
2. Partition the interval $[a, b]$ into n equal parts of width h and the interval $[c, d]$ into m equal parts of width k.
3. Place a grid on the rectangle R by drawing vertical and horizontal lines through the points with coordinates (x_i, y_j), where

$$x_i = a + ih, \quad \text{for each } i = 0, 1, \ldots, n, \quad \text{and}$$
$$y_j = c + jk, \quad \text{for each } j = 0, 1, \ldots, m.$$

4. The lines $x = x_i$ and $y = y_j$ are **grid lines**, and their intersections are the **mesh points** of the grid.

Selecting a Grid

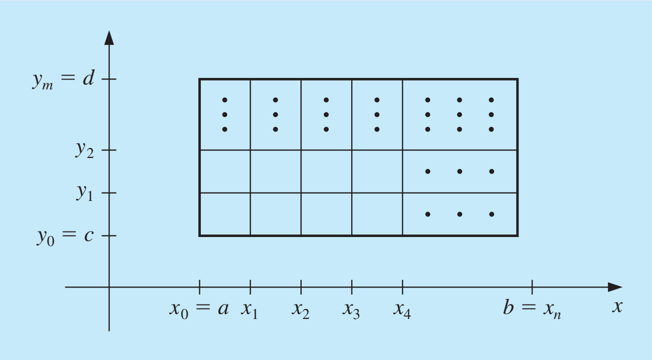

Figure: Figure 12.4

ALGORITHM 12.1 POISSON EQUATION FINITE-DIFFERENCE

To approximate the solution to the Poisson equation

$$\frac{\partial^2 u}{\partial x^2}(x, y) + \frac{\partial^2 u}{\partial y^2}(x, y) = f(x, y), \quad a \leq x \leq b, \quad c \leq y \leq d,$$

subject to the boundary conditions

$$u(x, y) = g(x, y) \quad \text{if } x = a \text{ or } x = b \quad \text{and} \quad c \leq y \leq d$$

and

$$u(x, y) = g(x, y) \quad \text{if } y = c \text{ or } y = d \quad \text{and} \quad a \leq x \leq b :$$

INPUT endpoints a, b, c, d; integers $m \geq 3$, $n \geq 3$; tolerance TOL; maximum number of iterations N.

OUTPUT approximations $w_{i,j}$ to $u(x_i, y_j)$ for each $i = 1, \ldots, n-1$ and for each $j = 1, \ldots, m-1$ or a message that the maximum number of iterations was exceeded.

ALGORITHM 12.1 POISSON EQUATION FINITE-DIFFERENCE

Step 1 Set $h = (b - a)/n$;
$k = (d - c)/m$.

Step 2 For $i = 1, \ldots, n - 1$ set $x_i = a + ih$. (*Steps 2 & 3 construct mesh pts.*)

Step 3 For $j = 1, \ldots, m - 1$ set $y_j = c + jk$.

Step 4 For $i = 1, \ldots, n - 1$
for $j = 1, \ldots, m - 1$ set $w_{i,j} = 0$.

Step 5 Set $\lambda = h^2/k^2$;
$\mu = 2(1 + \lambda)$;
$l = 1$.

Step 6 While $l \leq N$ do Steps 7–20. (*Steps 7–20 perform*
(*Gauss-Seidel iterations.*)

Step 7 Set $z = \left(-h^2 f(x_1, y_{m-1}) + g(a, y_{m-1}) + \lambda g(x_1, d)\right.$
$\left.+ \lambda w_{1,m-2} + w_{2,m-1}\right) / \mu$;
$NORM = |z - w_{1,m-1}|$;
$w_{1,m-1} = z$.

ALGORITHM 12.1 POISSON EQUATION FINITE-DIFFERENCE

Step 8 For $i = 2, \ldots, n-2$

set $z = \big(-h^2 f(x_i, y_{m-1}) + \lambda g(x_i, d) + w_{i-1,m-1}$
$+ w_{i+1,m-1} + \lambda w_{i,m-2} \big) / \mu;$

if $|w_{i,m-1} - z| > NORM$ then set $NORM = |w_{i,m-1} - z|;$

set $w_{i,m-1} = z.$

Step 9 Set $z = \big(-h^2 f(x_{n-1}, y_{m-1}) + g(b, y_{m-1}) + \lambda g(x_{n-1}, d)$
$+ w_{n-2,m-1} + \lambda w_{n-1,m-2} \big) / \mu;$

if $|w_{n-1,m-1} - z| > NORM$ then set $NORM = |w_{n-1,m-1} - z|;$

set $w_{n-1,m-1} = z.$

Step 10 For $j = m-2, \ldots, 2$ do Steps 11, 12, and 13.

Step 11 Set $z = \big(-h^2 f(x_1, y_j) + g(a, y_j) + \lambda w_{1,j+1}$
$+ \lambda w_{1,j-1} + w_{2,j} \big) / \mu;$

if $|w_{1,j} - z| > NORM$ then set $NORM = |w_{1,j} - z|;$

set $w_{1,j} = z.$

ALGORITHM 12.1 POISSON EQUATION FINITE-DIFFERENCE

Step 12 For $i = 2, \ldots, n-2$
set $z = \left(-h^2 f(x_i, y_j) + w_{i-1,j}\right.$
$\left.+\lambda w_{i,j+1} + w_{i+1,j} + \lambda w_{i,j-1}\right)/\mu;$
if $|w_{i,j} - z| > NORM$ then set $NORM = |w_{i,j} - z|;$
set $w_{i,j} = z.$

Step 13 Set $z = \left(-h^2 f(x_{n-1}, y_j) + g(b, y_j) + w_{n-2,j}\right.$
$\left.+\lambda w_{n-1,j+1} + \lambda w_{n-1,j-1}\right)/\mu;$
if $|w_{n-1,j} - z| > NORM$ then set $NORM = |w_{n-1,j} - z|;$
set $w_{n-1,j} = z.$

Step 14 Set $z = \left(-h^2 f(x_1, y_1) + g(a, y_1) + \lambda g(x_1, c) + \lambda w_{1,2} + w_{2,1}\right)/\mu;$
if $|w_{1,1} - z| > NORM$ then set $NORM = |w_{1,1} - z|;$
set $w_{1,1} = z.$

ALGORITHM 12.1 POISSON EQUATION FINITE-DIFFERENCE

Step 15 For $i = 2, \ldots, n-2$ set
$z = \left(-h^2 f(x_i, y_1) + \lambda g(x_i, c) + w_{i-1,1} + \lambda w_{i,2} + w_{i+1,1}\right) / \mu;$
if $|w_{i,1} - z| > NORM$ then set $NORM = |w_{i,1} - z|$;
set $w_{i,1} = z$.

Step 16 Set $z = \left(-h^2 f(x_{n-1}, y_1) + g(b, y_1)\right.$
$\left. + \lambda g(x_{n-1}, c) + w_{n-2,1} + \lambda w_{n-1,2}\right) / \mu;$
if $|w_{n-1,1} - z| > NORM$ then set $NORM = |w_{n-1,1} - z|$;
set $w_{n-1,1} = z$.

Step 17 If $NORM \leq TOL$ then do Steps 18 and 19.
Step 18 For $i = 1, \ldots, n-1$
for $j = 1, \ldots, m-1$ OUTPUT $(x_i, y_j, w_{i,j})$.
Step 19 STOP. (*The procedure was successful.*)
Step 20 Set $l = l + 1$.

Step 21 OUTPUT ('Maximum number of iterations exceeded');
STOP. (*The procedure was unsuccessful.*)

BACKWARD DIFFERENCE

The Backward-Difference method involves the mesh points (x_i, t_{j-1}), (x_{i-1}, t_j), and (x_{i+1}, t_j) to approximate the value at (x_i, t_j), as illustrated in Figure 12.9.

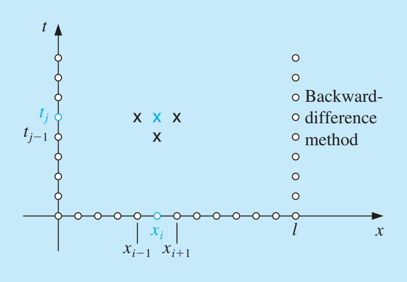

Figure: Figure 12.9

Algorithm 12.2: HEAT EQUATION BACKWARD-DIFFERENCE

To approximate the solution to the parabolic partial differential equation

$$\frac{\partial u}{\partial t}(x,t) - \alpha^2 \frac{\partial^2 u}{\partial x^2}(x,t) = 0, \quad 0 < x < l, \quad 0 < t < T,$$

subject to the boundary conditions

$$u(0,t) = u(l,t) = 0, \quad 0 < t < T,$$

and the initial conditions

$$u(x,0) = f(x), \quad 0 \le x \le l:$$

Algorithm 12.2: HEAT EQUATION BACKWARD-DIFFERENCE

INPUT endpoint l; maximum time T; constant α; integers $m \geq 3$, $N \geq 1$.

OUTPUT approximations $w_{i,j}$ to $u(x_i, t_j)$ for each $i = 1, \ldots, m-1$ and $j = 1, \ldots, N$.

Step 1 Set $h = l/m$;
$k = T/N$;
$\lambda = \alpha^2 k/h^2$.

Step 2 For $i = 1, \ldots, m-1$ set $w_i = f(ih)$. (*Initial values.*)

(*Steps* 3–11 *solve tridiagonal linear system by Algorithm* 6.7.)

Step 3 Set $l_1 = 1 + 2\lambda$;
$u_1 = -\lambda/l_1$.

Step 4 For $i = 2, \ldots, m-2$ set $l_i = 1 + 2\lambda + \lambda u_{i-1}$;
$u_i = -\lambda/l_i$.

Algorithm 12.2: HEAT EQUATION BACKWARD-DIFFERENCE

Step 5 Set $l_{m-1} = 1 + 2\lambda + \lambda u_{m-2}$.
Step 6 For $j = 1, \ldots, N$ do Steps 7–11.
  Step 7 Set $t = jk$; (*Current* t_j.)
    $z_1 = w_1 / l_1$.
  Step 8 For $i = 2, \ldots, m-1$ set $z_i = (w_i + \lambda z_{i-1})/l_i$.
  Step 9 Set $w_{m-1} = z_{m-1}$.
  Step 10 For $i = m-2, \ldots, 1$ set $w_i = z_i - u_i w_{i+1}$.
  Step 11 OUTPUT (t); (*Note:* $t = t_j$.)
    For $i = 1, \ldots, m-1$ set $x = ih$;
      OUTPUT (x, w_i). (*Note*: $w_i = w_{i,j}$.)
Step 12 STOP. (*The procedure is complete*.)

FORWARD DIFFERENCE

The Forward-Difference method involves the mesh points (x_{i-1}, t_{j-1}), (x_i, t_{j-1}), and (x_{i+1}, t_{j-1}) to approximate the value at (x_i, t_j), as illustrated in Figure 12.10. This method has stability problems, so the Backward-Difference method is preferred.

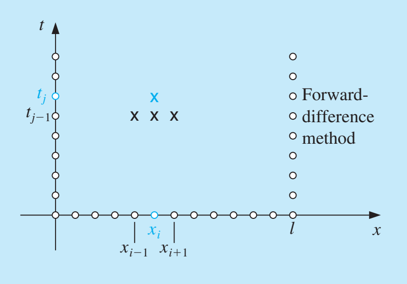

Figure: Figure 12.10

CRANK-NICLOSON

The Crank-Nicolson represents an averaging of the backward-difference method and forward-difference method involving the mesh points (x_{i-1}, t_j), (x_i, t_j), (x_{i+1}, t_j), (x_{i-1}, t_{j+1}),and (x_{i+1}, t_{j+1}) to approximate the value at (x_i, t_{j+1}), as illustrated in Figure 12.11.

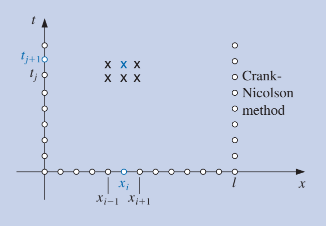

Figure: Figure 12.11

Algorithm 12.3: CRANK-NICOLSON METHOD

To approximate the solution to the parabolic partial differential equation

$$\frac{\partial u}{\partial t}(x,t) - \alpha^2 \frac{\partial^2 u}{\partial x^2}(x,t) = 0, \quad 0 < x < l, \quad 0 < t < T,$$

subject to the boundary conditions

$$u(0,t) = u(l,t) = 0, \quad 0 < t < T,$$

and the initial conditions

$$u(x,0) = f(x), \quad 0 \le x \le l:$$

Algorithm 12.3: CRANK-NICOLSON METHOD

INPUT endpoint l; maximum time T; constant α; integers $m \geq 3$, $N \geq 1$.

OUTPUT approximations $w_{i,j}$ to $u(x_i, t_j)$ for each $i = 1, \ldots, m-1$ and $j = 1, \ldots, N$.

Step 1 Set $h = l/m$;
$k = T/N$;
$\lambda = \alpha^2 k/h^2$;
$w_m = 0$.

Step 2 For $i = 1, \ldots, m-1$ set $w_i = f(ih)$. (*Initial values.*)

(*Steps* 3–11 *solve a tridiagonal linear system using Algorithm* 6.7.)

Step 3 Set $l_1 = 1 + \lambda$;
$u_1 = -\lambda/(2l_1)$.

Step 4 For $i = 2, \ldots, m-2$ set $l_i = 1 + \lambda + \lambda u_{i-1}/2$;
$u_i = -\lambda/(2l_i)$.

Algorithm 12.3: CRANK-NICOLSON METHOD

Step 5 Set $l_{m-1} = 1 + \lambda + \lambda u_{m-2}/2$.
Step 6 For $j = 1, \ldots, N$ do Steps 7–11.
    Step 7 Set $t = jk$;   (*Current* t_j.)

$$z_1 = \left[(1-\lambda)w_1 + \tfrac{\lambda}{2}w_2\right] \Big/ l_1.$$

    Step 8 For $i = 2, \ldots, m-1$ set

$$z_i = \left[(1-\lambda)w_i + \tfrac{\lambda}{2}(w_{i+1} + w_{i-1} + z_{i-1})\right] \Big/ l_i.$$

    Step 9 Set $w_{m-1} = z_{m-1}$.
    Step 10 For $i = m-2, \ldots, 1$ set $w_i = z_i - u_i w_{i+1}$.
    Step 11 OUTPUT (t);   (*Note:* $t = t_j$.)
        For $i = 1, \ldots, m-1$ set $x = ih$;
            OUTPUT (x, w_i).   (*Note:* $w_i = w_{i,j}$.)
Step 12 STOP.   (*The procedure is complete.*)

Algorithm 12.4: WAVE EQUATION FINITE-DIFFERENCE

To approximate the solution to the wave equation

$$\frac{\partial^2 u}{\partial t^2}(x,t) - \alpha^2 \frac{\partial^2 u}{\partial x^2}(x,t) = 0, \quad 0 < x < l, \quad 0 < t < T,$$

subject to the boundary conditions

$$u(0,t) = u(l,t) = 0, \quad 0 < t < T,$$

and the initial conditions

$$u(x,0) = f(x), \quad \text{and} \quad \frac{\partial u}{\partial t}(x,0) = g(x), \quad \text{for} \quad 0 \le x \le l,$$

INPUT endpoint l; maximum time T; constant α; integers $m \ge 2$, $N \ge 2$.

Algorithm 12.4: WAVE EQUATION FINITE-DIFFERENCE

OUTPUT approximations $w_{i,j}$ to $u(x_i, t_j)$ for each $i = 0, \ldots, m$ and $j = 0, \ldots, N$.

Step 1 Set $h = l/m$;
$k = T/N$;
$\lambda = k\alpha/h$.

Step 2 For $j = 1, \ldots, N$ set $w_{0,j} = 0$;
$w_{m,j} = 0$;

Step 3 Set $w_{0,0} = f(0)$;
$w_{m,0} = f(l)$.

Step 4 For $i = 1, \ldots, m-1$ (*Initialize for $t = 0$ and $t = k$.*)
set $w_{i,0} = f(ih)$;
$w_{i,1} = (1-\lambda^2)f(ih) + \frac{\lambda^2}{2}[f((i+1)h)$
$+f((i-1)h)] + kg(ih)$.

Algorithm 12.4: WAVE EQUATION FINITE-DIFFERENCE

Step 5 For $j = 1, \ldots, N-1$ (*Perform matrix multiplication.*)
for $i = 1, \ldots, m-1$
set $w_{i,j+1} = 2(1-\lambda^2)w_{i,j} + \lambda^2(w_{i+1,j} + w_{i-1,j}) - w_{i,j-1}$.

Step 6 For $j = 0, \ldots, N$
set $t = jk$;
for $i = 0, \ldots, m$
set $x = ih$;
OUTPUT $(x, t, w_{i,j})$.

Step 7 STOP. (*The procedure is complete.*)

DEFINING THE ELEMENTS

The first step is to divide the region into a finite number of sections, or elements, of a regular shape, either rectangles or triangles. (Figure 12.14.) The set of functions used for approximation is generally a set of piecewise polynomials of fixed degree in x and y, and the approximation requires that the polynomials be pieced together in such a manner that the resulting function is continuous with an integrable or continuous first or second derivative on the entire region.

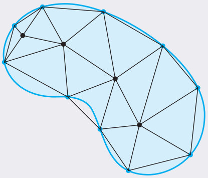

Figure: Figure 12.14

Algorithm 12.5: FINITE-ELEMENT METHOD

To approximate the solution to the partial differential equation

$$\frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(q(x,y)\frac{\partial u}{\partial y}\right)+r(x,y)u=f(x,y), \quad (x,y)\in D$$

subject to the boundary conditions

$$u(x,y)=g(x,y), \quad (x,y)\in \mathcal{S}_1$$

and

$$p(x,y)\frac{\partial u}{\partial x}(x,y)\cos\theta_1 \quad +q(x,y)\tfrac{\partial u}{\partial y}(x,y)\cos\theta_2+g_1(x,y)u(x,y) = g_2(x,y), \quad (x,y)\in \mathcal{S}_2,$$

where $\mathcal{S}_1\cup\mathcal{S}_2$ is the boundary of D, and θ_1 and θ_2 are the direction angles of the normal to the boundary:

Algorithm 12.5: FINITE-ELEMENT METHOD

Step 0 Divide the region D into triangles $T_1, \ldots, T_M$ such that:

$T_1, \ldots, T_K$ are the triangles with no edges on $\mathcal{S}_1$ or $\mathcal{S}_2$;

(*Note:* $K = 0$ *implies that no triangle is interior to* D.)

$T_{K+1}, \ldots, T_N$ are the triangles with at least one edge on $\mathcal{S}_2$;

$T_{N+1}, \ldots, T_M$ are the remaining triangles.

(*Note:* $M = N$ *implies that all triangles have edges on* $\mathcal{S}_2$.)

Label the three vertices of the triangle T_i by

$\left(x_1^{(i)}, y_1^{(i)}\right)$, $\left(x_2^{(i)}, y_2^{(i)}\right)$, and $\left(x_3^{(i)}, y_3^{(i)}\right)$.

Label the nodes (vertices) $E_1, \ldots, E_m$ where

$E_1, \ldots, E_n$ are in $D \cup \mathcal{S}_2$ and $E_{n+1}, \ldots, E_m$ are on $\mathcal{S}_1$.

(*Note:* $n = m$ *implies that* $\mathcal{S}_1$ *contains no nodes*.)

Algorithm 12.5: FINITE-ELEMENT METHOD

INPUT integers K, N, M, n, m; vertices $\left(x_1^{(i)}, y_1^{(i)}\right), \left(x_2^{(i)}, y_2^{(i)}\right), \left(x_3^{(i)}, y_3^{(i)}\right)$ for each $i = 1, \ldots, M$; nodes E_j for each $j = 1, \ldots, m$.
(*Note: All that is needed is a means of corresponding a vertex* $\left(x_k^{(i)}, y_k^{(i)}\right)$ *to a node* $E_j = (x_j, y_j)$.)

OUTPUT constants $\gamma_1, \ldots, \gamma_m$; $a_j^{(i)}, b_j^{(i)}, c_j^{(i)}$ for each $j = 1, 2, 3$ and $i = 1, \ldots, M$.

Step 1 For $l = n + 1, \ldots, m$ set $\gamma_l = g(x_l, y_l)$. (*Note:* $E_l = (x_l, y_l)$.)
Step 2 For $i = 1, \ldots, n$
set $\beta_i = 0$;
for $j = 1, \ldots, n$ set $\alpha_{i,j} = 0$.

Algorithm 12.5: FINITE-ELEMENT METHOD

Step 3 For $i = 1, \ldots, M$

$$\text{set } \Delta_i = \det \begin{vmatrix} 1 & x_1^{(i)} & y_1^{(i)} \\ 1 & x_2^{(i)} & y_2^{(i)} \\ 1 & x_3^{(i)} & y_3^{(i)} \end{vmatrix};$$

$$a_1^{(i)} = \frac{x_2^{(i)}y_3^{(i)} - y_2^{(i)}x_3^{(i)}}{\Delta_i}; \quad b_1^{(i)} = \frac{y_2^{(i)} - y_3^{(i)}}{\Delta_i}; \quad c_1^{(i)} = \frac{x_3^{(i)} - x_2^{(i)}}{\Delta_i};$$

$$a_2^{(i)} = \frac{x_3^{(i)}y_1^{(i)} - y_3^{(i)}x_1^{(i)}}{\Delta_i}; \quad b_2^{(i)} = \frac{y_3^{(i)} - y_1^{(i)}}{\Delta_i}; \quad c_2^{(i)} = \frac{x_1^{(i)} - x_3^{(i)}}{\Delta_i};$$

$$a_3^{(i)} = \frac{x_1^{(i)}y_2^{(i)} - y_1^{(i)}x_2^{(i)}}{\Delta_i}; \quad b_3^{(i)} = \frac{y_1^{(i)} - y_2^{(i)}}{\Delta_i}; \quad c_3^{(i)} = \frac{x_2^{(i)} - x_1^{(i)}}{\Delta_i};$$

for $j = 1, 2, 3$

define $N_j^{(i)}(x, y) = a_j^{(i)} + b_j^{(i)}x + c_j^{(i)}y$.

Algorithm 12.5: FINITE-ELEMENT METHOD

Step 4 For $i = 1, \ldots, M$ (*The integrals in Steps 4 & 5 can be evaluated using numerical integration.*)

for $j = 1, 2, 3$

for $k = 1, \ldots, j$ (*Compute all double integrals over.*) *the triangles.)*

set $z_{j,k}^{(i)} = b_j^{(i)} b_k^{(i)} \iint_{T_i} p(x,y)\,dx\,dy$
$+c_j^{(i)} c_k^{(i)} \iint_{T_i} q(x,y)\,dx\,dy$
$-\iint_{T_i} r(x,y) N_j^{(i)}(x,y) N_k^{(i)}(x,y)\,dx\,dy$;

set $H_j^{(i)} = -\iint_{T_i} f(x,y) N_j^{(i)}(x,y)\,dx\,dy$.

Step 5 For $i = K+1, \ldots, N$ (*Compute all line integrals.*)

for $j = 1, 2, 3$

for $k = 1, \ldots, j$

set $J_{j,k}^{(i)} = \int_{\mathcal{S}_2} g_1(x,y) N_j^{(i)}(x,y) N_k^{(i)}(x,y)\,dS$;

set $I_j^{(i)} = \int_{\mathcal{S}_2} g_2(x,y) N_j^{(i)}(x,y)\,dS$.

Algorithm 12.5: FINITE-ELEMENT METHOD

Step 6 For $i = 1, \ldots, M$ do Steps 7–12. (*Assembling the integrals over each triangle into the.*) *linear system.*)

Step 7 For $k = 1, 2, 3$ do Steps 8–12.

Step 8 Find l so that $E_l = \left(x_k^{(i)}, y_k^{(i)}\right)$.

Step 9 If $k > 1$ then for $j = 1, \ldots, k-1$ do Steps 10, 11.

Step 10 Find t so that $E_t = \left(x_j^{(i)}, y_j^{(i)}\right)$.

Step 11 If $l \leq n$ then

if $t \leq n$ then set $\alpha_{lt} = \alpha_{lt} + z_{k,j}^{(i)}$;

$\alpha_{tl} = \alpha_{tl} + z_{k,j}^{(i)}$

else set $\beta_l = \beta_l - \gamma_t z_{k,j}^{(i)}$

else

if $t \leq n$ then set $\beta_t = \beta_t - \gamma_l z_{k,j}^{(i)}$.

Algorithm 12.5: FINITE-ELEMENT METHOD

Step 12 If $l \leq n$ then set $\alpha_{ll} = \alpha_{ll} + z_{k,k}^{(i)}$;

$$\beta_l = \beta_l + H_k^{(i)}.$$

Step 13 For $i = K+1, \ldots, N$ do Steps 14–19. (*Assembling the line integrals into.*) *into the linear system.*)

Step 14 For $k = 1, 2, 3$ do Steps 15–19.

Step 15 Find l so that $E_l = \left(x_k^{(i)}, y_k^{(i)}\right)$.

Step 16 If $k > 1$ then for $j = 1, \ldots, k-1$ do Steps 17, 18.

Step 17 Find t so that $E_t = \left(x_j^{(i)}, y_j^{(i)}\right)$.

Algorithm 12.5: FINITE-ELEMENT METHOD

Step 18 If $l \leq n$ then

if $t \leq n$ then set $\alpha_{lt} = \alpha_{lt} + J_{k,j}^{(i)}$;

$\alpha_{tl} = \alpha_{tl} + J_{k,j}^{(i)}$

else set $\beta_l = \beta_l - \gamma_t J_{k,j}^{(i)}$

else

if $t \leq n$ then set $\beta_t = \beta_t - \gamma_l J_{k,j}^{(i)}$.

Step 19 If $l \leq n$ then set $\alpha_{ll} = \alpha_{ll} + J_{k,k}^{(i)}$;

$\beta_l = \beta_l + I_k^{(i)}$.

Step 20 Solve the linear system $A\mathbf{c} = \mathbf{b}$ where $A = (\alpha_{l,t})$, $\mathbf{b} = (\beta_l)$ and $\mathbf{c} = (\gamma_t)$ for $1 \leq l \leq n$ and $1 \leq t \leq n$.

Algorithm 12.5: FINITE-ELEMENT METHOD

Step 21 OUTPUT $(\gamma_1, \ldots, \gamma_m)$.

(*For each* $k = 1, \ldots, m$ *let* $\phi_k = N_j^{(i)}$ *on* T_i *if* $E_k = \left(x_j^{(i)}, y_j^{(i)}\right)$.
Then $\phi(x, y) = \sum_{k=1}^{m} \gamma_k \phi_k(x, y)$ *approximates*
$u(x, y)$ on $D \cup \mathcal{S}_1 \cup \mathcal{S}_2$.)

Step 22 For $i = 1, \ldots, M$

for $j = 1, 2, 3$ OUTPUT $\left(a_j^{(i)}, b_j^{(i)}, c_j^{(i)}\right)$.

Step 23 STOP. (*The procedure is complete*.)