4 scanner online

Midterm examination Computational Geophysics
June 27, 2012, 13:15-15:00,
Room BBL 065

Solve the following problems 3.2-3.3, 5.10, 5.11 in the printout from Chapters 3 and 5 of the coursenotes.
Give clear explanations with your derivations and argumentation in answering the questions.

In finite difference methods the domain V is discretized with a grid of N nodal points $\mathbf{x}_I = (x_I, y_I)$ and the unknown function $u(\mathbf{x})$ is replaced by a vector of nodal point values,

$$\mathbf{U} = (u(\mathbf{x}_1), u(\mathbf{x}_2), \cdots u(\mathbf{x}_N))^T \tag{3.5}$$

In the previous chapter we have seen how, for a 1-D problem, the PDE can be discretized using a central difference approximation for the second derivative. This approach is extended for multi-dimensional problems in the next section.

3.2 A central difference method

We shall use a rectangular geometry of the domain V and we define an equidistant 2-D grid of nodal points by,

$$\mathbf{x}_{ij} = (x_i, y_j) = (x_0 + i \times h, y_0 + j \times h) \tag{3.6}$$

problem 3.2. *Derive the following difference approximation of the 2-D Laplace operator,*

$$\begin{aligned} \nabla^2 u(\mathbf{x}_{ij}) &\approx D_h^2 u(\mathbf{x}_{ij}) \\ &= \frac{(u(x_i + h, y_j) + u(x_i - h, y_j) - 4u(x_i, y_j) + u(x_i, y_j + h) + u(x_i, y_j - h))}{h^2} \\ &= \frac{(u(x_{i+1}, y_j) + u(x_{i-1}, y_j) - 4u(x_i, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}))}{h^2} \end{aligned} \tag{3.7}$$

Show that the local truncation error in the discretized Laplace operator defined as,

$$E = D_h^2 u(\mathbf{x}_{ij}) - \nabla^2 u(\mathbf{x}_{ij}) \tag{3.8}$$

is of second order in the grid spacing h, i.e. $E = O(h^2)$.
Hint: *expand the functions in the difference formula in a Taylor series in h in the neighborhood of the grid point $\mathbf{x}_{ij}$. Do this separately for both x and y dependence.*

The degrees of freedom of the discretized problem have been organized in an N vector $\mathbf{U} \in \mathbb{R}^N$ in (3.5) and the sequence of the vector components depends on the nodalpoint numbering of the finite difference mesh, mapping the grid indices i, j (row, column) onto the index I of the N-vector $\mathbf{U}$. Assuming n_{row} rows and n_{col} columns in a rectangular grid, a straightforward mapping is obtained by a column wise numbering of the nodal points $I = (j_{col} - 1)n_{row} + i_{row}$,

$$U_I = u(\mathbf{x}_I) = u(x_{j_{col}}, y_{i_{row}}),\ j_{col} = 1, \ldots, n_{col},\ i_{row} = 1, \ldots, n_{row} \tag{3.9}$$

In case of a boundary value problem with prescribed values of the unknown potential on the complete boundary, a Dirichlet type boundary condition (3.2), the described discretization and nodal point numbering result in a system of linear algebraic equations for the internal nodal point values of the potential function.

The bandwidth w of the matrix $\mathbf{S}$ is defined by the location of non-zero diagonals in the upper and lower triangle matrix, excluding the main diagonal,

$$w = max|I - J|, \quad S_{IJ} \neq 0 \tag{3.10}$$

According to this definition a diagonal matrix has a zero bandwidth and tri-diagonal matrix has a bandwidth of one. From this we find that the smallest bandwidth is obtained by defining the grid columns in the direction of the smallest dimension of the rectangular domain. Efficient algorithms are available for the solution of systems of linear algebraic equations that are based on a so called bandmatrix structure where only matrix elements within the bandwidth defined in (3.10) are stored in computer memory. Minimizing the bandwith of the matrix will result in minimizing the computer requirements (memory, compute time) when using such bandmatrix solvers.

problem 3.3. *Show that the operator D_h^2 applied to the nodal point values of a uniform grid produces a system of linear algebraic equations for a discrete approximation of (3.1). Consider the special case of a rectangular grid with two rows of internal gridpoints and apply a (grid) columnwise numbering of the degrees of freedom of the problem as in (3.9). What is the structure of the matrix* $\mathbf{S}$ *of the resulting system?*

3.3 A difference formula for variable grid spacing and variable coefficient

The difference formula for the Laplace operator (3.7) is derived for the special case of a uniform coefficient $c(\mathbf{x})$. Besides this the grid used in section 3.2 is equidistant. Here we introduce the more general case with variable coefficient and apply a so called structured grid, defined by the product of two 1-D grids with variable nodal point spacing in both coordinate directions. This allows local mesh refinement. We consider a 2-D rectangular domain subdivided in rectangular grid cells spanned by the nodalpoints, illustrated in Fig. 3.1. Similar as in 2.2, but now for a 2-D problem, we integrate the PDE (3.1) over a small area, a finite volume, surrounding a single gridpoint P_0, illustrated in Fig. 3.1.

Relative coordinates of neighboring gridpoints of P_0 are parameterized as follows,

$$\begin{aligned} P_0 &= (0,0) \\ P_1 &= (s_1 h, 0) \\ P_2 &= (0, s_2 h) \\ P_3 &= (-s_3 h, 0) \\ P_4 &= (0, -s_4 h), \quad 0 < s_K \leq 1, \quad K = 1, \ldots, 4 \end{aligned} \tag{3.11}$$

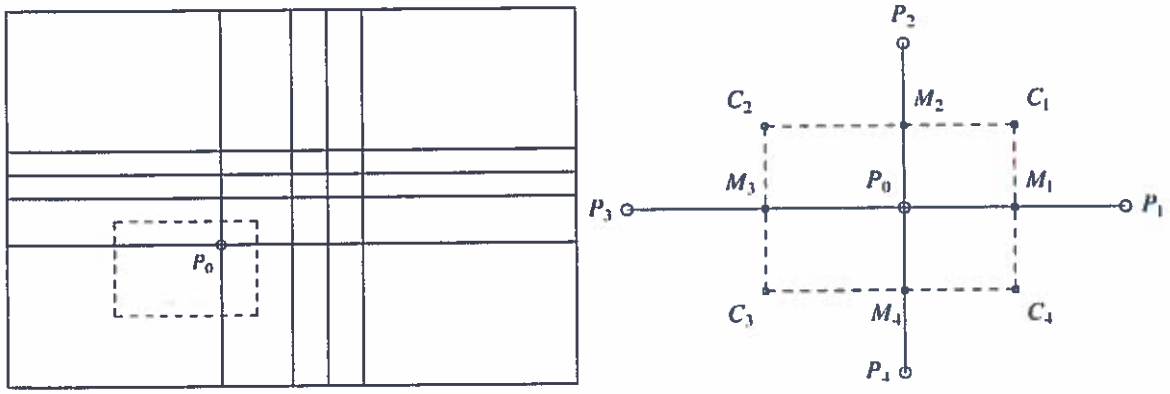

Figure 3.1: *Left: rectangular computational domain showing local mesh refinement and a single (dashed) finite volume. Right: zoom-in finite volume with five grid points in a mesh with variable grid point spacing. $M_K, K = 1, \ldots, 4$ are midpoints positioned halfway two neighboring gridpoints P_0, P_K. A difference equation is derived by integration over the rectangle spanned by the corner points $C_1, \ldots, C_4$.*

The PDE (3.1) is integrated over the rectangular area V spanned by the corner points $C_1, \ldots, C_4$ in Fig.3.1, that are the centre points of the neighboring grid cells.

$$I = \int_V \nabla \cdot c(\mathbf{x}) \nabla u \, dV = \int_{\partial V} c(\mathbf{x}) \partial_i u \, n_i \, dA = -\int_V f(\mathbf{x}) dV \tag{3.12}$$

5.2 Structure of the coefficient matrices

From the definition (5.9) and (5.10) of the coefficient matrices of (5.6) we can immediately deduce the symmetry property,

$$M_{IJ} = M_{JI},\ S_{IJ} = S_{JI} \tag{5.14}$$

If we choose for the basis functions the piecewise linear (hat) functions as displayed in Fig. 5.1, these matrices are also tri-diagonal i.e. there bandwidth is one.

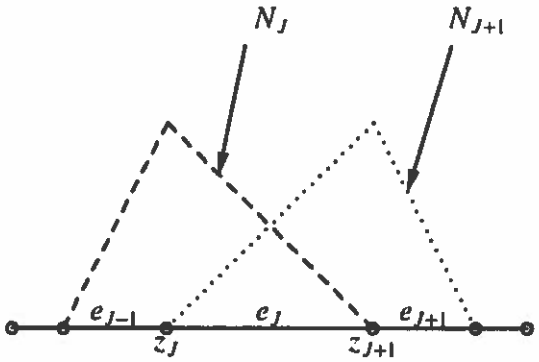

Figure 5.1: *Piecewise linear basis functions N_J (dashed) and N_{J+1} (dotted) on a 1-D grid. The support of basis function N_J is $e_{J-1} \cup e_J$.*

For multi-dimensional problems (2-D,3-D) we find in a similar way that the finite element matrices are sparse, i.e. most of the matrix elements are zero. This follows directly from the expression for the matrix coefficients. For the heatcapacity matrix **M**, also denoted as the mass matrix in the literature, we have for example,

$$M_{IJ} = \int_V N_I N_J \, dz \tag{5.15}$$

It follows that the matrix element is zero if $S(N_I) \cap S(N_J) = \emptyset$, where $S(N_I)$ is the support of the basis function N_I. The basis function is defined as a piecewise Lagrange interpolating polynomial on the elements that contain nodal $\mathbf{x}_I$, and zero elsewhere. This means that N_I differs from zero only in the direct neighborhood of nodal point $\mathbf{x}_I$ and most of all the possible productfunctions $N_I N_J$ are zero, resulting in a corresponding zero matrix element M_{IJ}. From (5.10) it follows that the stiffness matrix **S** has the same sparsity structure as the mass matrix **M**. This situation resembles the finite difference methods introduced earlier, where only combinations of neighboring nodal points, connected by a 'difference molecule' resulted in non-zero matrix contributions.

problem 5.3. *Show from the above that both matrices* **S** *and* **M** *are tri-diagonal for 1-D problems.*

5.3 Computation of the matrix elements

The integrals defining the matrix and right hand side vector elements can be split in a sum of contributions from the individual finite elements e_J, in this 1-D case $e_J = [z_J, z_{J+1}]$. In this context the complete matrices in (5.9), (5.10) are known as global matrices and the contribution from a single element is known as an element matrix. In finite element computations the global matrices are computed in an 'assembly procedure' where the coefficients of the element matrices are added to the corresponding coefficients of the global matrix in

a loop over elements. The righthand side vector **R** is assembled in a similar way in a loop over elements. In section 5.4 the implementation of the assembly proces will be treated in more detail.

5.3.1 The mass matrix M

The mass (or heat capacity matrix) appears in the 'inertial' term of the differential equations of the finite element solution (5.7). The matrix is defined as the sum contribution of the $N-1$ elements,

$$M_{JL} = \int_0^{z_{max}} N_L N_J \, dz = \sum_{K=1}^{N-1} \int_{z_K}^{z_{K+1}} N_L N_J \, dz = \sum_{K=1}^{N-1} M_{JL}^{(K)} \tag{5.16}$$

The mass matrix for element $e_K = [z_K, z_{K+1}]$ is defined as,

$$M_{JL}^{(K)} = \int_{z_K}^{z_{K+1}} N_L N_J \, dz \tag{5.17}$$

For the piecewise linear basis functions considered here we see from Fig.5.1 that only the following four coefficients of the element matrix are non-zero, $M_{KK}, M_{KK+1}, M_{K+1K}, M_{K+1K+1}$. Using the local numbering of the nodal points and basis functions on e_K and $h = z_2 - z_1$ we obtain,

$$M_{11} = \int_{z_1}^{z_2} N_1 N_1 \, dz = \int_{z_1}^{z_2} \left(1 - \frac{z - z_1}{h}\right)^2 dz = \int_0^1 (1-\zeta)^2 \, h \, d\zeta = \frac{h}{3} \tag{5.18}$$

$$\begin{aligned} M_{12} = \int_{z_1}^{z_2} N_1 N_2 \, dz &= \int_{z_1}^{z_2} \left(1 - \frac{z - z_1}{h}\right)\left(\frac{z - z_1}{h}\right) dz \\ &= \int_0^1 (1-\zeta)\,\zeta h \, d\zeta = \frac{h}{6} \end{aligned} \tag{5.19}$$

Since $M_{11} = M_{22}$ and $M_{12} = M_{21}$ we find for the element matrix,

$$\mathbf{M}^{(K)} = \frac{h_K}{6} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \tag{5.20}$$

problem 5.4. *Derive for row number J of the global mass matrix,*

$$M_{JL} = \begin{cases} \frac{h_{J-1}}{6}, & L = J-1 \\ \frac{h_{J-1}+h_J}{3}, & L = J \\ \frac{h_J}{6}, & L = J+1 \end{cases} \tag{5.21}$$

A test for a software implementation of the element mass matrix can be devised from the following:
An innerproduct of two real valued functions f and g on the interval $[0, z_{max}]$ is defined by,

$$(f \cdot g) = \int_0^{z_{max}} fg \, dz \tag{5.22}$$

Expansion in basis functions gives

$$(f^h \cdot g^h) = \int_0^{z_{max}} \left\{ \sum_I F_I N_I(z) \sum_J G_J N_J(z) \right\} dz = \sum_I \sum_J M_{IJ} F_I G_J = (\mathbf{MG} \cdot \mathbf{F}) \tag{5.23}$$

It follows that the innerproduct (5.22) is exactly represented by (5.23) for piecewise linear functions (including uniform functions), by using expression (5.20).

From (5.23) it follows that the mass matrix is positif definite,

$$(\mathbf{MX} \cdot \mathbf{X}) > 0, \ \mathbf{X} \neq \mathbf{0} \tag{5.24}$$

The lumped mass matrix
In applications the sparse mass matrix is often replaced by an approximating diagonal matrix the so called 'lumped' mass matrix, $\mathbf{M}^*$. For the general case in more dimensions we have,

$$M_{IJ} = \int_V N_I N_J \, dV = \int_V \Phi(\mathbf{x}) \, dV \tag{5.25}$$

When the basis functions N_K are piecewise Lagrange polynomials of degree p, the integrand $\Phi(\mathbf{x})$ is a piecewise polynomial of degree $2p$. This latter polynomial can be approximated in the usual way by interpolation,

$$\begin{aligned} M_{IJ} &= \int_V \Phi \, dV \\ &\approx M^*_{IJ} = \int_V \sum_K \Phi_K N_K(\mathbf{x}) \, dV = \sum_K N_I(\mathbf{x}_K) N_J(\mathbf{x}_K) \int_V N_K(\mathbf{x}) \, dV \\ &= \sum_K \delta_{IK}\delta_{JK} \int_V N_K(\mathbf{x}) \, dV = \delta_{IJ} \int_V N_I(\mathbf{x}) \, dV \end{aligned} \tag{5.26}$$

problem 5.5. *Derive the following expression for the lumped version of the element mass matrix from the mass matrix in (5.20).*
Solution:

$$M^{*(K)}_{IJ} = \frac{h_K}{2}\delta_{IJ} \tag{5.27}$$

Verify that the sum of the matrix elements in a row (row sum) is conserved in this matrix lumping procedure.
Hint: make use of the property of the basis functions,

$$\sum_L N_L(\mathbf{x}) = 1 \tag{5.28}$$

Derive for the global lumped mass matrix,

$$M^*_{IJ} = \left(\frac{h_{I-1} + h_I}{2}\right)\delta_{IJ} \tag{5.29}$$

5.3.2 The stiffness matrix S

The stifness matrix which appears in the diffusion term in the matrix equation (5.7) is defined as,

$$S_{IL} = \int_0^{z_{max}} \kappa \frac{\partial N_L}{\partial z}\frac{\partial N_I}{\partial z} \, dz \tag{5.30}$$

The derivatives have the same support as the basis functions,

$$\frac{\partial N_I}{\partial z} = \begin{cases} \frac{1}{h_{I-1}}, & z \in e_{I-1} \\ -\frac{1}{h_I}, & z \in e_I \\ 0, & z \ni e_I \cup e_{I-1} \end{cases} \tag{5.31}$$

We further assume here that the diffusion coefficient is piecewise constant. The element matrix becomes,

$$S^{(K)}_{IL} = \kappa_K \int_{z_K}^{z_{K+1}} \frac{\partial N_I}{\partial z}\frac{\partial N_L}{\partial z} \, dz \tag{5.32}$$

and substituting (5.31) we get for the diagonal terms,

$$S_{11} = \kappa_K \int_{z_K}^{z_{K+1}} \left(\frac{\partial N_1}{\partial z}\right)^2 dz = \frac{\kappa_K}{h_K}, \quad S_{22} = S_{11} \tag{5.33}$$

For the off-diagonal terms we get,

$$S_{12} = \kappa_K \int_{z_K}^{z_{K+1}} \frac{\partial N_1}{\partial z}\frac{\partial N_2}{\partial z}\, dz = -\frac{\kappa_K}{h_K}, \quad S_{21} = S_{12} \tag{5.34}$$

$$\mathbf{S}^{(K)} = \frac{\kappa_K}{h_K}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \tag{5.35}$$

5.3.3 The righthand side vector R

Here we consider the contribution to the righthand side vector from the righthand side function f of the partial differential equation (5.1) defined as,

$$Q_I = \int_0^{z_{max}} f N_I \, dz \tag{5.36}$$

Corresponding contributions from boundary conditions are defined in (5.7). For a given function f the integral in (5.36) can be evaluated numerically. In a alternative procedure $f(z)$ is expanded in the same basis functions as the solution $T(z)$,

$$\begin{aligned} Q_I &\approx \int_0^{z_{max}} \sum_J F_J N_J(z) N_I(z)\, dz = \sum_J \int_0^{z_{max}} N_J(z) N_I(z)\, dz F_J \\ &= \sum_J M_{IJ} F_J, \quad F_J = f(z_J) \end{aligned} \tag{5.37}$$

where $\mathbf{M}$ is the mass matrix. This way the righthand side vector is computed by means of a matrix-vector multiplication of the mass matrix and the nodal point vector of the righthanside function of the PDE (5.1). In software implementations the righthandside vector is assembled element-wise by summing element vectors $\mathbf{Q}^K$ in a program loop over elements e_K,

$$Q_I = \sum_K Q_I^K, \quad Q_I^K = \sum_J M_{IJ}^K F_J^K, \quad \mathbf{Q} = \mathbf{MF} \tag{5.38}$$

$$\begin{pmatrix} Q_1^K \\ Q_2^K \end{pmatrix} = \frac{h_K}{6}\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} F_1^K \\ F_2^K \end{pmatrix} \tag{5.39}$$

problem 5.6. *How can the vector* $\mathbf{Q}$ *be defined for righthand side function* $f(z) = c\delta(z - z_s)$*, corresponding to a pointsource concentrated in the sourcepoint* z_s*. Where* c *is a constant and* δ *is the Dirac delta function. Show that the number of non-zero vector elements of the right-hand side vector for this case is two.*

5.4 Implementation of the assembly proces

The matrix and right hand side vectors of the discretized equations are computed by summing element contributions. This procedure is known as matrix and vector assembly respectively. As an example of the general procedure we describe here an implementation for a 1-D problem that can be generalized for multi-dimensional problems. A so called location matrix for the discretized domain is used in the implementation. This is an $M \times 2$

matrix, M the number of elements in the 1-D grid. For each element, the corresponding row of the location matrix contains the global sequence numbers of the two degrees of freedom corresponding to the element. In the assembly proces these sequence numbers are used as pointers to the global matrix coefficients where the coefficients of the element matrix are added. Fig. 5.2 shows a structure diagram of an algorithm for the computation of the location matrix in a program array `kelem`.

The location matrix is used in the assembly process. An algorithm for the assembly of the stiffness matrix and righthand side vector is described in the structure diagram of Fig. 5.3. In this scheme the 2×2 element matrices are computed in an element routine `elem` which contains an implementation of the expression (5.35). The actual summation of the element matrix coefficient to the global matrix coefficients is performed in a procedure `addmat`. In this procedure the special (band) structure of the sparse global matrix can be exploited to obtain effcient memory storage of the matrix array. The element righthand side vector is computed in a routine `elrhs` which contains an implementation of (5.39).

problem 5.7. *Verify that the assembly proces for the element matrices (5.35) results in the same matrix as obtained with the finite volume method in section Chapter 2. Hint: compare a single matrix row for both cases.*

```
nelem              - number of elements
kelem(1:nelem,1:2) - location matrix
bc1, bcn           - type boundary conditions in nodalpoints 1 resp. n
                     (1-essential | 2 natural)
```

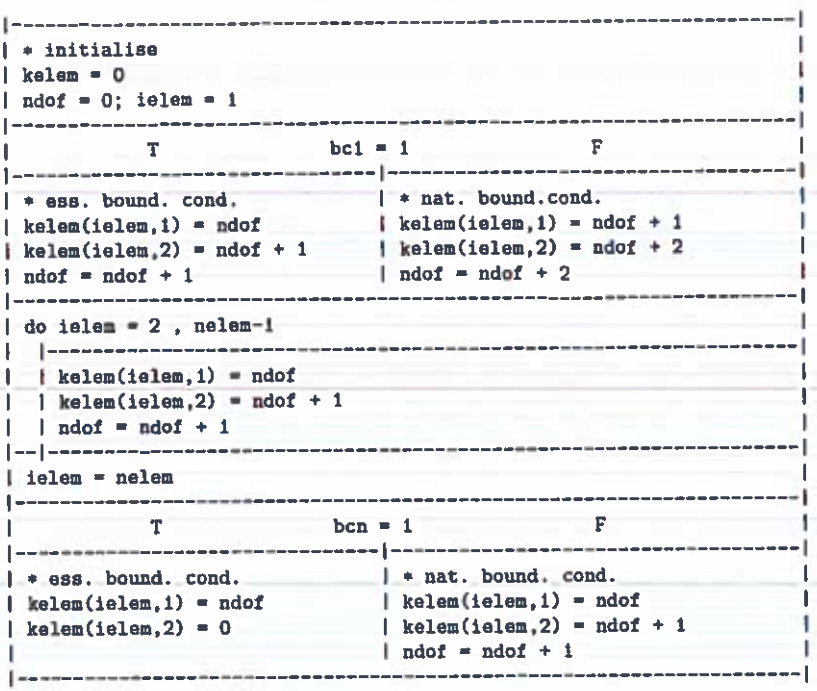

Figure 5.2: *Structure diagram of an algorithm for filling the location matrix in an array* `kelem`. *Note the different treatment of essential and natural boundary conditions.*

```
nelem    - number of elements
coord    - nodalpoint coordinates
kelem    - location matrix
glomat   - global matrix
glovec   - global right hand side vector
elmat    - element matrix
elvec    - element vector
```

```
|-------------------------------------------------------------------|
| do ielem = 1, nelem                                               |
|  |----------------------------------------------------------------|
|  | idof1 = kelem(ielem,1)                                         |
|  | idof2 = kelem(ielem,2)                                         |
|  | * element matrix                                               |
|  | call elem(ielem,coord,elmat)                                   |
|  | * element vector                                               |
|  | call elrhs(ielem,coord,elvec)                                  |
|  |----------------------------------------------------------------|
|  |                  T                 idof1 > 0      F           |
|  |----------------------------------------------------------------|
|  | * diagonal element glob. matr.          | * essent. bound.cond.|
|  | call addmat(idof1,idof1,elmat(1,1),glomat)|                    |
|  | * r.h.s. vector                         |                      |
|  | glovec(idof1)=glovec(idof1)+elvec(1)    |                      |
|  |----------------------------------------------------------------|
|  |                  T                 idof2 > 0      F           |
|  |----------------------------------------------------------------|
|  | * diagonaal element glob. matr.         | * essent. bound.cond.|
|  | call addmat(idof2,idof2,elmat(2,2),glomat)|                    |
|  | * r.h.s. vector                         |                      |
|  | glovec(idof2)=glovec(idof2)+elvec(2)    |                      |
|  |----------------------------------------------------------------|
|  | * element outside main diagonal                                |
|  |----------------------------------------------------------------|
|  |                  T            idof1*idof2 != 0   F           |
|  |----------------------------------------------------------------|
|  | * outside main diagaonal                  | * compute r.h.s.   |
|  | * fill uppertriangle (symmetry)           | contrib.           |
|  | idofmn = min(idof1,idof2)                 | bound.cond.        |
|  | idofmx = max(idof1,idof2)                 |                    |
|  | call addmat(idofmn,idofmx,elmat(1,2),glomat) |                  |
|-------------------------------------------------------------------|
```

Figure 5.3: *Structure diagram of an algorithm for matrix/vector assembly using the location matrix in an array* `kelem`.

5.5 Solving equations with a tri-diagonal matrix

The solution of 1-D partial differential equations using discretization methods like the finite difference method or the finite elment method often requires the solution of linear algebraic equations with a tri-diagonal matrix. A simple recursion algorithm can be used to compute such solutions. To derive the algorithm the matrix is written as,

$$\begin{pmatrix} b_1 & c_1 & 0 & \dots & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & \dots & 0 & 0 & 0 \\ 0 & a_3 & b_3 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & b_{N-2} & c_{N-2} & 0 \\ 0 & 0 & 0 & \dots & a_{N-1} & b_{N-1} & c_{N-1} \\ 0 & 0 & 0 & \dots & 0 & a_N & b_N \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-2} \\ u_{N-1} \\ u_N \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{N-2} \\ d_{N-1} \\ d_N \end{pmatrix} \tag{5.40}$$

We apply Gauss elimination on this system of equations. Eliminate the unknown u_{i-1} from equation number i using equation number $i-1$, working downward and starting in the first

column of row number two. Arriving in row number i we have,

$$\alpha_{i-1}u_{i-1} + c_{i-1}u_i = s_{i-1} \tag{5.41}$$

$$a_i u_{i-1} + b_i u_i + c_i u_{i+1} = d_i \tag{5.42}$$

where $\alpha_1 = b_1,\ s_1 = d_1$. Elimination of u_{i-1} gives,

$$\left(b_i - \frac{a_i c_{i-1}}{\alpha i - 1}\right) u_i + c_i u_{i+1} = d_i - \frac{a_i s_{i-1}}{\alpha_{i-1}} \tag{5.43}$$

$$\alpha_i = b_i - \frac{a_i c_{i-1}}{\alpha_{i-1}},\quad s_i = d_i - \frac{a_i s_{i-1}}{\alpha_{i-1}},\quad i = 2, 3, \ldots \tag{5.44}$$

The Gauss elimination process results in a matrix **A** with two non-zero coefficients per matrix row, the diagional coefficient, $A_{ii} = \alpha_i$ and $A_{i\ i+1} = c_i$. After the Gauss elimination the solution vector is obtained from the matrix **A** by back substitution, applying $c_N = 0$,

$$u_N = \frac{s_N}{\alpha_N} \tag{5.45}$$

$$u_i = \frac{1}{\alpha_i}\left(s_i - c_i u_{i+1}\right),\ i = N-1,\ N-2,\ \ldots, 1 \tag{5.46}$$

This can be summarized in the following two-stage procedure. For given vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$:

- compute the vectors $\alpha_i, s_i,\ i = 1, 2, \ldots N$
- compute the solution vector **U** recursevely, $u_i,\ i = N,\ N-1,\ \ldots,\ 1$.

5.6 Implementation of boundary conditions

5.6.1 Natural boundary conditions

Here $\partial T/\partial z$ is given in one of the boundary points $z = 0, z = z_{max}$. This boundary condition can be substituted directly into (5.7). In case natural boundary conditions are given for both boundary points, we get a system of N equations in N unknowns, where N is the number of nodal points of the 1-D mesh.

The special case of a steady state problem with $\partial T/\partial t = 0$ must be considered separately here. We have seen in Chapter 3 that the potential problem with natural boundary condition on the whole boundary requires a compatibility condition for the boundary condition and that the solution, is non-unique (problem 3.13). This can be verified to hold also for the 1-D finite element case treated here, where the fem equations are (5.7),

$$\sum_J S_{IJ}T_J = Q_I + \left(\kappa\frac{\partial T}{\partial z}\right)_{z_{max}} \delta_{IN} - \left(\kappa\frac{\partial T}{\partial z}\right)_0 \delta_{I1},\ I = 1, 2, \ldots, N \tag{5.47}$$

As an example consider the special case with a single linear element,

$$\mathbf{ST} = \frac{\kappa}{h}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}\begin{pmatrix} T_1 \\ T_2 \end{pmatrix} = \begin{pmatrix} Q_1 \\ Q_2 \end{pmatrix} + \begin{pmatrix} -q_1 \\ q_2 \end{pmatrix} \tag{5.48}$$

where q_I are the heatflow density values in the nodal points. The element stiffness matrix is singular (det S= 0) and non-unique solutions exist only if a compatibility condition holds for the right-hand side of the equation. This condition is found by summation of the two equations.

$$0 = Q_1 + Q_2 + q_2 - q_1 \tag{5.49}$$

problem 5.8. *Give a physical interpretation of the compatibility condition (5.49).*

problem 5.9. *Verify that homogeneous natural boundary conditions are implicitly accounted for in the finite element method.*

5.6.2 Essential boundary conditions

In case essential boundary conditions apply in both boundary points the (1-D) problem has $N - 2$ degrees of freedom, corresponding to $N - 2$ internal nodal points. The solution can be written as,

$$\begin{aligned} T(z,t) &\approx \sum_{L=2}^{N-1} T_L(t)N_L(z) + T_1(t)N_1(z) + T_N(t)N_N(z) \\ &= T^*(z,t) + T_1(t)N_1(z) + T_N(t)N_N(z) \end{aligned} \tag{5.50}$$

The function T^* introduced in (5.50) is in a subspace $S_0 \subset S$ of functions with zero boundary values. Where S is the function space spanned by the basis functions $N_J(x), J = 1, \ldots N$. Apply the Galerkin principle to S_0 instead of S, i.e. let $I = 2, 3, \ldots, N-1$ in the testfunctions. In that case the boundary term in the Galerkin-finite element equations is zero. The terms in T_1 and T_N in the equation (5.7) contain only known quantities,

$$\sum_{J=1}^{N} M_{IJ}\frac{\partial T_J}{\partial t} + \sum_{J=1}^{N} S_{IJ}T_J = Q_I,\ I = 2, 3, \ldots,\ N-1 \tag{5.51}$$

Therefore these terms contribute to the righthand side vector. This is illustrated in the following example for a steady-state problem,

$$\mathbf{SU} = \mathbf{Q} \tag{5.52}$$

or writing the matrix vector product,

$$\sum_{J=1}^{N} S_{IJ}U_J = Q_I,\ I = 2, \ldots,\ N-1 \tag{5.53}$$

$$\sum_{J=2}^{N-1} S_{IJ}U_J = Q_I - S_{I1}U_1 - S_{IN}U_N,\ I = 2, \ldots,\ N-1 \tag{5.54}$$

for given values of $U_1,\ U_N$. The matrix equation is reduced in size and the righthand side vector modified to,

$$\mathbf{R} = \mathbf{Q} - U_1\mathbf{S_1} - U_N\mathbf{S_N} \tag{5.55}$$

where $\mathbf{S_1}$ and $\mathbf{S_N}$ are the first and last column vectors of the stiffness matrix $\mathbf{S}$

5.7 Steady state problems

An important special case of the equations described in the previous sections occurs if the solution is independent of time. As an example we describe a case with homogeneous essential boundary condition for $z = 0$, $(T)_0 = 0$. For the other boundary point we consider two possibilities,

1. $(T)_{z_{max}} = T_m$ *(essential boundary condition)* (5.56)

2. $(\kappa\partial T/\partial z)_{z_{max}} = \dfrac{q_m}{\rho c_P} = Q_m$ *(natural boundary condition)* (5.57)

This problem corresponds to a steady state heatconduction problem for a laterally homogeneous layer with vertically varying thermal diffusivity κ, prescribed temperature on the surface $z = 0$ and for z_{max} either a prescribed temperature or a a prescribed heatflow. The equations for both cases with explicit right hand side contributions of the boundary conditions are,

1. $$\sum_{J=2}^{N-1} S_{IJ}T_J = F_I - S_{IN}T_m,\ I = 2, \ldots, N-1 \tag{5.58}$$

2. $$\sum_{J=2}^{N} S_{IJ}T_J = F_I + Q_m\delta_{IN},\ I = 2, \ldots, N \tag{5.59}$$

where the righthand side vector $\mathbf{F}$ is defined as,

$$F_I = \int_0^{z_{max}} f(z)N_I(z)\ dz \tag{5.60}$$

and $f(z)$ is the distribution of internal heating. For given f, T_m or Q_m this system can be solved for the unknown temperature T.

5.8 Using higher order basis functions

In the previous sections we considered mainly application of linear basis functions, here we shall compare solutions of some simple problems solved by applying various basis functions.

We consider the solution of the one-dimensional Poisson equation, $-d^2u/dz^2 = f$ on the domain $0 \leq z \leq 1$ with essential boundary conditions in the boundary points $z = (0, 1)$.

A solution derived from linear basis functions

To investigate the finite element solution of the 1-D Poisson problem we apply a uniform 1-D mesh consisting of a total of four equidistant nodal points spanning three finite elements with corresponding linear basis functions. Each of the nodal points is associated with a specific basis function with a unit value in the nodal point considered. A global stiffness matrix for this problem can be constructed from the three 2×2 element matrices defined in section 5.3.2 by the assembly process discussed in section 5.4.

problem 5.10. *Derive the following global matrix by assembling the three element matrices,*

$$\mathbf{S} = \frac{1}{h}\begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 1 \end{pmatrix} \tag{5.61}$$

where $h = 1/3$ is the length of the elements.

problem 5.11. *As an application of (5.61) we consider first a case with $f = 0$ corresponding to a 1-D Laplace equation. Define the essential boundary conditions as, $u(0) = 0$ and $u(1) = 1$ and derive the following system of equations $\mathbf{SU} = \mathbf{R}$ for the degrees of freedom in the finite element solution corresponding to the internal nodal points,*

$$\frac{1}{h}\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}\begin{pmatrix} U_2 \\ U_3 \end{pmatrix} = \frac{1}{h}\begin{pmatrix} U_1 \\ U_4 \end{pmatrix} = \frac{1}{h}\begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{5.62}$$

Solve these equations and compare the nodal point solution values with the analytical solution of the corresponding 1-D Laplace equation. Verify that in this case the finite element solution and the analytical solution are identical.

Next we consider an extension of the above problem to a 1-D Poisson equation with a uniform right hand side function $f(z) = H$, $H > 0$ a constant. We specify homogeneous essential boundary conditions $u(0) = u(1) = 0$. This represents the steady state temperature u in a heat conduction problem for a layer with uniform internal heat production rate H and prescribed zero temperature at the top and bottom boundary.

problem 5.12. *Derive the following analytical solution for this Poisson problem,*

$$u(z) = \frac{1}{2}Hz(1 - z) \tag{5.63}$$

The Poisson problem can be solved numerically on the same four-point finite element mesh as before.

problem 5.13. *Verify that the following finite element equations hold for the Poisson problem on the four-point mesh.*

$$\frac{1}{h}\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}\begin{pmatrix} U_2 \\ U_3 \end{pmatrix} = \begin{pmatrix} F_2 \\ F_3 \end{pmatrix} = hH\begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{5.64}$$

Derive the numerical solution, $U_2 = U_3 = h^2H$ and compare this result with the analytical solution.