the finite element methods

w. spaans

Vloiestofmechanica
Afd. Weg- en Waterbouwkunde
Technische Hogeschool Delft
FINITE ELEMENT METHODS

AND AN APPLICATION FOR A ONE DIMENSIONAL LONG PERIOD SHALLOW WATER WAVE USING THE METHOD OF GALERKIN AND THE FOUR STEP RUNGA KUTTA METHOD.

Verslag van afstudeer werk, verricht in opdracht van Professor ir. L. J. Mostertman.

Delft, october 1975

W. Spaans.

Technische Hogeschool Delft.
Afdeling Civiele Techniek.
Vakgroep Vloeistofmechanica.
Finite Element Methods

A general review of finite element methods, a discussion of weighted residual methods and the time derivative and a working out of the one dimensional long period shallow water equations in a channel with a rectangular cross section.

SUMMARY

CH 1 INTRODUCTION
1.1 Models and Computational Hydraulics
1.2 Some solution Methods for Long Period Shallow Water Waves

CH 2 BASIC EQUATIONS FOR A LONG PERIOD SHALLOW WATER WAVE IN A WATERWAY WITH A REALISTIC BED CONFIGURATION

CH 3 THE FINITE ELEMENT METHODS
3.1 Why and Where Finite Element Methods are Applied
3.2 The Principle of Finite Element Methods
3.3 The Variational Methods
3.4 The Methods of Weighted Residuals
3.5 The Orthogonalizing Principle
3.6 The Number of Equations
3.7 The Set of Equations for a One Dimensional Case
3.8 Boundary Conditions and the Overdetermined System
3.9 Some Methods of Weighted Residuals
3.10 Principle of Some Weighted Residual Methods
3.11 The Methods of Weighted Residuals, A Generalisation
3.12 A General Set of Equations, Using the Orthogonalizing Principle (Weighted residuals)

CH 4 THE TIME DERIVATED AND INTEGRATION IN TIME
4.1 Some Methods
4.2 Time Discretization
4.3 Runga Kutta 4 Step Method explicit, \( k = 1 \)
4.4 Space - Time Elements

CH 5 THE ONE DIMENSIONAL CASE WITH ONE DEPENDENT VARIABLE USING THE METHOD OF GALERKIN AND LINEAR LANGRANGE SHAPE FUNCTIONS (Hat Functions)
5.1 The Set of Equations
5.2 Comparing with a certain Difference Scheme
5.3 Some Results

THE ONE DIMENSIONAL CASE, A LONG PERIOD SHALLOW WATER WAVE IN A CHANNEL WITH A RECTANGULAR CROSS SECTION USING FOR DISTANCE INTEGRATION THE METHOD OF GALERKIN AND FOR TIME INTEGRATION THE FOUR STEP METHOD OF RUNGA KUTTA

6.1 General Conditions
6.2 The General Set of Equations
6.3 The Mass Equation
6.4 The Momentum Equation
6.5 Combining the Mass- and Momentum Equations
6.6 The Method of Runge Kutta for Time Integration
6.7 The Final Set of Equations (boundary conditions not taken into account)
6.8 The System and the Boundary Conditions
6.9 The Final Set of Equations, Using Langrange Shape Functions and Considering the Boundary Conditions
6.10 The Program and the Flow Diagram
6.11 Some Results
6.12 Comparing with a Finite Difference Program, System 11

FURTHER POSSIBLE INVESTIGATION

APPENDIX A
A.1 Defining the Validity of Equations
A.2 Interpolation Functions
A.3 Trial Functions or Trial Solutions
A.4 Interpolation Functions and Boundary Conditions
A.5 Example of a Trial Solution for the One Dimensional Case Using Langrange Shape Functions
A.6 Curve Fitting

APPENDIX B
B.1 Kinds of Elements
B.2 The One Dimensional Element
B.3 The Two Dimensional Element
B.4 The Three Dimensional Element
SUMMARY

The Finite Element Methods

This report gives a review of some finite element method techniques for the solution of hyperbolic partial differential equations as well as a solution to the one dimensional case of a long period shallow water wave in a waterway with a realistic bed configuration.

The report consists of two main parts

- Chapters 1 to 7, where the finite element methods and some applications are discussed.
- Appendix A and B, where some mathematical principles, together with some types of elements and their interpolation functions are discussed.

First an introduction is given, where some solution methods are discussed, including the harmonic methods, the method of characteristics, the finite difference- and the finite element methods. After the introduction of the basic equations (which describe the phenomenon), a review of the finite element methods is given and the methods of weighted residuals are discussed, especially

- the collocation method
- the subdomain method
- the least squares method
- the method of Galerkin

It is mentioned that these methods can be generalized into the so called "methods of weighted residuals", which are introduced as an orthogonalizing principle.

Next, time propagation is discussed using normal time discretization, the four step method of Runga Kutta or space-time elements. Then an application is given of the scalar wave equation using the method of Galerkin and time discretization. The same set of equations is derived using a six point difference scheme, and some results are also given.
After this, an application is given for the shallow water wave equations, using again the method of Galerkin and the four step Runga-Kutta method. The program is discussed, a flow diagram is given, some results are plotted, the results of a certain case are compared with a finite difference program and the time-step is discussed.
List of Symbols Used in This Report

\( a \) = general writing of a dependent variable

\( a \) = approximated value of the general variable

\( \bar{a} \) = exact solution, defined over the whole region

\( a_k \) = dependent variable, defined at nodal point \( k \)

\( \bar{a} \) = trial solution or trial function, defined over the region

\( \bar{a}_i \) = interpolated value of \( a \), only defined in element \( i \)

\( a_i \) = linearizing coefficient, defined as \( 1 / b_s \)

\( A \) = area of the whole cross section

\( (*) \) \( A \) = \( n * (2m_1+1) \) array in which the elements of the mass matrix can be stored

\( b_1 \) = linearizing coefficient, defined as \( 2Q / A \)

\( b_m \) = mathematical width, defined as \( A / h \)

\( b_s \) = storage width

\( c \) = some constant value

\( C \) = coefficient of De Chézy

\( c_i \) = linearizing coefficient, defined as \( -(Q^2 / h) \cdot b_m + (Q^2 / A^2) \cdot h \cdot \frac{\partial b_m}{\partial h} - g.A \)

\( d_i \) = linearizing coefficient, defined as \( \frac{g.\|Q\|}{C^2.A.R_h} \)

\( e \) = super script, indicating the relation to one element

\( f \) = vector, containing 'constant' values

\( g \) = acceleration due to the gravitation

\( h \) = stage above an arbitrary reference level

\( i \) = sub- or super script, indicating the relation with element \( i \)

\( j \) = sub- or super script, indicating some counter

\( k \) = sub- or super script, indicating the relation with nodal point \( k \)

\( ke \) = number of nodal points in an element

\( L \) = general differential operator

\( m_1 \) = number of upper and/or under diagonals in the mass matrix
\( M \) = mass matrix
\( n \) = identifier of time level \( n \)
\( (*) \) \( n = 2^m_{1+1} \) = number of diagonals in the mass matrix \( M \)
\( \text{ne} \) = number of nodal points within the region
\( \text{nf} \) = number of elements within the region
\( p \) = number of basic equations = number of dependent variables which are defined at each nodal point
\( q \) = lateral inflow
\( Q \) = discharge
\( R \) = residual = difference between the values of the differential equations because of substituting into the trial solution and the exact solution
\( R_h \) = hydraulic radius according to Engelund
\( S \) = stiffness matrix
\( s, d, k \) = subdomain, related to nodal point \( k \)
\( u \) = velocity in \( x \)-direction
\( v \) = velocity in \( y \)-direction
\( v_q \) = velocity of lateral inflow
\( W \) = weighting function
\( W_k \) = weighting function, related to nodal point \( k \)
\( \vec{X} \) = vector, containing somehow the dependent variables on the new time level
\( \vec{Y} \) = vector, containing only terms with 'known' values
\( \alpha \) = velocity distribution coefficient over the whole cross section
\( \varphi \) = angle of lateral inflow
\( \phi \) = interpolation function
\( \phi_k \) = interpolation function, related to nodal point \( k \)

\( (*) \) means only in a special text
CHAPTER 1

INTRODUCTION

1.1 Models and Computational Hydraulics

For process-control, for forecasting, and in general, for describing a phenomenon it is very often useful to build up a model which describes the given phenomenon. Models are used in many ways and, of course, the rule of models is also very important in hydraulics. There are several ways to build up a model. These can be divided into two main groups: Physical modelling and Mathematical modelling.

In physical modelling a problem has to be scaled down to workable proportions, so that the case to be studied can be simulated. A knowledge of hydraulics is necessary but there is no direct use of basic equations.

In mathematical modelling there is a direct use of the basic equations. Moreover, when the basic equations are of the (partial) differential type, some mathematical techniques are necessary to solve them. Contrary to the use of physical models, a certain knowledge and use of mathematical principles is necessary. This way of modelling is often called computational hydraulics.

In the following, only the latter type of modelling will be discussed. Much work has been done already in computational hydraulics and implicitly in the problem of long period shallow water waves. Various methods have been developed in order to solve basic hyperbolic partial differential equations which describe the water movement of these waves.

First a kind of review is given of the most well-known solution methods and next the advantages and disadvantages of these methods are discussed. In the chapters following the introduction, only the finite element methods will be discussed. For further information about the other methods some references are given.

1.2 Some Solution Methods for Long Period Shallow Water Waves

a.) The harmonic methods in which the Fourier series are used to describe the phenomenon.

b.) The method of characteristics, in which Riemann invariants along paths of propagation of disturbances lead to the solution.
c.) The finite difference methods, where a difference notation is used for the differentials with a constant mesh size. A grid is laid over the whole region and the dependent variables are in some way defined in the grid points. The method used can be an implicit or an explicit one, depending on whether there is a direct or an indirect relation between the unknowns in the grid points (including the boundary conditions). A well known technique for an implicit scheme is the "double sweep method".

d.) The finite element methods, where interpolation functions, trial solutions and element integration are used. Within the region and on the boundaries, nodal points are chosen on arbitrary places. In these nodal points the dependent variables are defined. By connecting in some way these nodal points, the elements are constructed.

Finite element methods can be divided into two main groups:
- methods based on the variational principle where the solution is reached by using a minimizing integral (thus called a minimizing principle).
- methods based upon orthogonalizing the residual, which is the difference between the value of the differential equations in which the approximated solution is substituted and the value of the same equations when the exact solution is substituted. A very popular name for this last type of method is "the methods of the weighted residuals".

Of course, all these methods have their advantages and also their disadvantages. A short review of them will be given below:

The harmonic methods [5]

The advantages of the harmonic methods are that by substituting into the basic equations a set of Fourier terms like

\[ h = h_0 + h_1 \sin(\omega t - \phi_1) + h_2 \sin(\omega t - \phi_2) + \ldots \]
\[ Q = Q_0 + Q_1 \sin(\omega t - \psi_1) + Q_2 \sin(\omega t - \psi_2) + \ldots \]

a solution is reached which gives a physical insight into the phenomena, while the solution can be gained by 'hand calculation'. Further, there is the time dimension (which is a rather big problem, for example, in finite elements methods and also in other methods) which here is more or less
disappearing into the solution and which, during the calculation, needs no special attention.

There are certain disadvantages to these methods, for instance when the resistance term is introduced, the whole expression becomes very complicated. The use of harmonic terms with 2, 3 etc. also gives large expressions. Furthermore, the calculation is very time consuming even when only one harmonic term is used. In conclusion, these methods are not especially suitable for a computer, so at present they are not frequently used.

The method of characteristics [1] [2] [5]

The advantage of this method is that when using linearized and far simplified equations, a quick and good physical insight is given into the behaviour of the system. This makes the method very useful for the computation of shock waves, and indeed, this method is often used for getting a first impression of the phenomena.

The disadvantages of this method are that it becomes very complex when the basic equations used become more complex, when the waterway being investigated has a realistic bed configuration instead of a rectangular cross section, when the waterway is getting suddenly wider or smaller, or when there is a weir in the river etc.

In general, the method of characteristics is not suitable for computer calculations because of the very complex program organisation that is necessary.

The finite difference methods [3] [5] [11] [18]

The advantage of this method is the 'simplicity' of transforming differential equations into difference equations and also the relatively little need of computation time (depending, of course, on the difference scheme that is used and the complexity of the system). This is the reason why, at the moment, finite difference methods are the most suitable and the most used computer methods for solving differential equations.

The disadvantages are that no physical insight into the equations is possible (that is to say that from the difference equations that are gained, nothing can be seen about the behaviour of the phenomena). The mesh of the grid
must be locally refined if a detailed look is necessary within a part of the region. This gives a complex program organisation and also many more grid points and variables. Thus requiring much more execution time.

The finite element methods

In general, these types of solution methods for the one dimensional case, do not have the advantage of needing relatively little computer time. The advantages of these methods, however, is that the nodal points (in which the dependent variables are given or to be calculated) can be chosen arbitrarily. This means that a certain grid is not followed. When the places of the nodal points have been fixed then the size of the elements (the shape has been chosen before) is also more or less fixed. The great advantage of finite element methods is the freedom to choose the points where the dependent variables are defined and how to fill up the region with the elements and the choice of element shape. In figure 1.1 one can see how easy refining is possible if the finite element methods are used.

![finite difference grid](image1)

![finite elements](image2)

fig. 1.1 Subdividing a certain region

A disadvantage of the finite element methods is that the calculations are much more complex than with the difference methods. As it is also the case using an implicit difference scheme, rather large matrices must be built up. However, because of the freedom of choosing nodal points, there are in general less points where information is required than in the case when using finite difference methods. So the size of the matrices can be reduced. However, when better matrix calculations are known, and less storage is
necessary (by developing more advanced systems), the finite element methods will be used more in the near future for computational physics and also in computational hydraulics.

As a summary, we can say that for solving hyperbolic differential equations using a computer, two types of methods are applicable: i.e. finite difference methods and finite element methods. The choice of which method to use for a certain case, depends upon the particular problem, and can be different in every case. However, at this moment, in one dimensional problems, finite difference methods normally must be preferred. In two and more dimensional problems often finite elements methods can be at least a very good alternative.

In this report a short review will be given dealing with the two main principles used in finite element methods (FEM). The most useful type of methods, including the very well known method of Galerkin, is discussed. Only a short summary of the other type is given. For using FEM some mathematical principles must be known. The most important ones are discussed in Appendix A while a review of applicable types of elements including their interpolation functions is given in Appendix B.
CHAPTER 2

BASIC EQUATIONS FOR A LONG PERIOD SHALLOW WATER WAVE IN A WATERWAY WITH A REALISTIC BED CONFIGURATION

The derivation of the equations is from [18]

Mass equation

\[
\frac{\partial h}{\partial t} + \frac{1}{b_s} \frac{\partial Q}{\partial x} - q = 0 \tag{2.1}
\]

Momentum equation

\[
\frac{\partial Q}{\partial t} - \frac{2\alpha Q b_s}{A} \frac{\partial h}{\partial t} - \left\{ \frac{\alpha}{A} \left( b_m + h \frac{\partial b_m}{\partial x} \right) - g A \right\} \frac{\partial h}{\partial x} +
\]

\[
\frac{g \sqrt{Q |Q|}}{C_h A R_h} + \left( \frac{2\alpha Q}{A} - \nu_1 \cos \gamma \right) q = 0 \tag{2.2}
\]

in which:

- \( h \) = stage above an arbitrary horizontal reference level
- \( Q \) = discharge
- \( b_s \) = storage width
- \( b_m \) = mathematical width defined as \( A/h \)
- \( \alpha \) = velocity distribution coefficient over the whole cross section
- \( A \) = area of the whole cross section
- \( C \) = coefficient of De Chézy
- \( R_h \) = hydraulic radius according to Engelund
- \( q \) = lateral inflow
- \( \nu_1 \) = velocity of lateral inflow
- \( \gamma \) = angle of lateral inflow
By assuming that:
- there is no lateral inflow
- the coefficient $\alpha$ equals one

and by substituting equation (1) into (2) in order to eliminate the time dependent factor, the set of equations becomes:

\[
\frac{\partial h}{\partial t} + \frac{1}{b_s} \frac{\partial Q}{\partial x} = 0 \tag{2.3}
\]

\[
\frac{\partial Q}{\partial t} + \frac{Q}{A} \frac{\partial Q}{\partial x} - \left( \frac{Q^2}{h} b_m + \frac{Q^2}{A} h \frac{\partial b_m}{\partial h} - q A \right) \frac{\partial h}{\partial x} + \frac{2|Q|}{C^2 A R_h} Q = 0 \tag{2.4}
\]

Taking the lateral inflow into account, the equations become:

\[
\frac{\partial h}{\partial t} + \frac{1}{b_s} \frac{\partial Q}{\partial x} - q = 0 \tag{2.1}
\]

\[
\frac{\partial Q}{\partial t} + \frac{Q}{A} \frac{\partial Q}{\partial x} - \left( \frac{Q^2}{h} b_m + \frac{Q^2}{A} h \frac{\partial b_m}{\partial h} - q A \right) \frac{\partial h}{\partial x} + \frac{2|Q|}{C^2 A R_h} Q - v \cos \varphi \cdot q = 0 \tag{2.5}
\]

This set of hyperbolic partial differential equations (2.3) and (2.4) can be linearized as follows:

\[
\frac{\partial h}{\partial t} + a_i \frac{\partial Q}{\partial x} = 0 \tag{2.6}
\]

\[
\frac{\partial Q}{\partial t} + b_i \frac{\partial Q}{\partial x} + c_i \frac{\partial h}{\partial x} + d_i Q = 0 \tag{2.7}
\]
in which:

\[ a_1 = \frac{1}{b_5} \]
\[ b_1 = \frac{2Q}{A} \]
\[ c_1 = -\left( \frac{Q^2}{2} \frac{m}{A^2} + \frac{Q^2}{A^2} \frac{\partial}{\partial h} \frac{m}{h} - qA \right) \]
\[ d_1 = \frac{g^2 |Q|}{C^2 AR} \]

These coefficients are supposed to be known values.

It is easy to see now that the lateral inflow can be taken into account by adding two coefficients to the equation. Because of the simplicity, this is not done here.

The general set of equations can be written as:

\[ L \cdot a + f = 0 \quad (2.8) \]

where:

- \( a \) = an arbitrary dependent variable
- \( L \) = some differential operator.
CHAPTER 3

THE FINITE ELEMENT METHODS

3.1 Why and Where Finite Element Methods are Applied

As mentioned already in the introduction, the great advantage of the Finite Element Methods (FEM) is the freedom in choosing nodal points. In fact, this is also the reason why the Finite Element Methods are going to find their own applications beside the Finite Difference Methods.

The first applications of FEM were in aeronautics. Afterwards they were introduced into physics and even later, in civil engineering. The first applications in civil engineering were for one dimensional and time independent problems (structures). They were used next in soil mechanics and ground water mechanics, for one and two dimensional, time independent and sometimes, time dependent problems with one dependent variable. Now FEM are even used in (computational) hydraulics for one and two dimensional time dependent problems with two or three dependent variables.

3.2 The Principles of Finite Element Methods

In Finite Difference Methods, a kind of grid is always constructed and the governing differential equations are approximated by using the values of the dependent variables which are defined in the grid points.

For example, the differential equation:

$$\frac{\partial h}{\partial t} = a \frac{\partial^2 h}{\partial x^2}$$  (3.1)
may give in difference notation:

\[
\frac{h_{j}^{n+1} - h_{j}^{n-1}}{2 \Delta t} = \frac{Q_{j+1}^{n} - Q_{j-1}^{n}}{2 \Delta x}
\]

(3.2)

where the subscript \( j \) indicates the grid point used in \( x \) direction, the superscript \( n \) indicates the grid point used in \( t \) direction.

In this way a system can be built up by combining the difference equation to a certain system. Note that the dependent variables are defined in the grid points.

In FEM the approach is different. Instead of choosing a grid with a certain grid size, there is the freedom for choosing nodal points in those places within the region, where the dependent variables are to be calculated. This gives an enormous freedom in calculating. When the nodal points are chosen, a trial function (what is the same as a trial solution) can be defined over the whole region. This is done by creating elements using the nodal points (see fig. 1.1). Over the elements interpolation functions are defined. Together with the nodal values, the trial solution can be defined. Note that the elements only fix up the interpolation functions that are used for defining the trial solution. Of course, before defining a trial solution, the type of elements must be chosen, and these must remain the same over the whole region. Within every element, an interpolated value of the dependent variables must be defined, using interpolated functions and the values in the nodal points. When putting all the interpolated values together (by combining all the elements to the total region) an approximated (interpolated) trial solution is built up, first within each element, and then combined over the whole region. Note that the trial solution is unique within each element. The trial solution looks like:

\[
\hat{a} = \sum_{i=1}^{n_f} \sum_{k=1}^{k_e} \phi_{i,k} a_{k}
\]

(3.3)
where:

\[ n_f = \text{number of elements within the region} \]
\[ k_e = \text{number of nodal points for an element} \]
\[ \phi = \text{interpolation function} \]
\[ k = \text{subscript, indicating the relation with nodal point } k \]
\[ i = \text{subscript indicating the relation with element } i \]
\[ a = \text{dependent variable} \]

(See also Appendix A and B)

Now that the trial solution has been defined, two types of FEM can be regarded, both using the same trial solution, but in a different way. It can be seen from here, that creating a finite number of elements is an essential part of the solution methods, but it is by no means the basic principle of FEM.

The two types of FEM are:

a.) The variational methods, often called the method of Ritz or, when it is about eigen-value problems, the method of Raleigh-Ritz.

b.) The method of weighted residuals, which is a generalisation of some orthogonalizing methods.

The variational methods are very briefly discussed in Chapter 3.3, while the weighted residuals methods are discussed from Chapter 3.4 onwards.

3.3 The Variational Methods

In variational methods the principle, besides defining a trial solution, is the use of a minimizing integral. It seems to be a somewhat 'tricky' principle because a minimizing integral must be known or must be found.

The principle is as follows:

After a trial solution has been defined over the whole region, a mini-
mizing integral must be known, in which the trial solution will be sub­stituted. The minimizing integral is minimized now, with respect to some conditions. This gives a set of equations which, when solved, gives a good approximation of the exact solution.

Here you see one of the great disadvantages of this principle: the fact that a minimizing integral must be known. When this integral is not known or cannot be found, then this principle cannot be used.

For further information, refer back to the literature [6] [15] [16] [20].

3.4 The Methods of Weighted Residuals

The term 'weighted residual' is thought out by Finlayon (see [6] [7]). It is not so much a principle as a common name for some methods.

The principle is as follows:

After a trial solution has been defined over the whole region, then this trial solution is substituted into the basic equations. When \( L \) is the differential operator, with \( \tilde{a} \) the (unknown) exact solution over the whole region and \( \hat{a} \) the trial solution over the whole region, then two sets of equations are gained.

\[
L \cdot \hat{a} + f = 0 \quad \text{all over the region} \quad (3.4)
\]

\[
L \cdot \tilde{a} + f = 0 \quad \text{all over the region} \quad (3.5)
\]

So, in every point of the region, there is a residual between the same basic equations because of the substituting of \( \tilde{a} \) or \( \hat{a} \), so:

\[
(L \cdot \hat{a} + f) - (L \cdot \tilde{a} + f) = R \quad (3.6)
\]

where \( R \) = residual.

The vector \( f \), which contains values, which are independent of \( \tilde{a} \) is set to zero in this report.
When, in the optimal case, the trial solution \( \hat{a} \) equals the exact solution \( a \) in every point of the region, then the residual also equals zero over the whole region.

\[ L \cdot a + f = 0 \]

\[ L \cdot \hat{a} + f = R \]

It can also be said that the residual is a measure for the rate of approximation. So, in principle, the residual can be bent to zero at every point of the region. One way to do this, is by using a kind of curve fitting (see Appendix A). By setting the residual to zero at every point of the region, the problem with an infinite set of variables is created. So the problem must be reduced to a finite set of equations. This can be done by introducing the orthogonalizing principle, (see Chapter 3.5), which says that the residual \( R \), defined over the whole region, is multiplied by some weighting function \( W \) and next integrated over the region. This integral is, at last, set to zero.

\[
\int_{\text{region}} W \cdot R \cdot d(\text{region}) = 0
\]  

(3.7)
The approximation to the exact solution depends here upon the choice of the type of weighting function that will be used. Also the number of independent equations that is gained from equ. (3.7), must equal the number of dependent variables \( a \). Some types of weighting functions are discussed in Chapters 3.9 and 3.10. Although the choice of the weighting function seems to be a little 'tricky', it will turn out to be a more or less logical method. "tricky" here means artificial.

In short, the procedure for using weighting functions is as follows:

a.) define the trial solutions over the region
b.) substitute the trial solution into the basic equations
c.) multiply the residual with a chosen set of weighting functions; integrate each of the multiplications over the region and set the integrals to zero.
d.) solve the set of equations in order to find the approximated trial solution.

3.5 The Orthogonalizing Principle

As mentioned before, we have to work with an exact solution, which is not known and with a trial solution. Also, as mentioned before, the basic equations in which the exact solution is substituted equal zero. In the case of substitution of the trial solution, this is only true when all the required (and given) boundary conditions are given in one point and then it is true in only that point (exact solution). All we want now, is to find those values of the dependent variables that are a good approximation of the exact solution. We can do this, somehow, by setting the basic equations, in which the trial solution is substituted, to zero. Setting only the basic equation to zero gives as many equations as there are types of dependent variables. However, we need as many equations as there are unknown dependent variables. This number of equations can, among others, be gained by making use of the orthogonalizing principle. As mentioned already in Chapter 3.4, where this principle is discussed, a set of weighting functions \( W \) must be created, which are defined somehow over (or within a part of) the

\[
\begin{align*}
\text{number of dependent variables} & = \text{number of types of dependent variables} \\
& \quad (\text{e.g. } Q \text{ and } h) \quad \ast \text{number of nodal points}.
\end{align*}
\]
region. There must be as many weighting functions as there are
nodal points in which the dependent variables are defined.

The basic equations are multiplied by one of the weighting functions
and integrated over the region, which in practice means integrating
over that part of the region where the weighting function is defined.
These integrals are set to zero. This procedure is repeated for
all the \( p \) weighting functions. This results in a set of \( n \times p \)
independent equations from which the approximated solution can be
derived (\( p \) is the number of basic equations).

From fig 3.3 it can be seen how an arbitrary weighting function \( W_k \)
can be defined within the region. Note that for specific orthogonalizing
methods, specific and less arbitrary weighting functions will be used.
(see also chapter 3.9)

As it is true that \( L \cdot \alpha = 0 \), it is also true that:

\[
\int_{\text{region}} W_k \cdot (L \cdot \alpha) \cdot d(\text{region}) = 0 \tag{3.8}
\]

So we can say that if \( \hat{\alpha} \) is a good approximation of the exact solution,
then it is also true that

\[
\int_{\text{region}} W_k \cdot (L \cdot \hat{\alpha}) \cdot d(\text{region}) \tag{3.9}
\]
tends to zero for all values of \( k \).

On the other hand it is very acceptable that, if the integral (3.9) is
set to zero for all values of \( k \) so that a set of equations is created,
the solution of this set of equations will give a good approximation
of the exact solution. The accuracy of this will depend, of course,
upon the choice of the weighting functions. For an application, see
chapter 5, where the scalar wave equation is worked out for a special
weighted residual method, the method of Galerkin.
fig. 3.3 Arbitrary weighting functions and orthogonalizing integrals
3.6 The Number of Equations

A problem is solvable when the number of independent equations equals the number of dependent variables. As mentioned before, we need a certain number of weighting functions. Now, it is usual in weighted residual method, that for every nodal point, a weighted function is created. Also in each nodal point the unknown dependent variables are defined. So the number of independent equations (= number of basic equations * number of weighting functions) equals the number of dependent variables. However, when the weighting functions are defined in every nodal point, then there are, in fact, more equations than unknowns because the required boundary conditions also give some equations. The system is then overdetermined and some equations have to be deleted. The problem, however, is which ones. Before discussing this, we give first, an example of the weighted residuals methods.

3.7 The Set of Equations for a One Dimensional Case

For the one dimensional case with one dependent variable (think for example of a groundwater flow in one direction, where only the phreatic water table is a dependent variable) a set of equations is composed.

Using the orthogonalizing principle and substituting into the basic equation

\[ L \cdot \mathbf{a} = 0 \quad (3.4) \]

the trial solution

\[ \mathbf{\hat{a}} = \sum_{k=1}^{\text{ne}} \phi_k \mathbf{\hat{a}}_k \]

where:

- \( L \) = differential operator
- \( \mathbf{a} \) = dependent variable known or to be known in each nodal point
- \( \phi \) = some weighting function, one for each nodal point
- \( k \) = subscript, indicating the nodal point \( k \),
Using now the orthogonalizing principle, we can write a set of ne equations \((k=1, 2, \ldots, ne)\):

\[
\begin{align*}
\int_{\text{region}} \phi_1 \cdot (L \cdot \hat{n}) \cdot d(\text{region}) &= 0 \\
\int_{\text{region}} \phi_k \cdot (L \cdot \hat{n}) \cdot d(\text{region}) &= 0 \\
\int_{\text{region}} \phi_{ne} \cdot (L \cdot \hat{n}) \cdot d(\text{region}) &= 0
\end{align*}
\]

(3.9-a)

Because there is, besides the set of ne equations, also one boundary condition required, we get for the ne unknown values a set of \((ne+1)\) equations, so there is an overdetermined system. For the two or three dimensional case and/or with more dependent variables, the procedure used is the same.

3.8 Boundary Conditions and the Overdetermined System

Because the system is overdetermined, one (or more) equations must be deleted. The question is now, which ones. In this chapter we will try to give an answer to the case discussed in Chapter 3.7.

In figure 3.4-a the differential equation in which the trial solution is substituted is sketched. Note that from 3.4-a it follows that the boundary condition is given at the left hand boundary because the approximated differential equation equals zero in point 1. In figure 3.4-c until 3.4-i the set of equations derived from equation (3.9).
fig. 3.4 Weighted residuals
is sketched. When the exact solution is substituted into the differential
equation, then the weighted residuals equal zero. This should also be
the case when the trial solution instead of the exact solution is substi-
tuted in the basic equations (see Chapter 3.5). When looking at figure
3.4-c one can see that because of the given boundary condition, the in-
tegral already tends much more to zero than all the other weighted re-
siduals. So, setting this first weighted residual to zero (using the or-
thogonalizing principle) is in fact writing almost an identity, namely
that tending to zero equals zero. This is also the case with the other
weighted residuals, but the tending to zero is, generally speaking,
not as strong as it is in the weighted residual that is related to the boun-
dary condition. From this, one can conclude that the best decision a-
bout how to delete one equation is to choose the equation that is influ-
enced by the boundary condition and the first or last weighting function
\((W_1 \text{ or } W_{ne})\). For the one dimensional case with more than one de-
pendent variable and also for the two and three dimensional case, the
sketching becomes more difficult. The deletion of the number of equa-
tions that overdetermine the system is done in the same way. However,
it would be too lengthy now, to discuss this. An application is given
in Chapter 6, but for only the one dimensional case using long period
shallow water wave equations (see Chapters 2 and 6).

3.9 Some Methods of Weighted Residuals

In the previous chapter the orthogonalizing principle has been
discussed:

The residual \(R\) is the difference between the value of the (partial
differential) basic equations for the case where the trial solution is
substituted and the case where the exact solution is substituted. The
residual is defined over the whole region. This residual was multiplied
by a set of weighting functions \(W\), which are defined for every nodal
point in the region, although it is not basically necessary to define them
for each nodal point. Every term of this set, which consists of the
residual and one weighting function, is orthogonalized by integrating
over the region, and next set to zero:

\[
\int_{\text{region}} W_k \cdot R \cdot d(\text{region}) \quad (k=1,2,\ldots,\text{ne})
\]  

(3.10)
The integral itself is called a weighted residual, the setting to zero of this integral gives an independent equation. In this way a set of ne independent equations was created from which, after considering the boundary conditions, the approximated values of the dependent variables can be solved. In the previous chapter was also discussed that the approximation is depending upon the choice of the weighting functions. Some weighted residual methods, using different kinds of weighting functions are discussed here. For every method care must be taken that a sufficient number of equations is available.

The methods which will be discussed here are:
- The Collocation Method
- The Subdomain Method
- The Least Squares Method
- The Galerkin Method

Other methods like the Moment Method will not be discussed here. In chapter 3.11 a generalization is given of these methods, showing that they all can be considered as the so called Weighted Residual Methods.

3.10 Principle of some Weighted Residual Methods.

a) The Collocation Method

Substituting the exact solution \( \mathbf{a} \) into the basic equations gives 0 for every point in the region:

\[
L \cdot \mathbf{a} = 0 \quad \text{for every point in the region} \quad (3.4)
\]

The approximation is optimal when it is true that the basic equations, in which the trial solution is substituted, equals zero in every point of the region or in other words that the residual equals zero in every point of the region:

\[
R = 0 \quad \text{for every point of the region}
\]

However, this would give an infinite number of equations.

In the collocation method now, the residual is set to zero in all the nodal points:

\[
R_k = 0 \quad (k=1, 2, \ldots, ne) \quad (3.11)
\]
This gives a set of \( ne \times p \) equations; while \( p \) equations can be deleted because of the \( p \) required boundary conditions. The set of \( (ne-1)\times p \) equations are necessary and sufficient for solving the problem.

\[
L \hat{\alpha} = R
\]

fig. 3.5 Collocation method for a one dimensional case

The advantage of the collocation method is that integration over the region is not necessary (no orthogonalizing principle using weighted functions). Why this method is mentioned under the chapter of weighted residual methods will be explained in chapter 3.11.

b) The Subdomain Method

The principle of this method is that the residual \( R \), integrated over an arbitrary subdomain, is set to zero.

\[
L \hat{\alpha} = R
\]

fig 3.6 Subdomain method for a one dimensional case

This is sketched in fig 3.6, where one can see that, when the residual, integrated over a subdomain \( s.d. \) equals zero, the residual is forced
to oscillate around the axis $R = 0$.

In a formula this method can be written as:

$$\int R \cdot d(s.d. k) = 0 \quad (k=1, 2, \ldots, ne)$$  \hspace{1cm} (3.12)

where

$s.d. k$ = part of the region, representing the sub-domain related to the nodal point $k$.

d(s.d k) = infinitesimal part of the sub-domain $k$

There must be as many subdomains as there are nodal points. The deleting of some equations because of overdetermining of the system follows again from the required boundary conditions. For a good approximation it is wishful that the set of sub-domains fill up completely the whole region.

c) The Least Squares Method.

We know already that

$$R = L \cdot \tilde{a} = L \cdot \sum_{k=1}^{ne} \phi_k \cdot \tilde{a}_k$$

so the residual can also be written as

$$R = R \left( \tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_k, \ldots, \tilde{a}_{ne} \right)$$  \hspace{1cm} (3.13)

First the square of the residual is integrated over the region:

$$\int_{\text{region}} R^2 \cdot d(\text{region})$$  \hspace{1cm} (3.14)

Next, for every nodal point, this integral is minimized with respect to the dependent variables. The integral (3.14) will be a minimum if:

$$\frac{\partial}{\partial \tilde{a}_k} \left( \int_{\text{region}} R^2 \cdot d(\text{region}) \right) = 0 \quad (k=1, 2, \ldots, ne)$$

or
\[ \int_{\text{region}} 2. R \cdot \frac{\partial R}{\partial \hat{a}_k} \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, \text{ne}) \]  

This gives a set of ne*p equations. The deleting of some equations because of overdetermining of the system follows again from the required boundary conditions.

d) The Method of Galerkin.

This method follows an orthogonalizing principle. The weighting functions that are used are a certain combination of the interpolation functions which are also used in the trial solution.

Setting the residual, which is integrated over the region, to zero, would give:

\[ \int_{\text{region}} R \cdot d(\text{region}) = \int_{\text{region}} (L \cdot \hat{a}) \cdot d(\text{region}) \]  

(3.16)

When \( \sum_{k=1}^{\text{ne}} \phi_k \) is a complete set of interpolation functions (filling up the whole region), (3.16) can be written as:

\[ \int_{\text{region}} \sum_{k=1}^{\text{ne}} \phi_k \cdot (L \cdot \hat{a}) \cdot d(\text{region}) = 0 \]  

(3.17)

Now the method of Galerkin tells that this shall not only be true for the complete set of interpolation functions, but for each of the individual interpolation functions:

\[ \int_{\text{region}} \phi_k \cdot (L \cdot \hat{a}) \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, \text{ne}) \]  

(3.18)

What is done now, using the method of Galerkin, is that the integration of the residual (and so of the differential equation in which the trial solution is substituted) is splitted up in a number of integrals. (Summing these integrals would give back the integration of the residual over the
This is sketched in fig. 3.7 where the one dimensional case with one dependent variable using linear hatfunctions (see appendix A) is made visible.

![Diagram](image)

**fig. 3.7** The method of Galerkin for the one dimensional case.
From fig 3.7 one can also see that, for example by setting
\[ \int_{\text{region}} \phi_k \cdot (L \cdot \hat{a}) \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, ne), \]
the function \( \phi_k \cdot (L \cdot \hat{a}) \) is forced to oscillate around the axis \( L \cdot \hat{a} = R = 0 \).

Also this method gives a set of \( ne \times p \) equations, while by taking into account the boundary conditions, \( p \) equations can be deleted. So the number of independent equations \( (ne-1) \times p \) equals the number of unknown dependent variables. This set of equations is also solvable. Note that more elements in a certain region give a better approximation to the exact solution.

3.11 The Methods of Weighted Residuals, a Generalization.

In literature about finite element methods, the method(s) of weighted residuals are often introduced as the generalization of the collocation-, subdomain-, least squares-, Galerkin- or moment method. Of course this is true, but one shall not forget that first each of these methods has been developed and that later on these methods are swepted together. [6], [7]. As already mentioned, the method of weighted residuals is an orthogonalizing principle with \( W \) as the weighting function and \( R \) the residual that is weighted.

\[ \int_{\text{region}} W_k \cdot R \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, ne) \quad (3.7) \]

In all the methods mentioned above, the residual is the same: namely being the difference between the differential equations in which the trial solution is substituted and the same equations, in which the exact solution is substituted. The method is characterised by the choice of the weighting functions \( W_k \).

When the weighting functions \( W_k \) are combinations of the same interpolation functions \( \phi_k \), which are also used in the trial solution, then the method of Galerkin appears:
\[ \int \varphi_k \cdot R \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, \text{ne}) \quad (3.20) \]

Because \( \varphi_k \) is only defined within a certain sub-region \( (s. r. K) \), (3.20) can be written as

\[ \int \varphi_k \cdot R \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, \text{ne}) \]

or

\[ \int \varphi_k \cdot (L \cdot \hat{a}) \cdot d(s. r. K) = 0 \quad (k=1, 2, \ldots, \text{ne}) \quad (3.21) \]

When as weighting functions are taken

\[ \frac{\partial R}{\partial \hat{a}_k} \quad (k=1, 2, \ldots, \text{ne}) \quad (3.22) \]

the method of the least squares appears:

\[ \int \frac{\partial R}{\partial \hat{a}_k} \cdot R \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, \text{ne}) \quad (3.23) \]

When the region is subdivided into a number of \( \text{ne} \) subdomains and the weighting functions are set to one in a certain subdomain and zero elsewhere, the subdomain method appears:

\[ \int 1 \cdot R \cdot d(\text{region}) = 0 \quad (k=1, 2, \ldots, \text{ne}) \quad (3.24) \]

When the weighting function is set to one in the nodal point \( k \), one gets the collocation method:

\[ \lim_{\mathcal{N}_k \to 0} \int R \cdot d(\mathcal{N}_k) = 0 \implies R_k = 0 \quad (k=1, 2, \ldots, \text{ne}) \quad (3.25) \]
where

\( J_k = \) an infinitesimal part of the region around nodal point \( k \)

From here it can be seen that the method(s) of weighted residuals are a somewhat 'tricky' way of writing a number of particular approximation methods. Once the background of the methods of weighted residuals is known, it will be much easier to use them as a generalisation of some particular methods.

### 3.12 A General Set of Equations, Using the Orthogonalizing Principle
(Weighted residuals)

In the previous chapters it has already been mentioned that partial differential equations can be written as

\[
L \cdot \mathbf{a} = 0 \quad (3.4)
\]

where

- \( L = \) a differential operator
- \( \mathbf{a} = \) a set of dependent variables

For example:

- the scalar wave equation
  \[
  \frac{\partial^2 a}{\partial t^2} + c \cdot \frac{\partial a}{\partial x} = 0 \quad (3.6)
  \]
  where
  - \( c = \) some constant value
  - \( a = \) the dependent variable

- the set of equations mentioned in chapter 2
  \[
  \frac{\partial h}{\partial t} + a_1 \cdot \frac{\partial Q}{\partial x} = 0 \quad (2.6)
  \]
  \[
  \frac{\partial Q}{\partial t} + b_1 \cdot \frac{\partial Q}{\partial x} + c_1 \cdot \frac{\partial h}{\partial x} + d_1 \cdot Q = 0 \quad (2.7)
  \]

From equation \((3.9.a)\) can be written, because

\[
\hat{a} = \sum_{i=1}^{nf} \sum_{k=1}^{ke} \varphi_{k,i}(x) \cdot a_k(t),
\]
while the weighting functions are only defined in a certain sub-region (let us say $s.r._{k}$),

$$
\begin{bmatrix}
\int_{s.r._{1}} \varphi_{1} \cdot (L \cdot \mathbf{\hat{a}}) \, d(s.r._{1}) &= 0 \\
\int_{s.r._{k}} \varphi_{k} \cdot (L \cdot \mathbf{\hat{a}}) \, d(s.r._{k}) &= 0 \\
\int_{s.r._{ne}} \varphi_{ne} \cdot (L \cdot \mathbf{\hat{a}}) \, d(s.r._{ne}) &= 0
\end{bmatrix} \quad (k=1, 2, \ldots, ne) \quad (3.28)
$$

From this a general set of equations can be written ($k=1, 2, \ldots, ne$),

$$
M \cdot \frac{\partial \mathbf{\hat{a}}}{\partial t} + S \cdot \mathbf{\hat{a}} = \mathbf{f} \quad (3.29)
$$

where $M$ and $S$ are $ne \times ne$ matrices (the boundary conditions are not yet taken into account) and $\mathbf{f}$ is a $ne$ dimensional vector. In chapter 5 and 6 these sets of equations are worked out for the scalar wave equations and the equations discussed in chapter 2. First, however, the time-derivative is discussed in chapter 4.
CHAPTER 4

THE TIME DERIVATIVE AND INTEGRATION IN TIME.

4.1 Some methods

In chapter 2 and 3 it has been shown that the general set of equations can be written as

\[ M \frac{\partial \xi}{\partial t} + S \xi = \xi \]

\[ M \dot{\xi} + S \xi = \xi \]  

(4.1)

Because all the values of the dependent variables \( \xi \) (or \( Q \) and \( h \)) are given or have to be calculated as non-derivatives, the time derivative of \( \xi \) has to be solved somehow.

Three solution methods are discussed here:

a) Time-discretization \[ [13], [14], [9] \]

b) The method of Runga Kutta \[ [13], [14] \]

c) Using time-space elements \[ [10], [4] \]

4.2 Time-discretization

A general known discretization of \( \dot{\xi} \) is

1) Euler explicit

\[ \xi^{n+1} = \xi^n + \Delta t \cdot \dot{\xi} \]

\[ \dot{\xi}^n = \frac{\xi^{n+1} - \xi^n}{\Delta t} \]  

(4.2)

, a normal discretization

This filled in into (4.1) gives

\[ M \left( \frac{\xi^{n+1} - \xi^n}{\Delta t} \right) = -S \xi^n + \xi^n \]

\[ M \xi^{n+1} = \Delta t (-S \xi^n + \xi^n) + M \xi^n \]  

(4.3)
Which system is solvable.

2) Heun explicit

\[ \ddot{a}^{n+1} = \ddot{a}^n + \Delta t \cdot \dddot{a}^n \]  \hspace{1cm} (4.4)

\[ \ddot{a}^{n+1} = \ddot{a}^n + \frac{\Delta t}{2} \left( \dddot{a}^n + \dddot{a}^{n+1} \right) \]  \hspace{1cm} (4.5)

From (4.1) follows \( \dddot{a}^n \)

\[ M(\dddot{a}^n) = -S \cdot \dddot{a}^n + \frac{f}{\Delta t} \Rightarrow \dddot{a}^n \]  \hspace{1cm} (4.6)

From (4.1) and (4.6) follows \( \dddot{a}^{n+1} \)

\[ M(\dddot{a}^{n+1}) = \Delta t \left( -S \cdot \dddot{a}^n + \frac{f}{\Delta t} \right) + M \ddot{a}^n \Rightarrow \dddot{a}^{n+1} \]  \hspace{1cm} (4.7)

From (4.5), (4.6) and (4.7) follows at last

\[ \ddot{a}^{n+1} = \ddot{a}^n + \frac{\Delta t}{2} \left( \dddot{a}^n + \dddot{a}^{n+1} \right) \Rightarrow \ddot{a}^{n+1} \]  \hspace{1cm} (4.8)

Of course are also other discretizations possible

3) Implicit discretization methods.

Often are these methods more complicated than the explicit ones

a) Heun implicit

\[ \ddot{a}^{n+1} = \ddot{a}^n + \Delta t \left( \dddot{a}^n + \dddot{a}^{n+1} \right) \]  \hspace{1cm} (4.9)

b) Euler implicit

\[ \ddot{a}^{n+1} = \ddot{a}^n + \Delta t \cdot \dddot{a}^{n+1} \]  \hspace{1cm} (4.10)

c) Gear1, k=2

\[ 3 \cdot \ddot{a}^{n+1} = 4 \cdot \ddot{a}^n - \ddot{a}^{n-1} + \Delta t \cdot \dddot{a}^{n+1} \]  \hspace{1cm} (4.11)

d) Milne Simpson implicit, k=2

\[ \ddot{a}^{n+1} = \ddot{a}^{n-1} + \frac{\Delta t}{3} \left( \dddot{a}^{n+1} + 4 \dddot{a}^n + \dddot{a}^{n-1} \right) \]  \hspace{1cm} (4.12)

4.3 Runge Kutta 4 Step Method explicit, k = 1

In general the four step Runge Kutta method is written as:
Equation (4.13) can be written as:

\[
\begin{align*}
\dot{a}^n &= \frac{\Delta t}{\Delta t} (a_{n+1}^n - a^n) \\
\dot{a}_{*+1} &= \frac{\Delta t}{\Delta t} (a_{*+1}^{n+1} - a^n) \\
\dot{a}_{*+1} &= \frac{\Delta t}{\Delta t} (a_{*+1}^{n+1} - a^n) \\
\dot{a}_{**+1} &= \frac{\Delta t}{\Delta t} (a_{**+1}^{n+1} - a^n)
\end{align*}
\] (4.15)

Substituting (4.15) into (4.1) gives:

\[
\begin{align*}
M. \left( \frac{\Delta t}{\Delta t} (a_{*+1}^{n+1} - a^n) \right) &= -S \cdot a^n + \frac{q^n}{2} \\
M. \left( \frac{\Delta t}{\Delta t} (a_{*+1}^{n+1} - a^n) \right) &= -S \cdot a_{*+1}^{n+1} + \frac{q^{n+1}}{2} \\
M. \left( \frac{\Delta t}{\Delta t} (a_{**+1}^{n+1} - a^n) \right) &= -S \cdot a_{**+1}^{n+1} + \frac{q^{n+1}}{2} \\
M. \left( \frac{\Delta t}{\Delta t} (a_{**+1}^{n+1} - a^n) \right) &= -S \cdot a_{**+1}^{n+1} + \frac{q^{n+1}}{2}
\end{align*}
\] (4.16)

Matrix \( S \) is a function of \( a \). So in fact \( S \) must have asterices too. This is not done, however.

From (4.6) follows now equation (4.17). This is a four step scheme, which is solvable by using certain matrix calculations. This method will be used in chapter 6 where an example is given for a one-dimensional case with two dependent variables \( Q \) and \( h \).
4.4 Space - Time Elements

Just like it is possible to create interpolation functions within an element only in space, it is also possible to create interpolation functions in space and time. Refer also literature [19] and [20]

Instead of using a trial solution of the type

\[ \hat{\alpha} (x, y, z, t) = \sum_{k=1}^{2n} \phi_k (x, y, z) \cdot \hat{\alpha}_k (t) \]  

(4.18)

a trial solution can be used of the type

\[ \hat{\alpha} (x, y, z, t) = \sum_{k=1}^{2n} \phi_k (x, y, z, t) \cdot \hat{\alpha}_k, n = \sum_{n=1}^{n+1} \sum_{k=1}^{n} \phi_k, n (x, y, z, t) \cdot \hat{\alpha}_k, n \]  

(4.19)

For the one dimensional case can this be explained as follows:

Consider a (rectangular) element in the x-t plane

Fig. 4.1 A rectangular element in the x-t plane
The linear interpolation functions over the element $e_{k,n}$ are sketched in fig 4.2 and 4.3.

**fig 4.2** The $x$-$t$ plane

\[
\phi_{k,n} = (x_{k+1} - x)(t^{n+1} - t) / (\Delta t \cdot \Delta x)
\]

**fig 4.3** The interpolation functions

\[
\phi_{k+1,n} = (x - x_k)(t^{n+1} - t) / (\Delta t \cdot \Delta x)
\]

Composed over the element $e_{k,n}$, the trial solution is like

\[
\hat{a}_{k,n} = \sum_{k} \sum_{n} \phi_{k,n} \cdot \hat{a}_{k,n}
\]

(4.20)
It will be clear that creating elements and interpolation functions in the x-t plane is principally the same as creating elements and interpolation functions in the two dimensional x-y plane. (see also appendix B).

Filling up now the whole region with elements and so defining the trial solution over the whole region, gives the connection with the new time step.

Using the method of Galerkin, as weighting functions a kind of two dimensional hat functions can be introduced, one for each nodal point. An example of how to do this is given in fig 4.6.
This gives the general Galerkin equation for every timestep

\[
\int_{\text{region}} \phi_{kn} (x, t) \cdot (L \cdot \hat{\alpha}) \cdot d(\text{region})
\]

(4.21)

or

\[
\int_{\text{region}} \int_{\text{region}} \phi_{kn} (x, t) \cdot (L \cdot \hat{\alpha}) \cdot d(\text{time}) \cdot d(\text{space})
\]

(4.22)

which gives for every timestep a set of \( ne \times p \) equations, where

\( ne \) = number of nodal points

\( p \) = number of dependent variables in each nodal point.

Taking the boundary conditions into account, the number of equations becomes

\( 2 \times P \times (ne-1) \), the set of \( 2 \times P \times ne \) equations minus the required boundary conditions. So the system is solvable.

Of course, other types of elements can be used but the rectangular shape is the most suitable for the one dimensional case.

For the two dimensional case, the rectangular shaped and the triangular based prism shaped elements can be used.

Fig 4.8 Two dimensional time-space elements.
CHAPTER 5

THE ONE DIMENSIONAL CASE WITH ONE DEPENDENT VARIABLE, USING THE METHOD OF GALERKIN AND LINEAR LAGRANGE SHAPE FUNCTIONS

5.1 The set of Equations.

For this one dimensional case, the equation that describes the phenomenon is the scalar wave equation (5.1). Only one boundary condition is required because there is only one dependent variable. It is assumed here that the left boundary value is given in time direction (nodal point 1).

The scalar wave equation is given as

\[ \frac{\partial a}{\partial t} + c \frac{\partial a}{\partial x} = 0 \]  \hspace{1cm} (5.1)

where

- \( a \) = dependent variable
- \( c \) = some constant value

Using now the method of Galerkin, equation (5.1) becomes

\[ \int_{\text{region}} \phi_k \left( \frac{\partial \hat{a}}{\partial t} + c \frac{\partial \hat{a}}{\partial x} \right) \, dx = 0 \]  \hspace{1cm} (k=1, 2, \ldots, \text{ne}) \hspace{1cm} (5.2)

where

\[ a = \sum_{k=1}^{\text{ne}} \phi_k \cdot \hat{a}_k \]

\( \phi_k \) = hat function, defined in nodal point k

\( \hat{a}_k \) = unknown approximated value of dependent variable \( a \) in nodal point k.

\[ \begin{array}{c}
\text{fig. 5.1 Hat function in nodal point } k.
\end{array} \]
Because the weighting function $\phi_k$ is only defined in the elements $\Delta_{k-1}$ and $\Delta_k$ and equals zero in the other elements, the set of equations (5.2) becomes

$$ \int_{\Delta_{k-1} + \Delta_k} \phi_k \left( \frac{\partial^2 \phi_k}{\partial t^2} + c \frac{\partial \phi_k}{\partial x} \right) \, dx = 0 \quad (k=1, 2, \ldots, ne) \quad (5.3) $$

Writing out in (5.3) the trial solution, gives

$$ \int_{\Delta_{k-1} + \Delta_k} \phi_k \cdot \left( \frac{\partial \hat{\phi}_k}{\partial t} + \phi_k \cdot \hat{a}_{k-1} + \phi_k \cdot \hat{a}_k + \phi_{k+1} \cdot \hat{a}_{k+1} + \ldots + \phi_{ne} \cdot \hat{a}_{ne} \right) + 
+ c \frac{\partial}{\partial x} \left( \phi_k \cdot \hat{a}_{k-1} + \phi_k \cdot \hat{a}_k + \phi_{k+1} \cdot \hat{a}_{k+1} + \ldots + \phi_{ne} \cdot \hat{a}_{ne} \right) \, dx = 0 \quad (k=1, 2, \ldots, ne) \quad (5.4) $$

where

$$ \phi_k = \phi_{k, k-1} + \phi_{k, k} \quad \text{(see Appendix A)} $$

This set of equations can also be written as

$$ \int_{\Delta_{k-1} + \Delta_k} \phi_k \left( \frac{\partial}{\partial t} (\phi_{k, k-1} \cdot \hat{a}_{k-1} + \phi_k \cdot \hat{a}_k + \phi_{k+1} \cdot \hat{a}_{k+1} + \ldots + \phi_{ne} \cdot \hat{a}_{ne}) \right) + 
+ c \frac{\partial}{\partial x} (\phi_{k, k-1} \cdot \hat{a}_{k-1} + \phi_k \cdot \hat{a}_k + \phi_{k+1} \cdot \hat{a}_{k+1} + \ldots + \phi_{ne} \cdot \hat{a}_{ne}) \, dx = 0 \quad (k=1, 2, \ldots, ne) \quad (5.5) $$

As mentioned before, the trial solution can be written as

$$ \hat{a} = \sum_{k=1}^{ne} \phi_k \cdot \hat{a}_k = \sum_{k=1}^{ne} \phi_k (x) \cdot \hat{a}_k (t) \quad (5.6) $$
The derivatives of (5.6) can be written as

\[ \frac{\partial}{\partial t}(\phi_k \cdot \delta_k) = \phi_k \cdot \frac{\partial \delta_k}{\partial t}, \quad \frac{\partial}{\partial x}(\phi_k \cdot \delta_k) = \frac{\partial \phi_k}{\partial x} \cdot \delta_k \quad (5.7) \]

Substituting now (5.7) into (5.5) gives

\[ \int \left( \phi_k \cdot \phi_{k-1} \cdot \frac{\partial a_{k-1}}{\partial t} + \phi_k \cdot \phi_k \cdot \frac{\partial a_k}{\partial t} + \phi_k \cdot \phi_{k+1} \cdot \frac{\partial a_{k+1}}{\partial t} + \right. \\
\left. + \phi_k \cdot \phi_{k-1} \cdot \frac{\partial a_k}{\partial x} + c \cdot \phi_k \cdot \frac{\partial \phi_k}{\partial x} \cdot a_k + c \cdot \phi_k \cdot \frac{\partial a_{k+1}}{\partial x} \right) \, dx = 0 \\
(k=1, 2, \ldots, ne) \quad (5.8) \]

which can also be written as

\[ \int \phi_k \cdot \phi_{k-1} \cdot \frac{\partial a_{k-1}}{\partial t} \, dx + \int \phi_k \cdot \phi_k \cdot \frac{\partial a_k}{\partial t} \, dx + \int \phi_k \cdot \phi_{k+1} \cdot \frac{\partial a_{k+1}}{\partial t} \, dx + \right. \\
\left. + \int \phi_k \cdot \phi_{k-1} \cdot \frac{\partial a_k}{\partial x} \, dx + \int \phi_k \cdot \phi_k \cdot \frac{\partial a_k}{\partial x} \, dx + \int \phi_k \cdot \phi_{k+1} \cdot \frac{\partial a_{k+1}}{\partial x} \, dx + \right. \\
\left. + \int c \cdot \phi_k \cdot \phi_{k-1} \cdot \frac{\partial a_{k-1}}{\partial t} \, dx + \int c \cdot \phi_k \cdot \phi_k \cdot \frac{\partial a_k}{\partial t} \, dx + \int c \cdot \phi_k \cdot \phi_{k+1} \cdot \frac{\partial a_{k+1}}{\partial t} \, dx + \right. \\
\left. + \int c \cdot \phi_k \cdot \phi_{k-1} \cdot \frac{\partial a_k}{\partial x} \, dx + \int c \cdot \phi_k \cdot \phi_k \cdot \frac{\partial a_k}{\partial x} \, dx + \int c \cdot \phi_k \cdot \phi_{k+1} \cdot \frac{\partial a_{k+1}}{\partial x} \, dx = 0 \\
(k=1, 2, \ldots, ne) \quad (5.9) \]
Because $\hat{a}_k$ is given as a value in a number of discrete nodal points, so independent of $x$, (5.9) becomes

$$\frac{\partial \hat{a}_{k-1}}{\partial t} \int_{\Delta k-1} \phi_{k_{k-1, k-1}} \cdot \hat{d}z + \frac{\partial \hat{a}_k}{\partial t} \int_{\Delta k} \phi_{k_{k, k-1}} \cdot \hat{d}z +$$

$$+ \frac{\partial \hat{a}_k}{\partial t} \int_{\Delta k} c \cdot \phi_{k_k, k} \cdot \hat{d}z + \frac{\partial \hat{a}_k}{\partial t} \int_{\Delta k} c \cdot \phi_{k_{k+1}, k} \cdot \hat{d}z +$$

$$+ \hat{a}_{k-1} \int_{\Delta k-1} \phi_{k_{k-1, k-1}} \frac{\partial \phi_{k_{k-1, k-1}}}{\partial x} \cdot \hat{d}z + \hat{a}_k \int_{\Delta k} \phi_{k_{k, k-1}} \frac{\partial \phi_{k_{k-1, k-1}}}{\partial x} \cdot \hat{d}z +$$

$$+ \hat{a}_k \int_{\Delta k} c \cdot \phi_{k_k, k} \frac{\partial \phi_{k_{k-1, k-1}}}{\partial x} \cdot \hat{d}z + \hat{a}_k \int_{\Delta k} c \cdot \phi_{k_{k+1}, k} \frac{\partial \phi_{k_{k+1, k}}}{\partial x} \cdot \hat{d}z = 0$$

(k=1, 2, ..., ne) (5.10)

Note that when $k=1$ and $k=ne$, the shape functions $\phi_{k_{k-1}, k-1}$ (when $k=1$) and $\phi_{n_{k+1}, ne}$ (when $k=ne$) do not exist.

Because the left hand boundary condition is given can the first equation in (5.10) be deleted. The terms with $\phi_{ne+1, ne}$ are set to zero.

The integrals that are used in (5.10) consist of the following components:
Substituting these components into equ. (5.10) gives:

\[
\int_{\Delta k} \phi_{k-1,k-1} \cdot \phi_{k-1,k-1} \cdot \frac{\Delta - x}{\Delta} \cdot dx = \int_{\Delta k} \phi_{k,k} \cdot \phi_{k,k} \cdot \frac{x}{\Delta} \cdot dx = \int_{\Delta k} \phi_{k,k+1} \cdot \phi_{k,k} \cdot \frac{x}{\Delta} \cdot dx = \int_{\Delta k} \frac{\Delta - x}{\Delta} \cdot \frac{x}{\Delta} \cdot dx = \frac{\Delta}{3}
\]

\[
\int_{\Delta k} \phi_{k,k-1} \cdot \phi_{k,k-1} \cdot \frac{x}{\Delta} \cdot dx = \int_{\Delta k} \frac{x}{\Delta} \cdot \frac{x}{\Delta} \cdot dx = \frac{\Delta}{3}
\]

\[
\int_{\Delta k} \phi_{k,k} \cdot \phi_{k,k} \cdot \frac{\Delta - x}{\Delta} \cdot dx = \int_{\Delta k} \frac{\Delta - x}{\Delta} \cdot \frac{\Delta - x}{\Delta} \cdot dx = \frac{\Delta}{3}
\]

\[
\int_{\Delta k} \phi_{k,k+1} \cdot \frac{\partial \phi_{k-1,k+1}}{\partial x} \cdot \frac{x}{\Delta} \cdot dx = \int_{\Delta k} \frac{x}{\Delta} \cdot \frac{1}{\Delta} \cdot dx = -\frac{1}{k}
\]

\[
\int_{\Delta k} \phi_{k,k-1} \cdot \frac{\partial \phi_{k,k+1}}{\partial x} \cdot \frac{x}{\Delta} \cdot dx = \int_{\Delta k} \frac{x}{\Delta} \cdot \frac{1}{\Delta} \cdot dx = \frac{1}{k}
\]

\[(k=1, 2, \ldots, n) \quad (5.11)\]
\[
\int_{\Delta} \phi_{kh} \cdot \frac{\partial \phi_{kh}}{\partial x} \cdot dx = \frac{\Delta - x}{\Delta} \cdot \frac{1}{\Delta} = \frac{1}{\Delta} \\
\int_{\Delta} \phi_{kh} \cdot \frac{\partial \phi_{kh}}{\partial x} \cdot dy = \frac{\Delta - y}{\Delta} \cdot \frac{1}{\Delta} = \frac{1}{\Delta}
\]

\[\text{(k=1, 2, \ldots, ne) (5.12)}\]

Substituting these integrals into (5.10), the set of equations becomes

\[
\begin{align*}
\frac{\Delta}{6} \frac{\partial a}{\partial t} + \frac{x}{3} \frac{\partial a}{\partial t} + \frac{\Delta}{6} \frac{\partial a^2}{\partial t} - \frac{c}{2} \frac{\partial a}{\partial t} + \frac{c}{2} \frac{\partial a}{\partial t} &= 0 \\
\cdots \\
\frac{\Delta}{6} \frac{\partial a_{kn}}{\partial t} + \frac{x}{3} \frac{\partial a_{kn}}{\partial t} + \frac{\Delta}{6} \frac{\partial a_{kn}}{\partial t} - \frac{c}{2} \frac{\partial a_{kn}}{\partial t} + \frac{c}{2} \frac{\partial a_{kn}}{\partial t} &= 0 \\
\cdots \\
\frac{\Delta}{6} \frac{\partial a_{ke}}{\partial t} + \frac{x}{3} \frac{\partial a_{ne}}{\partial t} - \frac{c}{2} \frac{\partial a_{ne-1}}{\partial t} + \frac{c}{2} \frac{\partial a_{ne}}{\partial t} &= 0
\end{align*}
\]

\[\text{(k=1, 2, \ldots, ne) (5.13)}\]

Multiplied by \(\frac{\Delta}{6}\), (5.13) becomes

\[
\begin{align*}
\frac{\partial a}{\partial t} + 4 \frac{\partial a}{\partial t} + \frac{\partial a}{\partial t} - \frac{3c}{\Delta} \frac{\partial a}{\partial t} + \frac{3c}{\Delta} \frac{\partial a}{\partial t} &= 0 \\
\cdots \\
\frac{\partial a_{kn}}{\partial t} + 4 \frac{\partial a_{kn}}{\partial t} + \frac{\partial a_{kn}}{\partial t} - \frac{3c}{\Delta} \frac{\partial a_{kn}}{\partial t} + \frac{3c}{\Delta} \frac{\partial a_{kn}}{\partial t} &= 0 \\
\cdots \\
\frac{\partial a_{ne}}{\partial t} + \frac{3c}{\Delta} \frac{\partial a_{ne}}{\partial t} - \frac{3c}{\Delta} \frac{\partial a_{ne-1}}{\partial t} + \frac{3c}{\Delta} \frac{\partial a_{ne}}{\partial t} &= 0
\end{align*}
\]

\[\text{(k=1, 2, \ldots, ne) (5.14)}\]
Because the left hand boundary date is given, is the dependent variable which is considered in the first nodal point not an approximated value, so
\[ \hat{a}_1 = a_1 \] for every timelevel

Equation (5.14) becomes now

\[
\begin{bmatrix}
1 & 4 & 1 \\
\vdots & \vdots & \vdots \\
1 & 4 & 1 \\
\end{bmatrix}
\begin{bmatrix}
\frac{\partial a}{\partial t} \\
\vdots \\
\frac{\partial a}{\partial t} \\
\end{bmatrix}
+ C
\begin{bmatrix}
-\frac{2}{\Delta t} & 0 & \frac{2}{\Delta t} \\
\vdots & \ddots & \vdots \\
-\frac{2}{\Delta t} & 0 & \frac{2}{\Delta t} \\
\end{bmatrix}
\begin{bmatrix}
\hat{a}_1 \\
\vdots \\
\hat{a}_n \\
\end{bmatrix}
= 0
\]

(k=1, 2, ..., ne) (5.15)

which in generalised shape can be written as

\[
M \frac{\partial \hat{a}}{\partial t} + S \cdot \hat{a} = 0
\] (5.16)

The time derivative \( \frac{\partial \hat{a}}{\partial t} \) is solved here by using a very simple time discretization

\[
\frac{\partial \hat{a}}{\partial t} = \frac{\hat{a}^{n+1} - \hat{a}^n}{\Delta t} = \frac{\hat{a}^t - \hat{a}}{\Delta t}
\] (5.17)

Substitution of (5.17) into (5.15) gives
Separating the dependent variables on the new time level and also the known boundary data $a_i$ and $a_i'$, (5.18) becomes

\[
\begin{bmatrix}
\frac{1}{\Delta T} & 1 & 4 & 1 \\
4 & 1 & 4 & 1 \\
\cdot & \cdot & \cdot & \cdot \\
1 & 4 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
a_i' - a_i \\
a_h' - a_h \\
\cdot \\
a_m' - a_m
\end{bmatrix}
\begin{bmatrix}
-\frac{2}{\Delta} & 0 & \frac{3}{\Delta} \\
0 & -\frac{3}{\Delta} & 0 \\
\cdot \\
-\frac{3}{\Delta} & \frac{3}{\Delta}
\end{bmatrix}
\begin{bmatrix}
a_i' + a_i \\
a_h' + a_h \\
\cdot \\
a_m' + a_m
\end{bmatrix}
= 0
\]

(5.18)

(5.19)

Setting $\frac{2c \Delta t}{2 \Delta} = \delta$ gives at last
This is a set of equations which is solvable. (See chapter 5.3)

5.2 Comparing with a Certain Finite Difference Scheme

Sometimes a finite element method gives the same set of equations as when a certain difference method is used. This will be shown here for the scalar wave equation (5.1), assuming that also the left-hand boundary condition is known for every time level. An implicit difference scheme will be used (fig 5.2)

Using this scheme, the following equations can be taken (note that also other equations are possible):
\[
\frac{\partial a}{\partial t} = \frac{a^{n+1} - a^n}{\Delta t} \quad (5.23)
\]
\[
\frac{\partial a}{\partial x} = \frac{a_{k+1} - a_{k-1}}{2. \Delta x} = \frac{a_{k+1} + a_{k-1}}{2} - \frac{a_{k-1} + a_{k-1}}{2} \quad (5.24)
\]

Substituting these equations into (5.1) gives:

\[
\left(\frac{a^{n+1} - a^n}{\Delta t}\right) + c \left(\frac{a_{k+1} - a_{k-1}}{2. \Delta x}\right) = 0 \quad (k=2, 3, \ldots, ne-1) \quad (5.25)
\]

or

\[
\left(\frac{a^{n+1}_{k-1} + 4a^{n+1}_k + a^{n+1}_{k+1}}{6\Delta t}\right) + \left(\frac{c \cdot a^{n+1}_{k-1} - c \cdot a^{n+1}_{k+1}}{4 \cdot \Delta x}\right) = \\
\left(\frac{a^n_{k-1} + 4a^n_k + a^n_{k+1}}{6\Delta t}\right) + \left(\frac{-c \cdot a^n_{k-1} + c \cdot a^n_{k+1}}{4 \cdot \Delta x}\right) \quad (k=2, 3, \ldots, ne-1) \quad (5.26)
\]

This can be written as:

\[
\left(\frac{a^{n+1}_{k-1} - c \cdot a^{n+1}_k}{6\Delta t} + \frac{4a^{n+1}_k}{6\Delta t}\right) + \left(\frac{a^{n+1}_{k+1}}{6\Delta t} + \frac{c \cdot a^{n+1}_{k+1}}{4 \cdot \Delta x}\right) = \\
\left(\frac{a^n_{k-1} + c \cdot a^n_k}{6\Delta t} + \frac{4a^n_k}{6\Delta t}\right) + \left(\frac{a^n_{k+1}}{6\Delta t} - \frac{c \cdot a^n_{k+1}}{4 \cdot \Delta x}\right) \quad (k=2, 3, \ldots, ne-1) \quad (5.27)
\]

Multiplying by \(6 \cdot \Delta t\) and setting \(\frac{3c \cdot \Delta t}{2 \cdot \Delta x} = \delta\), equ. (5.27) becomes:

\[
\begin{align*}
& \left[\left(1 - \delta\right) a^{n+1}_{k-1} + 4 \cdot a^{n+1}_{k} + \left(1 + \delta\right) a^{n+1}_{k+1}\right] = \\
& \left[\left(1 + \delta\right) a^{n}_{k-1} + 4 \cdot a^{n}_{k} + \left(1 - \delta\right) a^{n}_{k+1}\right] \quad (k=2, 3, \ldots, ne-1)
\end{align*}
\]

These equations are centered in x-direction in the nodal point \(k\). So a set of (ne-2) equations can be derived. (k=2, 3, \ldots, ne-1).
For the nodal points \( k=1 \) and \( k=ne \), the derivation is a little different, because here a four point scheme must be used. (fig. 5.2)

For \( k=ne \) the equations \((5.21, \ldots, 5.24)\) become:

\[
\begin{align*}
a^{n+1} & = \frac{a_{ne-1}^{n} + 2 \cdot a_{ne}^{n}}{3} \\
a^n & = \frac{a_{ne-1}^{n} + 2 \cdot a_{ne}^{n}}{3} \\
\frac{\partial a}{\partial t} & = \frac{a^{n+1} - a^n}{\Delta t} \\
\frac{\partial a}{\partial x} & = \frac{a_{ne} - a_{ne-1}}{\Delta x}
\end{align*}
\]

Substituting equations \((5.29, \ldots, 5.32)\) into \((5.1)\) gives:

\[
\begin{align*}
\left(\frac{a_{ne-1}^{n+1} + 2 \cdot a_{ne}^{n+1}}{3 \Delta t}\right) + \left(- \frac{c \cdot a_{ne-1}^{n} + c \cdot a_{ne}^{n+1}}{2 \Delta x}\right) = \\
\left(\frac{a_{ne-1}^{n} + 2 \cdot a_{ne}^{n}}{3 \Delta t}\right) + \left(+ c \cdot a_{ne-1}^{n} - c \cdot a_{ne}^{n}\right)
\end{align*}
\]

which can be written as:

\[
\begin{align*}
\left(\frac{a_{ne-1}^{n+1} - c \cdot a_{ne-1}^{n}}{3 \Delta t} \cdot \frac{1}{2 \Delta x}\right) + \left(\frac{2 \cdot a_{ne}^{n+1}}{3 \Delta t} + \frac{c \cdot a_{ne}^{n+1}}{2 \Delta x}\right) = \\
\left(\frac{a_{ne-1}^{n}}{3 \Delta t} + \frac{c \cdot a_{ne-1}^{n}}{2 \Delta x}\right) + \left(\frac{2 \cdot a_{ne}^{n}}{3 \Delta t} - \frac{c \cdot a_{ne}^{n}}{2 \Delta x}\right)
\end{align*}
\]

Multiplying by \(3 \Delta t\) and again setting \(\frac{3c \cdot \Delta t}{2 \cdot \Delta x} = \delta\), equ. \((5.34)\) becomes:

\[
\left[ (1-\delta) \frac{a_{ne-1}^{n+1}}{(1+\delta) \cdot a_{ne-1}^{n}} + (2+\delta) \cdot a_{ne}^{n+1} \right] = \left[ (1+\delta) \cdot a_{ne-1}^{n} + (2-\delta) \cdot a_{ne}^{n} \right]
\]

Also for \( k=1 \), (where the boundary condition is given), an equation can be derived. However we are looking for a comparison with the finite element set of equations. So this equation will not be taken into account.

Note that \( \delta \) is the same as in equation \((5.20)\).
By putting together the equations (5.28) and (5.35) to form one system or set of equations \((k = 2, 3, \ldots, ne)\), we get

\[
\begin{bmatrix}
1-\delta & 4 & 1+\delta \\
1-\delta & 4 & 1+\delta \\
1-\delta & 2+\delta
\end{bmatrix}
\begin{bmatrix}
a_1 \\
\hat{a}_k \\
\hat{a}_n
\end{bmatrix}
= 
\begin{bmatrix}
1+\delta & 4 & 1-\delta \\
1+\delta & 4 & 1-\delta \\
1+\delta & 2-\delta
\end{bmatrix}
\begin{bmatrix}
a_1 \\
\hat{a}_k \\
\hat{a}_n
\end{bmatrix}
\]

\((k=2, 3, \ldots, ne)\) \hspace{1cm} (5.36)

or because \(a_1^{n+1}\) is a known value (boundary condition):

\[
\begin{bmatrix}
4 & 1+\delta \\
1-\delta & 4 & 1+\delta \\
1-\delta & 2+\delta
\end{bmatrix}
\begin{bmatrix}
\hat{a}_1 \\
\hat{a}_k \\
\hat{a}_n
\end{bmatrix}
= 
\begin{bmatrix}
4 & 1-\delta \\
1+\delta & 4 & 1-\delta \\
1+\delta & 2-\delta
\end{bmatrix}
\begin{bmatrix}
\hat{a}_1 \\
\hat{a}_k \\
\hat{a}_n
\end{bmatrix}
+ 
\frac{-(a_1^{n+1} - \hat{a}_k^{n+1})}{2+\delta(a_1^{n+1} + \hat{a}_k^{n+1})}
\]

\((k=2, 3, \ldots, ne)\) \hspace{1cm} (5.37)

which is exactly the same set of equations as the set derived by using the method of Galerkin (equation (5.20)).
Fig. 5. 15 A scalar wave moving with varying timesteps
Plottings for every two hours.
CHAPTER 6

THE ONE DIMENSIONAL CASE, A LONG PERIOD SHALLOW WATER WAVE IN A CHANNEL WITH A RECTANGULAR CROSS-SECTION USING AS DISTANCE INTEGRATION THE METHOD OF GALERKIN AND FOR TIME INTEGRATION THE FOUR-STEP METHOD OF RUNGA-KUTTA

6.1 General Conditions

The linearised basic equations follow from chapter 2:

Mass equation

\[
\frac{\partial h}{\partial t} + a_i \frac{\partial Q}{\partial x} = 0
\]

Momentum equation

\[
\frac{\partial Q}{\partial t} + b_i \frac{\partial Q}{\partial x} + c_i \frac{\partial h}{\partial x} + d_i Q = 0
\]

The shape functions are the so-called 'hat functions'

The weighting functions are following the method of Galerkin. Here the linear hatfunctions are used.

The boundary data are of the Dirichlet-type. At the lefthand boundary the value of \( h \) and at the right hand boundary the value of \( Q \) is given.

The initial values are given at each nodal point for \( Q \) and \( h \).

For the mass- as well as for the momentum equation the complete set of weighting functions using the method of Galerkin is:

\[
\phi = \sum_{k=1}^{ne} \phi_k
\]

Taking the boundary conditions into account, the set of weighting functions becomes:
For the mass equation:
\[ \phi_2, \phi_3, \ldots, \phi_{ne} \]

For the momentum equation:
\[ \phi_1, \phi_2, \ldots, \phi_{ne-1} \]

Because two equations will be deleted, there is, together with the two required boundary data, a set of \(2\times ne\) independent equations, which equals the number of dependent variables (\(2\times ne\)). So the system is solvable.

The time derivative. For time integration a four step Runge Kutta method is used.

In the next chapters first the set of equations will be derived without taking into account the boundary conditions. This means that the set of equations is also useable for a case with other types of boundary data. (for example \(Q_{\text{left}}\) and \(Q_{\text{right}}\) given). At last also the boundary conditions are taken into account.

6.2 The General Set of Equations

Following the method of Galerkin the general equations become

Mass equation

\[
\int_{\text{region}} \phi_k \left( \frac{\partial \phi}{\partial t} + a_i \frac{\partial \phi}{\partial x} \right) \, dx = 0 \\
(k=1, 2, \ldots, ne) \quad (6.1)
\]

Momentum equation

\[
\int_{\text{region}} \phi_k \left( \frac{\partial \phi}{\partial t} + b_i \frac{\partial \phi}{\partial x} + c_i \frac{\partial f}{\partial x} + d_i \phi \right) \, dx = 0 \\
(k=1, 2, \ldots, ne) \quad (6.2)
\]

In the following the \(\wedge\) marks, noting that an approximated value is used, will be omitted.

In this set of equations is assumed that the dependent variables \(Q\) and \(h\) as well as the coefficients \(a_i, b_i, c_i\) and \(d_i\) are defined in every nodal point \(k\).
According to the method of Galerkin, from the mass equation as well as from the momentum equation, two sets of ne equations can be derived. Both sets have the general shape

**Mass equation**
\[ M' \dot{\mathbf{u}} = \mathbf{S}' \mathbf{u} + \mathbf{f}' \]

**Momentum equation**
\[ M' \dot{\mathbf{Q}} = \mathbf{S}''(\mathbf{Q}, \mathbf{u}) + \mathbf{f}' \]

These two sets of equations are combined to the general shape

\[ \mathbf{M} \dot{\mathbf{u}} = \mathbf{S} \mathbf{u} + \mathbf{f} \]

or

\[ \mathbf{M}(\mathbf{Q}, \mathbf{u}) = \mathbf{S}(\mathbf{Q}, \mathbf{u}) + \mathbf{f} \]

6.3 **The Mass Equation**

The general mass equation using the method of Galerkin was (6.1)

\[
\int_{\text{region}} \phi_k \left( \frac{\partial}{\partial t} + a_i \frac{\partial}{\partial x_i} \right) dV = 0 \quad (k=1, 2, \ldots, ne) \quad (6.1)
\]

This gives the following set of equations:

\[
\begin{bmatrix}
\int_{\text{region}} \phi_1 \left( \frac{\partial}{\partial t} + a_i \frac{\partial}{\partial x_i} \right) dV = 0 \\
\int_{\text{region}} \phi_k \left( \frac{\partial}{\partial t} + a_i \frac{\partial}{\partial x_i} \right) dV = 0 \\
\int_{\text{region}} \phi_{ne} \left( \frac{\partial}{\partial t} + a_i \frac{\partial}{\partial x_i} \right) dV = 0 \\
\end{bmatrix} \quad (k=1, 2, \ldots, ne) \quad (6.3)
\]
Because the function $\phi_k$ is defined only in the elements $\Delta_{k-1}$ and $\Delta_k$, and elsewhere equals zero, the set of equations (6.3) can be written as:

$$
\int_{\Delta_{k-1} \cup \Delta_k} \phi_k \left( \frac{\partial}{\partial t} \left( \sum_{k=1}^{ne} \phi_k \cdot h_k \right) + \sum_{b=1}^{ne} \phi_b \cdot a_{b,k} \cdot \frac{\partial}{\partial x} \left( \sum_{k=1}^{ne} \phi_k \cdot Q_k \right) \right) \, dx = 0
$$

(k=1, 2, \ldots, ne) (6.4)

The $Q$ and $h$ as well as the coefficients can be written as trial functions defined over the whole region, and so as a function of $x$:

$$
Q = \sum_{k=1}^{ne} \phi_k(x) \cdot Q_k(t) \quad ; \quad h = \sum_{k=1}^{ne} \phi_k(x) \cdot h_k(t) \quad \forall \epsilon
$$

The set of equations becomes now:

$$
\begin{bmatrix}
\int_{\Delta_1} \phi_1 \left( \frac{\partial}{\partial t} \left( \sum_{k=1}^{ne} \phi_k \cdot h_k \right) + \sum_{b=1}^{ne} \phi_b \cdot a_{b,1} \cdot \frac{\partial}{\partial x} \left( \sum_{k=1}^{ne} \phi_k \cdot Q_k \right) \right) \, dx = 0 \\
\int_{\Delta_{n-1} \cup \Delta_n} \phi_{n-1} \left( \frac{\partial}{\partial t} \left( \sum_{k=1}^{ne} \phi_k \cdot h_k \right) + \sum_{b=1}^{ne} \phi_b \cdot a_{b,n-1} \cdot \frac{\partial}{\partial x} \left( \sum_{k=1}^{ne} \phi_k \cdot Q_k \right) \right) \, dx = 0 \\
\int_{\Delta_{ne-1}} \phi_{ne-1} \left( \frac{\partial}{\partial t} \left( \sum_{k=1}^{ne} \phi_k \cdot h_k \right) + \sum_{b=1}^{ne} \phi_b \cdot a_{b,ne-1} \cdot \frac{\partial}{\partial x} \left( \sum_{k=1}^{ne} \phi_k \cdot Q_k \right) \right) \, dx = 0
\end{bmatrix}
$$

(k=1, 2, \ldots, ne) (6.5)

This written out gives, because $\frac{\partial \phi_k}{\partial t} = 0$ and $\frac{\partial Q_k}{\partial x} = 0$
\[
\int \left( \left( \phi_i \phi_j \frac{\partial \phi_j}{\partial t} + \phi_i \phi_j \frac{\partial \phi_i}{\partial x} \right) + \\
+ \left( \phi_i \phi_j \frac{\partial \phi_j}{\partial x} a_i Q_j + \phi_i \phi_j \frac{\partial \phi_i}{\partial x} a_j Q_i \right) + \\
+ \left( \phi_i \phi_j \frac{\partial \phi_j}{\partial x} a_i Q_j + \phi_i \phi_j \frac{\partial \phi_i}{\partial x} a_j Q_i \right) \right) dx = 0
\]

\[
\int \left( \left( \phi_k \phi_k a_k Q_k + \phi_k \phi_k a_k Q_k + \phi_k \phi_k a_k Q_k \right) + \\
+ \left( \phi_k \phi_k a_k Q_k + \phi_k \phi_k a_k Q_k + \phi_k \phi_k a_k Q_k \right) + \\
+ \left( \phi_k \phi_k a_k Q_k + \phi_k \phi_k a_k Q_k + \phi_k \phi_k a_k Q_k \right) \right) dx = 0
\]

\[
\int \left( \left( \phi_{ne} \phi_{ne} a_{ne-1} Q_{ne-1} + \phi_{ne} \phi_{ne} a_{ne-1} Q_{ne-1} \right) + \\
+ \left( \phi_{ne} \phi_{ne} a_{ne-1} Q_{ne-1} + \phi_{ne} \phi_{ne} a_{ne-1} Q_{ne-1} \right) + \\
+ \left( \phi_{ne} \phi_{ne} a_{ne-1} Q_{ne-1} + \phi_{ne} \phi_{ne} a_{ne-1} Q_{ne-1} \right) \right) dx = 0
\]

\( (k=2, 3, \ldots, ne-1) \)  \hspace{1cm} (6.6)
fig. 6.1 The mass equation using the method of Galerkin

\[
\begin{bmatrix}
\int \phi \phi_1 \, dx & \int \phi \phi_2 \, dx & \cdots \\
\int \phi_1 \phi_1 \, dx & \int \phi_1 \phi_2 \, dx & \cdots \\
\int \phi_n \phi_1 \, dx & \int \phi_n \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_1 \, dx & \int \phi_{n-1} \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_{n-1} \, dx & \int \phi_{n-1} \phi_{n-2} \, dx & \cdots \\
\int \phi \phi_n \, dx & \int \phi \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_n \, dx & \int \phi_2 \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_2 \, dx & \int \phi_2 \phi_3 \, dx & \cdots \\
\int \phi_3 \phi_3 \, dx & \int \phi_3 \phi_4 \, dx & \cdots \\
\int \phi_n \phi_n \, dx & \int \phi_n \phi_{n-1} \, dx & \cdots
\end{bmatrix}
\begin{bmatrix}
\frac{\partial Q_1}{\partial t} \\
\frac{\partial Q_2}{\partial t} \\
\vdots \\
\frac{\partial Q_{n-1}}{\partial t} \\
\frac{\partial Q_n}{\partial t}
\end{bmatrix} +
\begin{bmatrix}
\int \phi_1 \phi_1 \, dx & \int \phi_1 \phi_2 \, dx & \cdots \\
\int \phi_2 \phi_1 \, dx & \int \phi_2 \phi_2 \, dx & \cdots \\
\int \phi_n \phi_1 \, dx & \int \phi_n \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_1 \, dx & \int \phi_{n-1} \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_{n-1} \, dx & \int \phi_{n-1} \phi_{n-2} \, dx & \cdots \\
\int \phi \phi_n \, dx & \int \phi \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_n \, dx & \int \phi_2 \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_2 \, dx & \int \phi_2 \phi_3 \, dx & \cdots \\
\int \phi_3 \phi_3 \, dx & \int \phi_3 \phi_4 \, dx & \cdots \\
\int \phi_n \phi_n \, dx & \int \phi_n \phi_{n-1} \, dx & \cdots
\end{bmatrix}
\begin{bmatrix}
\frac{\partial \bar{Q}_1}{\partial x} \\
\frac{\partial \bar{Q}_2}{\partial x} \\
\vdots \\
\frac{\partial \bar{Q}_{n-1}}{\partial x} \\
\frac{\partial \bar{Q}_n}{\partial x}
\end{bmatrix} +
\begin{bmatrix}
\int \phi_1 \phi_1 \, dx & \int \phi_1 \phi_2 \, dx & \cdots \\
\int \phi_2 \phi_1 \, dx & \int \phi_2 \phi_2 \, dx & \cdots \\
\int \phi_n \phi_1 \, dx & \int \phi_n \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_1 \, dx & \int \phi_{n-1} \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_{n-1} \, dx & \int \phi_{n-1} \phi_{n-2} \, dx & \cdots \\
\int \phi \phi_n \, dx & \int \phi \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_n \, dx & \int \phi_2 \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_2 \, dx & \int \phi_2 \phi_3 \, dx & \cdots \\
\int \phi_3 \phi_3 \, dx & \int \phi_3 \phi_4 \, dx & \cdots \\
\int \phi_n \phi_n \, dx & \int \phi_n \phi_{n-1} \, dx & \cdots
\end{bmatrix}
\begin{bmatrix}
\frac{\partial \bar{Q}_1}{\partial x} \\
\frac{\partial \bar{Q}_2}{\partial x} \\
\vdots \\
\frac{\partial \bar{Q}_{n-1}}{\partial x} \\
\frac{\partial \bar{Q}_n}{\partial x}
\end{bmatrix} +
\begin{bmatrix}
\int \phi_1 \phi_1 \, dx & \int \phi_1 \phi_2 \, dx & \cdots \\
\int \phi_2 \phi_1 \, dx & \int \phi_2 \phi_2 \, dx & \cdots \\
\int \phi_n \phi_1 \, dx & \int \phi_n \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_1 \, dx & \int \phi_{n-1} \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_{n-1} \, dx & \int \phi_{n-1} \phi_{n-2} \, dx & \cdots \\
\int \phi \phi_n \, dx & \int \phi \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_n \, dx & \int \phi_2 \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_2 \, dx & \int \phi_2 \phi_3 \, dx & \cdots \\
\int \phi_3 \phi_3 \, dx & \int \phi_3 \phi_4 \, dx & \cdots \\
\int \phi_n \phi_n \, dx & \int \phi_n \phi_{n-1} \, dx & \cdots
\end{bmatrix}
\begin{bmatrix}
\frac{\partial \bar{Q}_1}{\partial x} \\
\frac{\partial \bar{Q}_2}{\partial x} \\
\vdots \\
\frac{\partial \bar{Q}_{n-1}}{\partial x} \\
\frac{\partial \bar{Q}_n}{\partial x}
\end{bmatrix}
\begin{bmatrix}
\int \phi_1 \phi_1 \, dx & \int \phi_1 \phi_2 \, dx & \cdots \\
\int \phi_2 \phi_1 \, dx & \int \phi_2 \phi_2 \, dx & \cdots \\
\int \phi_n \phi_1 \, dx & \int \phi_n \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_1 \, dx & \int \phi_{n-1} \phi_2 \, dx & \cdots \\
\int \phi_{n-1} \phi_{n-1} \, dx & \int \phi_{n-1} \phi_{n-2} \, dx & \cdots \\
\int \phi \phi_n \, dx & \int \phi \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_n \, dx & \int \phi_2 \phi_{n-1} \, dx & \cdots \\
\int \phi_2 \phi_2 \, dx & \int \phi_2 \phi_3 \, dx & \cdots \\
\int \phi_3 \phi_3 \, dx & \int \phi_3 \phi_4 \, dx & \cdots \\
\int \phi_n \phi_n \, dx & \int \phi_n \phi_{n-1} \, dx & \cdots
\end{bmatrix}
\begin{bmatrix}
\frac{\partial \bar{Q}_1}{\partial x} \\
\frac{\partial \bar{Q}_2}{\partial x} \\
\vdots \\
\frac{\partial \bar{Q}_{n-1}}{\partial x} \\
\frac{\partial \bar{Q}_n}{\partial x}
\end{bmatrix} = 0
\]

(k=2, 3, ..., ne-1)  (6.7)
This set of \((n_e)\) equations contains the dependent variables or their
derivatives \(\frac{dx}{dt}\) and \(Q\).
Further writing out of equation (6.6) gives a set of equations of the
general shape

\[
M' \ddot{\mathbf{a}} + S' \mathbf{a} = \mathbf{f}'
\]
or for this case

\[
M' \ddot{\mathbf{a}} + S' \mathbf{Q} = \mathbf{f}'
\]

where

\(M = \) so called Mass matrix
\(S = \) so called Stiffness matrix
\(\mathbf{f} = \) vector which contains only 'constant' terms.

The terms Mass- and Stiffness matrix are from Structure Analyses.
In fig 6.1 this set of equations is written out as a matrix.

6.4 The Momentum Equation

Although a bit more complicated than the mass equation, the building
up of the set of equations of the momentum equation is the same. Because
of completeness the whole cycle is worked out again for the momentum
equation.

From the general equation

\[
\int_{\text{region}} \varphi_k \left( \frac{\partial Q}{\partial t} + b_i \frac{\partial Q}{\partial x} + c_i \frac{\partial Q}{\partial x} + d_i Q \right) dx = 0 \quad (k=1, 2, \ldots, n_e) \tag{6.2}
\]
a set of equations is derived:

\[
\begin{bmatrix}
\int_{\Delta x} \varphi_1 \left( \frac{\partial Q}{\partial t} + b_1 \frac{\partial Q}{\partial x} + c_1 \frac{\partial Q}{\partial x} + d_1 Q \right) dx = 0 \\
\vdots \\
\int_{\Delta x} \varphi_k \left( \frac{\partial Q}{\partial t} + b_k \frac{\partial Q}{\partial x} + c_k \frac{\partial Q}{\partial x} + d_k Q \right) dx = 0 \\
\vdots \\
\int_{\Delta x} \varphi_{n_e} \left( \frac{\partial Q}{\partial t} + b_{n_e} \frac{\partial Q}{\partial x} + c_{n_e} \frac{\partial Q}{\partial x} + d_{n_e} Q \right) dx = 0
\end{bmatrix} 
\]

\((k=2, 3, \ldots, n_e-1) \tag{6.8}\)
\[
\int \left( \left( a_{i,j,k} \frac{\partial^2 w}{\partial x^2} + a_{i,j,k} \frac{\partial^2 w}{\partial y^2} + a_{i,j,k} \frac{\partial^2 w}{\partial z^2} \right) + \left( b_{i,j,k} \frac{\partial w}{\partial x} + b_{i,j,k} \frac{\partial w}{\partial y} + b_{i,j,k} \frac{\partial w}{\partial z} \right) \right) dx = 0
\]

\[
\int \left( \left( a_{i,j,k} \frac{\partial^2 w}{\partial x^2} + a_{i,j,k} \frac{\partial^2 w}{\partial y^2} + a_{i,j,k} \frac{\partial^2 w}{\partial z^2} \right) + \left( b_{i,j,k} \frac{\partial w}{\partial x} + b_{i,j,k} \frac{\partial w}{\partial y} + b_{i,j,k} \frac{\partial w}{\partial z} \right) \right) dx = 0
\]

\[
\int \left( \left( a_{i,j,k} \frac{\partial^2 w}{\partial x^2} + a_{i,j,k} \frac{\partial^2 w}{\partial y^2} + a_{i,j,k} \frac{\partial^2 w}{\partial z^2} \right) + \left( b_{i,j,k} \frac{\partial w}{\partial x} + b_{i,j,k} \frac{\partial w}{\partial y} + b_{i,j,k} \frac{\partial w}{\partial z} \right) \right) dx = 0
\]

\[
\int \left( \left( a_{i,j,k} \frac{\partial^2 w}{\partial x^2} + a_{i,j,k} \frac{\partial^2 w}{\partial y^2} + a_{i,j,k} \frac{\partial^2 w}{\partial z^2} \right) + \left( b_{i,j,k} \frac{\partial w}{\partial x} + b_{i,j,k} \frac{\partial w}{\partial y} + b_{i,j,k} \frac{\partial w}{\partial z} \right) \right) dx = 0
\]

(\text{for } k=2, 3, \ldots, \text{ne}-1) \quad (6.9)
This set of equations can also be written in a general shape:

\[
[M''J[M] + [S''J][Q, q] = \bar{f}'
\]

6.5 Combining the Mass- and Momentum Equation

When the mass- and the momentum equations are combined, the set

\[
[k_{\text{mass}} - \text{shape}]
\quad + 
\quad \begin{bmatrix}
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
  \vdots \\
 \end{bmatrix}
\quad = 
\quad \begin{bmatrix}
  0 \\
  0 \\
  0 \\
  0 \\
  0 \\
\end{bmatrix}
\]

\[
(\text{k}=2, 3, \ldots, \text{ne}-1)
\]  \quad (6.10)

in which the identifier k denotes which shape function \( \phi_k \) has been used for the equation. Note that k can have the values 1, 2, \ldots, ne or the values 2, 3, \ldots, ne-1, what is the same because in the first and the last equation k is written explicit as 1 or ne.
The matrix elements are the following:

\[
\begin{align*}
m_{i,1} &= m_{ne,1,ne,1} = \int_{\Delta_i} \phi_i \phi_i \, dx \\
m_{i,2} &= m_{ne,1,ne,2} = \int_{\Delta_i} \phi_i \phi_2 \, dx \\
m_{k,k} &= m_{ne+k,ne+k-1} = \int_{\Delta_{k-1}} \phi_k \phi_k \, dx \\
m_{k,k} &= m_{ne+k,ne+k} = \int_{\Delta_k} \phi_k \phi_k \, dx \\
m_{ne,ne} &= m_{ne,ne} = \int_{\Delta_{ne-1}} \phi_{ne} \phi_{ne-1} \, dx \\
m_{ne,ne} &= m_{ne,ne} = \int_{\Delta_{ne-1}} \phi_{ne} \phi_{ne-1} \, dx \\
\end{align*}
\]

\[
\left. \begin{array}{l}
\{ (k = 1, 2, \ldots, ne) \\
\end{array} \right. \\
\]

For the stiffness matrix the elements are:

\[
\begin{align*}
s_{i,ne,1} &= \int_{\Delta_i} \phi_i \phi_i \frac{\partial \phi}{\partial x} a_i \, dx + \int_{\Delta_i} \phi_i \phi_i \frac{\partial \phi}{\partial x} a_2 \, dx \\
s_{i,ne,2} &= \int_{\Delta_i} \phi_i \phi_2 \frac{\partial \phi}{\partial x} a_i \, dx + \int_{\Delta_i} \phi_i \phi_2 \frac{\partial \phi}{\partial x} a_2 \, dx \\
s_{k,ne+k-1} &= \int_{\Delta_{k-1}} \phi_k \phi_k \frac{\partial \phi}{\partial x} a_{k-1} \, dx + \int_{\Delta_{k-1}} \phi_k \phi_k \frac{\partial \phi}{\partial x} a_{k-2} \, dx \\
s_{k,ne+k} &= \int_{\Delta_k} \phi_k \phi_k \frac{\partial \phi}{\partial x} a_{k-1} \, dx + \int_{\Delta_k} \phi_k \phi_k \frac{\partial \phi}{\partial x} a_{k-2} \, dx \\
s_{k,ne+k} &= \int_{\Delta_k} \phi_k \phi_k \frac{\partial \phi}{\partial x} a_{k-1} \, dx + \int_{\Delta_k} \phi_k \phi_k \frac{\partial \phi}{\partial x} a_{k-2} \, dx \\
s_{ne,ne-1} &= \int_{\Delta_{ne-1}} \phi_{ne} \phi_{ne-1} \frac{\partial \phi_{ne-1}}{\partial x} a_{ne-1} \, dx + \int_{\Delta_{ne-1}} \phi_{ne} \phi_{ne-1} \frac{\partial \phi_{ne-1}}{\partial x} a_{ne-2} \, dx \\
s_{ne,ne} &= \int_{\Delta_{ne-1}} \phi_{ne} \phi_{ne-1} \frac{\partial \phi_{ne-1}}{\partial x} a_{ne-1} \, dx + \int_{\Delta_{ne-1}} \phi_{ne} \phi_{ne-1} \frac{\partial \phi_{ne-1}}{\partial x} a_{ne-2} \, dx \\
\end{align*}
\]

\[
\begin{align*}
s_{ne,ne} &= \int \phi_i \phi_i \frac{\partial \phi}{\partial x} b_1 \, dx + \int \phi_i \phi_i \frac{\partial \phi}{\partial x} b_2 \, dx + \\
&+ \int \phi_i \phi_i \phi_i \, dx + \int \phi_i \phi_i \phi_2 \, dx \\
s_{ne,ne} &= \int \phi_i \phi_i \frac{\partial \phi}{\partial x} b_1 \, dx + \int \phi_i \phi_i \phi_i \, dx + \\
&+ \int \phi_i \phi_i \phi_2 \, dx + \int \phi_i \phi_i \phi_2 \, dx \\
\end{align*}
\]
\[ s_{\text{mass}} = \int \phi_k \phi_k \frac{\partial^2 \psi}{\partial x^2} b_k \, dx + \int \phi_k \phi_k \frac{\partial b_k}{\partial x} \, dx \]
\[ + \int \phi_k \phi_k \phi_k b_k \, dx + \int \phi_k \phi_k \phi_k b_k \, dx \]
\[ s_{\text{stiff}} = \int \phi_k \phi_k \frac{\partial^2 \psi}{\partial x^2} b_k \, dx + \int \phi_k \phi_k \frac{\partial b_k}{\partial x} \, dx + \int \phi_k \phi_k \phi_k b_k \, dx + \int \phi_k \phi_k \phi_k b_k \, dx \]
\[ s_{\text{ne,mu}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]
\[ s_{\text{ne,ne}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]
\[ s_{\text{ne,ne}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]
\[ s_{\text{ne,ne}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]
\[ s_{\text{ne,ne}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]
\[ s_{\text{ne,ne}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]
\[ s_{\text{ne,ne}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]
\[ s_{\text{ne,ne}} = \int \phi_i \phi_i \frac{\partial \psi}{\partial x} c_i \, dx + \int \phi_i \phi_i \frac{\partial c_i}{\partial x} \, dx \]

Except the mass matrix and the stiffness matrix, also the vectors \( \frac{\partial q}{\partial x} \) and \( q \) are written explicitly. The vector \( \frac{\partial q}{\partial x} \) contains also the derivatives of \( h \) and \( Q_{\text{ne}} \). The vector \( q \) contains also the as boundary values given values of \( h \) and \( Q_{\text{ne}} \).

Because for the solving of the time problem the four-step method of Runge Kutta is used, can \( \frac{\partial h}{\partial t} \) be written in terms of \( h(t+\Delta t) \) and \( h(t) \) and in the same way can \( \frac{\partial Q_{\text{ne}}}{\partial t} \) be written in terms of \( Q_{\text{ne}}(t+\Delta t) \) and \( Q_{\text{ne}}(t) \).

This will be explained in chapter 6.6.
6.6 The Method of Runge Kutta for Time Integration

The general set of equations can be written as

\[ M \cdot \dot{\mathbf{a}} + S \cdot \mathbf{a} = \mathbf{f} \]

or

\[ M \cdot \dot{\mathbf{a}} = -S \cdot \mathbf{a} + \mathbf{f} \]

or

\[ \dot{\mathbf{a}} = M^{-1}(-S \cdot \mathbf{a} + \mathbf{f}) \quad (6.11) \]

This means that the time derivative can be written explicitly.

From (4.16 and 4.17) can be seen that, when the method of Runge Kutta is used, the general set of equations becomes

\[ M'(\mathbf{a}_n - \mathbf{a}^n) = \frac{\Delta t}{2}(-S \cdot \mathbf{a}_n + \mathbf{f}_n) \Rightarrow \mathbf{a}_n \Rightarrow \mathbf{f}_n \]

\[ M'(\mathbf{a}^n - \mathbf{a}_n) = \frac{\Delta t}{2}(-S \cdot \mathbf{a}_n + \mathbf{f}_n) \Rightarrow \mathbf{a}^n \Rightarrow \mathbf{f}^n \]

\[ M'(\mathbf{a}^{n+1} - \mathbf{a}_n) = \Delta t(-S \cdot \mathbf{a}^{n+1} + \mathbf{f}_n) \Rightarrow \mathbf{a}^{n+1} \Rightarrow \mathbf{f}^{n+1} \]

\[ \mathbf{a}^{n+1} = \frac{1}{3} \mathbf{a}_n + \frac{2}{3} \mathbf{a}^n + \frac{1}{3} \mathbf{a}_{n+1} + \frac{1}{6} \mathbf{a}^{n+1} - \frac{1}{2} \mathbf{a}^n \quad (6.12) \]

Note that the matrix \( M \) is only a function of the shape function \( \mathbf{x} \) and so time independent. The matrix \( S \) is a function of the coefficients \( a, b, c \) and \( d \) and so time dependent.

As can be seen from equation (6.12), can the value of \( \dot{\mathbf{a}} \) for every new time level be calculated. This is worked out in the next chapters.
6.7 **The Final Set of Equations** (boundary conditions not taken into account)

For the first Runga Kutta step, the set of equations (6.10) can be written as:

\[
\begin{bmatrix}
  m & m & m & 0 \\
  m & m & m & 0 \\
  0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
  \eta_1^o \\
  \eta_2^o \\
  \eta_3^o \\
  \eta_4^o \\
\end{bmatrix}
= \begin{bmatrix}
  Q_1^o \\
  Q_2^o \\
  Q_3^o \\
  Q_4^o \\
\end{bmatrix}
\]

Writing out equation (6.13) gives:

\[
\begin{bmatrix}
  \Delta t \\
  \Delta t \\
  \Delta t \\
  \Delta t \\
\end{bmatrix}
\begin{bmatrix}
  \eta_1^o \\
  \eta_2^o \\
  \eta_3^o \\
  \eta_4^o \\
\end{bmatrix}
+ \frac{\Delta t}{2} \begin{bmatrix}
  0 \\
  0 \\
  0 \\
  0 \\
\end{bmatrix}
= \begin{bmatrix}
  Q_1^o \\
  Q_2^o \\
  Q_3^o \\
  Q_4^o \\
\end{bmatrix}
\]

(6.13)
or simply

\[ [M] [g \left( h^*, \ldots, h^*_ne, Q^*_1, \ldots, Q^*_ne \right) - g \left( h^n, \ldots, h^n_ne, Q^n_1, \ldots, Q^n_ne \right)] = [F(h^n, h^n_ne, Q^n_1, Q^n_2, \ldots, Q^n_ne)] \]

\[ (6.15) \]

or

\[ M \cdot X = Y \]

\[ (6.16) \]

where

- \( M = 2 * ne \) by \( 2 * ne \) three diagonal banded matrix, only dependent of \( x \)
- \( X = 2 * ne \) vector, containing \( h^*_1 - h^n_1, \ldots, h^*_ne - h^n_ne, Q^*_1 - Q^n_1, \ldots, Q^*_ne - Q^n_ne \)
- \( Y = 2 * ne \) vector, containing terms with 'known' values.
To solve the vector \( X \) explicitly from equation (6.16), two procedures are used, called BANSOLA and BANDETA (see chapter 6.10)

For the first Runga Kutta step follows from

\[
M (a^* - a^n) = \dot{y}^n
\]

(6.17)

the value of \( a^* \) etc.

The whole system can be written now as

\[
\begin{align*}
M (a^* - a^n) &= \dot{y}^n \\
M (a^{**} - a^n) &= \dot{y}^{**} \\
M (a^{***} - a^n) &= \dot{y}^{***} \\
M (a^{****} - a^n) &= \dot{y}^{****}
\end{align*}
\]

(6.18)

Which stepwise can be solved.

\[
\Rightarrow \quad a^{n+1} = \frac{1}{3} a^* + \frac{2}{3} a^{**} + \frac{1}{3} a^{***} + \frac{1}{6} a^{****} - \frac{1}{6} a^n
\]

(6.19)

6.8 The System and the Boundary Conditions

For solving the long period shallow water wave equations, two boundary conditions are required. This gives two more equations to the total set and so an over-determined system. This means that two equations must be deleted. As mentioned already in chapter 3, the equations that can be deleted must be related as strongly as possible to the given boundary conditions. In practice this means that the first and the last equations are the ones that possibly can be deleted.

The given two boundary conditions can be of a different type:

\[
\begin{align*}
\text{a)} \quad \text{At each boundary } Q \text{ or } h. \\
\text{b)} \quad \text{At each boundary only } h. \\
\text{c)} \quad \text{At each boundary only } Q.
\end{align*}
\]

a) For the \( Q \) and \( h \) given boundary condition (let us say \( h \) and \( Q_{ne} \), the
mass equation using as weighting function \( \phi_1 \) as well as the momentum equation using as weighting function \( \phi_{ne} \) can be deleted.

b) For a h-h given boundary condition, the mass equations using as weighting functions \( \phi_1 \) and \( \phi_{ne} \) can be deleted.

c) For a \( Q-Q \) given boundary condition, the momentum equations using as weighting functions \( \phi_1 \) and \( \phi_{ne} \) can be deleted.

6.9 The final set of Equations Using Langrange Shape Functions and Considering the Boundary Conditions

The used shape functions are the so-called hat functions.

The element lengths are variable so

\[ \phi_1 = \phi_{k,1} = \frac{\Delta_1 - x}{\Delta_1} \; ; \; \frac{\partial \phi_1}{\partial x} = \frac{\partial \phi_{k,1}}{\partial x} = -1 \]

\[ \phi_k = \phi_{k, k-1} + \phi_{k, k} = \frac{x}{\Delta_{k-1}} + \frac{1-x}{\Delta_k} \quad (k=2,3,\ldots,ne-1) \]

\[ \frac{\partial \phi_k}{\partial x} = \frac{\partial \phi_{k, k-1}}{\partial x} + \frac{\partial \phi_{k, k}}{\partial x} = \frac{1}{\Delta_{k-1}} + \frac{-1}{\Delta_k} \quad (k=2,3,\ldots,ne-1) \]

\[ \phi_{ne} = \frac{x}{\Delta_{ne-1}} \; ; \; \frac{\partial \phi_{ne}}{\partial x} = \frac{1}{\Delta_{ne-1}} \]

Substituting \( \phi_1, \phi_2, \ldots, \phi_{ne} \) into the matrix elements \( m \) and \( s \) of equation (6.13) and working out these integrals gives:
The mass- and stiffness matrix elements for the one-dimensional case using linear Lagrange shape functions:

\[
\begin{align*}
\mathbf{m}_{ij} &= m_{\text{ne},i+1,\text{ne},j+1} = \frac{A_i}{3} \\
\mathbf{m}_{ij} &= m_{\text{ne},i+1,\text{ne},j+1} = \frac{A_i}{6} \\
\mathbf{m}_{kk} &= m_{\text{ne},i,\text{ne},k+1} = \frac{\Delta k}{6} \\
\mathbf{m}_{kk} &= m_{\text{ne},i,\text{ne},k+1} = \frac{\Delta k}{3} + \frac{\Delta k}{3} \\
\mathbf{m}_{ki} &= m_{\text{ne},i,\text{ne},k+1} = \frac{\Delta k}{6} \\
\mathbf{m}_{\text{ne},i} &= m_{\text{ne},i+1,\text{ne},i+1} = \frac{\Delta \text{ne}-1}{6} \\
\mathbf{m}_{\text{ne},\text{ne}} &= m_{\text{ne},\text{ne}} = \frac{\Delta \text{ne}-1}{3}
\end{align*}
\]

\[
\begin{align*}
\mathbf{s}_{\text{ne},i} &= -\frac{1}{2} a_i - \frac{1}{6} a_l \\
\mathbf{s}_{\text{ne},i} &= \frac{1}{2} a_i + \frac{1}{6} a_l \\
\mathbf{s}_{\text{ne},k} &= -\frac{1}{6} a_{k-1} - \frac{1}{3} a_k \\
\mathbf{s}_{\text{ne},k} &= \frac{1}{6} a_{k-1} - \frac{1}{6} a_{k+1} \\
\mathbf{s}_{\text{ne},k} &= \frac{1}{3} a_k + \frac{1}{6} a_{k+1} \\
\mathbf{s}_{\text{ne},k} &= \frac{1}{6} a_{k-1} - \frac{1}{3} a_{k+1} \\
\mathbf{s}_{\text{ne},\text{ne}} &= \frac{1}{6} a_{\text{ne}} + \frac{1}{3} a_{\text{ne}} \\
\mathbf{s}_{\text{ne},\text{ne}} &= -\frac{1}{6} a_{\text{ne}} - \frac{1}{3} a_{\text{ne}} \\
\mathbf{s}_{\text{ne},\text{ne}} &= \frac{1}{6} a_{\text{ne}} + \frac{1}{3} a_{\text{ne}}
\end{align*}
\]

\[
\begin{align*}
\mathbf{s}_{\text{ne},i} &= -\frac{1}{3} b_i - \frac{1}{6} b_2 + \frac{A_{i+1}}{12} d_1 + \frac{A_i}{12} d_2 \\
\mathbf{s}_{\text{ne},i} &= \frac{1}{3} b_i + \frac{1}{6} b_2 + \frac{A_{i+1}}{12} d_1 + \frac{A_i}{12} d_2 \\
\mathbf{s}_{\text{ne},k} &= -\frac{1}{6} b_{k-1} + \frac{1}{3} b_k + \frac{A_{k-1}}{12} d_{k-1} + \frac{A_k}{12} d_k \\
\mathbf{s}_{\text{ne},k} &= \frac{1}{6} b_{k-1} - \frac{1}{6} b_{k+1} + \frac{A_{k-1}}{12} d_{k-1} + \left(\frac{A_{k-1}}{12} + \frac{A_k}{12}\right) d_k + \frac{A_k}{12} d_{k+1} \\
\mathbf{s}_{\text{ne},k} &= \frac{1}{6} b_k + \frac{1}{6} b_{k+1} + \frac{A_k}{12} d_k + \frac{A_k}{12} d_{k+1} \\
\mathbf{s}_{\text{ne},\text{ne}} &= -\frac{1}{6} b_{\text{ne}-1} - \frac{1}{3} b_{\text{ne}} + \frac{A_{\text{ne}-1}}{12} d_{\text{ne}-1} + \frac{A_{\text{ne}}}{12} d_{\text{ne}} \\
\mathbf{s}_{\text{ne},\text{ne}} &= \frac{1}{6} b_{\text{ne}-1} + \frac{1}{3} b_{\text{ne}} + \frac{A_{\text{ne}-1}}{12} d_{\text{ne}-1} + \frac{A_{\text{ne}}}{4} d_{\text{ne}}
\end{align*}
\]
Substituting now these matrix elements into equation (6.14) gives:

\[
\begin{bmatrix}
\begin{array}{ccc}
  a_1 & 0 & \cdots \\
  0 & a_2 & \cdots \\
  \vdots & \vdots & \ddots \\
  0 & 0 & \cdots & a_n
\end{array}
\end{bmatrix}
\begin{bmatrix}
  \phi_1 \\
  \phi_2 \\
  \vdots \\
  \phi_n
\end{bmatrix}
= 
\begin{bmatrix}
  \beta_1 \\
  \beta_2 \\
  \vdots \\
  \beta_n
\end{bmatrix}
\]

\[
\begin{align*}
&\begin{bmatrix}
  \frac{1}{2} a_{n-1} - \frac{1}{2} a_n \cdot Q_1^n + \left( \frac{1}{2} a_{n-1} + \frac{1}{2} a_n \right) Q_2^n \\
  \left( \frac{1}{2} a_{n-1} - \frac{1}{2} a_n \right) Q_{n-1}^n + \left( \frac{1}{2} a_{n-1} + \frac{1}{2} a_n \right) Q_n^n \\
  \vdots \\
  \left( \frac{1}{2} a_{n-1} - \frac{1}{2} a_n \right) Q_{n-2}^n + \left( \frac{1}{2} a_{n-1} + \frac{1}{2} a_n \right) Q_{n-1}^n \\
\end{bmatrix} \\
&= -\frac{\Delta t}{2}
\end{align*}
\]

\[
\begin{align*}
&\begin{bmatrix}
  \left( \frac{1}{2} b_{n-1} - \frac{1}{2} b_n \right) L_{n-1}^n + \left( \frac{1}{2} b_{n-1} + \frac{1}{2} b_n \right) L_n^n + \left( \frac{\Delta b_n}{\Delta t} \right) L_{n-1}^n \\
  \left( \frac{1}{2} b_{n-1} - \frac{1}{2} b_n \right) L_{n-1}^n + \left( \frac{1}{2} b_{n-1} + \frac{1}{2} b_n \right) L_n^n + \left( \frac{\Delta b_n}{\Delta t} \right) L_{n-1}^n \\
  \vdots \\
  \left( \frac{1}{2} b_{n-1} - \frac{1}{2} b_n \right) L_{n-1}^n + \left( \frac{1}{2} b_{n-1} + \frac{1}{2} b_n \right) L_n^n + \left( \frac{\Delta b_n}{\Delta t} \right) L_{n-1}^n \\
\end{bmatrix} \\
&= \left( k=2, 3, \ldots, n+1 \right) (6.20)
\end{align*}
\]

Taking now the boundary conditions into account, (see Chapter 6.1 and 6.8), the first equation and the last one can be omitted.

The equation (6.13) becomes:

\[
( k=2, 3, \ldots, n+1 )
\]
Bringing the known dependent variables to the right hand terms, the set of equations (6.20) becomes after multiplication by 12:

\[
\begin{align*}
&\begin{bmatrix}
\Delta t
\end{bmatrix}
\end{align*}
\]
This equation can be written again as:

\[ M \left( \mathbf{a}^n - \mathbf{a}^n \right) = \mathbf{y}^n \quad (6.17) \]

or

\[ M \cdot \mathbf{x} = \mathbf{y} \quad (6.16) \]

6.10 **The Program and the Flow Diagram**

What the program is doing in fact is solving from

\[ M \cdot \mathbf{x} = \mathbf{y} \quad (6.16) \]

the value of \( \mathbf{x} \).

We know that the basic equations, using the method of Galerkin and the four-step method of Runge-Kutta, can be written as a set of equations:
\[ M \cdot (a^* - a^n) = y^n \]  

(6.17)

The elements of the matrix \( M \) can be calculated from the element lengths \( \Delta_k \) (\( k = 1, 2, \ldots, ne-1 \); \( ne \) = number of nodal points)

The elements of the vector \( a^n \) are known (initial values or calculated values on the old time level).

The elements of the vector \( y^n \) can be calculated from the coefficients of the basic equations and the values of \( a^n \).

For solving \( x \) from \( M \cdot x = y \), two procedures have been developed, called BANDET and BANSOL. These procedures have been published and discussed in the journal 'Numerischen Mathematik', translated from the computer language Algol into Fortran and also modified by the computational group of the mathematical division of the Technological University of Delft. For the problem, discussed in this report, both procedures have been modified and translated into PL-1 by the author of this report.

In order to distinguish the original versions from the used ones, the procedures used in this report are called BANDETA and BANSOLA. See the flow diagram.

The procedure Bandeta is:

1- Calculating the elements of the (using linear interpolation functions three diagonal) band-matrix \( M \) (\( 2* (ne-1) \), \( 2* (ne-1) \)).

2- Storing the band-matrix \( M \) as: an \((2* (ne-1), n) \) array \( A \) (\( 1:2* (ne-1), -m_1:m_1 \)), where \( m_1 \) = the number of sub- and/or super diagonals.

3- Factorizing the matrix \( A \) into the product of a lower triangular matrix and a super triangular matrix, using partial pivoting. The lower triangle is stored as an \( n \times m_1 \) array and the upper triangle is overwritten on \( A \). Details of the row inter-changes are stored in the array INTER.

Note that, because the element lengths are not changing, the procedure Bandeta is used only one time during the computation.

The procedure Bansola is:

1- Calculating the values of the vector \( y^n \)

2- Calculating the values of the vector \( x \) from \( M \cdot x^* = y^n \) or

\[ (L \cdot U) \cdot x^* = y^n \quad \text{or} \quad (L \cdot U) \cdot (a^* - a^n) = y^n \]

(6.18)

From this \( a^* \) is calculated and so the values of the dependent variables on the 'new' time level.
FLOW DIAGRAM FOR THE ONE_DIMENSIONAL CASE

BEGIN

NUMBER OF NODAL-POINTS (ne).
NUMBER OF UPPER/LOWER DIAGONALS (m1)

calc. NUMBER OF ELEMENTS (nf)

ELEMENT-LENGTHS (dx)
TIME-STEP (dt)

PROCEDURE BANDETA
a) calc. of MASS-matrix coeff.
b) transformation of MASS-matrix

CHEZY-VALUES IN EACH NODAL-POINT
WIDTH IN EACH NODAL-POINT
INITIAL DISCHARGE IN EACH NODAL POINT (Qn)
INITIAL WATERLEVEL IN EACH NODAL POINT (hn)

calc. of the function, which describes the BOUNDARY-CONDITIONS

PROCEDURE BANSOLA

calc. 1/3.Qn(1) and 1/3.hn(1)

BOUNDARY-CONDITIONS ON TIME-LEVEL t+dt/2

PROCEDURE BANSOLA

calc. 2/3.Qn(2) and 2/3.hn(2)

PROCEDURE BANSOLA

calc. 1/3.Qn(3) and 1/3.hn(3)

BOUNDARY-CONDITIONS ON TIME-LEVEL t+dt

PROCEDURE BANSOLA
CALCULATION OF DISCHARGE AND WATERLEVEL
ON NEW TIME-STEP

\[ Q_{NW} = \frac{1}{3}.Q_n^{(1)} + \frac{2}{3}.Q_n^{(2)} + \frac{1}{3}.Q_n^{(3)} + \frac{1}{6}.Q_n^{(4)} - \frac{1}{2}.Q_n \]

\[ H_{NW} = \frac{1}{3}.h_n^{(1)} + \frac{2}{3}.h_n^{(2)} + \frac{1}{3}.h_n^{(3)} + \frac{1}{6}.h_n^{(4)} - \frac{1}{2}.h_n \]

STOP CONDITION

\[ Q_n = Q_{NW} \]
\[ h_n = H_{NW} \]
6.11 Some results.

From stability calculations it follows (for the used set of equations), that the maximum possible time step is about:

\[
\Delta t < \frac{5 \Delta X}{3 \sqrt{v^2 + gh}}
\]

where

\( v \) = velocity of the water.

Calculations are made for three cases:

1) A 'hat-wave' entering a rectangular channel with horizontal bottom.
   Four runs have been made: with different time steps and neglecting the resistance as well as not neglecting the resistance.

2) A sinus shaped wave entering a rectangular channel with horizontal bottom.
   Two runs have been made with varying space step and time step.
   Resistance has been neglected.

3) A sinus shaped wave with a longer period than in case 2) has been plotted for every quarter of an hour. Resistance has been neglected.

Case 1)
Initial conditions:
Length of the channel: 123.000 m
number of nodal points: 42
number of elements: 41
water level: 10 m
discharge: 0 m³/sec
width: 200 m

Boundary conditions:
\[ h_1(t) : \begin{align*}
\text{t} < 0 & \quad h = h_{\text{initial}} \\
0 \leq t < 2250 \text{ s} & \quad h = h_{\text{initial}} + \frac{t}{2250} \\
2250 \leq t < 4500 \text{ s} & \quad h = h_{\text{initial}} + \frac{h_{\text{tidal}}}{2250} \left( t - \frac{2250}{2250} \right) \\
t \geq 4500 \text{ s} & \quad h = h_{\text{initial}}
\end{align*} \]

(see fig 6.3-a)
\[ Q_{\text{in}}(t) = 0 \text{ m}³/\text{sec}. \]
fig 6.3 -a, b, c  A hat-wave in a rectangular channel

\[ \Delta x = 3000 \text{ m} \quad \text{resistance neglected} \]

\[ \Delta t = 225 \text{ sec} \]

\[ \Delta t = 450 \text{ sec} \]
$T = 3 \text{ hr}$

$H(\text{m})$

$T = 3 \frac{3}{4} \text{ h}$

$H(\text{m})$

$T = 4 \text{ hrs}$

$\Delta x = 3000 \text{ m}$

$\Delta t = 225 \text{ sec}$: 

$\Delta t = 450 \text{ sec}$: 

$\Delta t = 75 \text{ sec}$: 

resistance neglected

fig. 6. 3 - c, d, e
fig 6.3 - f, g, h A sinus shaped wave in a rectangular channel.

Resistance neglected.
$H(m) = 12$

$T = 1 \text{ hr}$

$T = 2 \text{ hrs}$

$T = 3 \text{ hrs}$

$\Delta x = 3000 \text{ m}$

$\Delta t = 225 \text{ sec}$ and $\Delta t = 450 \text{ sec}$

fig. 6.3 - i, j, k
$H(m)$

$T = 4$ hrs

$T = 5$ hrs

$T = 6$ hrs

$\Delta x = 3000$ m

$\Delta t = 225$ and $450$ sec

Including resistance

Fig. 6.3 l, m, n A sinus shaped wave in a rectangular channel
First the calculation is done with C=0, so resistance neglected. This calculation is done for two timesteps: \( \Delta t = 450 \) sec and \( \Delta t = 225 \) sec. A plotting is made for every hour (during 6 hrs). In order to show the reflecting at the right hand boundary \( (Q = 0) \), an extra plot is made for \( T = 3 \frac{3}{4} \) hrs. At this time level also a plot is made for a calculation with time step \( \Delta t = 75 \) sec.

The maximum possible time step can be calculated from equation (6.17) \( \Delta t_{\text{max}} = 450 \) sec. See fig. 6. 3 b up to h. From these figures it seems that the timestep of 225 sec gives a less accurate description of the phenomenon. This needs more investigation.

Next the same calculations as mentioned above are made, but now the resistance is included. Plotting for the two timesteps gives nearly the same results and is not to be seen in the plots. See fig 6. 3 i up to n. Note that there is no calculation made on time level 3 3/4 hrs.

Case 2)
Initial conditions:
Length of the channel : 160,000 m
number of nodal points : 41 ; 21
number of elements : 40 ; 20
water level : 10 m
discharge : 0 m³/sec
width : 200 m

Boundary conditions:
\[ h_{\text{ne}}(t) = 10 + \sin 2\pi \frac{t}{T} \text{ m}, \ T = 3 \text{ hrs}. \]
\[ Q_{\text{i}}(t) = 0 \text{ m}^3/\text{sec} \]
Resistance neglected.

See fig 6. 4. A plot has been made for every hour for the two cases:
\[ \Delta x = 8000\text{m} \quad t = 600 \text{ sec} \quad \text{line } \]
\[ \Delta x = 4000\text{m} \quad t = 300 \text{ sec} \quad \text{line } \]

Also here further investigation is necessary.
\[ \Delta x = 8.000 \text{ m} \quad \Delta t = 600 \text{ sec} \quad ne = 21 \]

\[ \Delta x = 4.000 \text{ m} \quad \Delta t = 300 \text{ sec} \quad ne = 41 \]

Plotting for every hour. Resistance neglected.

fig. 6.4 A sinus shaped wave in a rectangular channel.
\[ \Delta x = 1500 \text{m} \]
\[ \Delta t = 22.5 \text{ sec} \]
\[ n = \text{number of nodal points} = 41 \]

Initial values:
\[ Q(t=0) = 0 \text{ m}^3 / \text{sec} \]
\[ h(t=0) = 40 \text{ m} \]

Resistance neglected

Boundary condition:
\[ h(40) = 10 + \sin 2t \text{ m} \]

\[ Q(0) = 0 \text{ m}^3 / \text{sec} \]
The diagram shows the relationship between time (t) and height (h) over a period of 40 hours, where:

- At t = 2 hours, h(m) = 11.
- At t = 2 1/4 hours, h(m) = 10.
- At t = 2 1/2 hours, h(m) = 10.
- At t = 2 3/4 hours, h(m) = 10.
- At t = 3 hours, h(m) = 10.
- At t = 3 1/4 hours, h(m) = 10.
- At t = 3 1/2 hours, h(m) = 10.
- At t = 3 3/4 hours, h(m) = 10.

The height decreases over time.

Wave coming in from the right-hand boundary with height $h(40) = 10 + \sin \frac{2 \cdot t}{T}$ m while the left hand boundary has been defined closed ($Q(0) = 0$ m$^3$/sec)

Plotting for every 4 timesteps $= 4 \Delta t = 900$ sec = $1/4$ hr.

Wave period $T = 6$ hrs

---

**fig. 6.4** A sinus shaped wave in a channel with a rectangular cross-section and horizontal bottom. Resistance neglected.
Case 3)
Computation of a sinus wave as plotted in fig. 6.5.
The conditions are given in the same figure.

6.12 Comparing with a finite difference scheme, system 11.
A comparison of some results from the calculation, using the finite elements program and a finite difference program, has been made. However, a few serious remarks must be made:
The intention of this comparison is by no means to say something about the quality of the results. For then a lot more investigations must be done. The only intention of this chapter is to show that the representing of a certain case is on one hand the same but on the other hand it can differ a little. Once more must be said that no conclusions can be derived from these two cases.
See fig 6.5
The conditions were for both cases the same:
Initial conditions:
Length of the channel : 123.000 m
number of nodal points 42
number of elements 41

Boundary conditions:
\[ h_1(t) = h_{\text{init}} + \sin 2 \frac{t}{T}, \quad T = 3 \text{ hrs.} \]
\[ Q_{\text{ne}}(t) = 0 \text{ m}^3/\text{sec} \]
For every time step a calculation has been made for the values in every nodal point.
For every hour a plotting has been made for the values in only 11 (s11) or 12 (fem) nodal points.
Plot for time = 1 hr. No resistance.

FEM: \( \Delta t = 450 \text{ sec} \)

Plot for time = 2 hr. No resistance

Plot for time = 3 hrs. No resistance

fig. 6.5 -a, b, c Comparing with a finite difference scheme
fig. 6.5 - c, d, e

ne=42  Δx= 3000 m  S11:  Δt=600 sec  +  Plot for time= 4 hrs.  No resistance

FEM: Δt=450 sec Δ

Plot for time= 5 hrs.  No resistance

Plot for time= 6 hrs.  No resistance.
ne = 42  \Delta x = 3000 \text{ m}  \quad S11: \Delta t = 600 \text{ sec}  \\
FEM: \Delta t = 450 \text{ sec}  \\
Plot for time = 1 \text{ hr.}  \quad \text{Resistance included}

Plot for time = 2 \text{ hrs.}  \quad \text{Resistance included}

Plot for time = 3 \text{ hrs.}  \quad \text{Resistance included}
ne = 42  $\Delta x = 3000$ m  S11: $\Delta t = 600$ sec  
Plot for time = 4 hrs. Resistance included.
FEM $\Delta t = 450$ sec

Plot for time = 5 hrs. Resistance included.

Plot for time = 6 hrs. Resistance included.

fig 6.5  g, h, i  Comparing with a finite difference scheme
CHAPTER 7

FURTHER POSSIBLE INVESTIGATION

This report does not have, as already mentioned, the intention to discuss realistic problems, but rather to give a review of the various kinds of finite element methods available for solving partial differential equations.

When using one finite element method (the method of Galerkin) the hyperbolic one-dimensional long period shallow water equations are solved using linear interpolation functions.

Of course it is also possible to use quadratic interpolation functions (see Appendix B) and other weighted residual methods (see Chapter 3), and even to extend the case, discussed in Chapter 6, to a system of channels.

fig 7.1 An interconnected system of waterways

Another possibility is to take into account the two-dimensional instead of the one-dimensional case. Then the long period shallow water wave equations (including windstress, bottom elevation, Coriolis effect and surface pressure) become:
Mass equation
\[ \frac{\partial h}{\partial t} + \frac{\partial p}{\partial x} + \frac{\partial q}{\partial y} = 0 \]

Momentum equation in X-direction
\[ \frac{\partial p}{\partial t} + \frac{\partial}{\partial x} \left( \frac{p^2}{h} \right) + \frac{\partial}{\partial y} \left( \frac{pq}{h} \right) + q \cdot \frac{h}{\partial x} \left( l + H \right) + \frac{qNp^2+q^2}{c^2h^2} \cdot p - \kappa q - W_x + q \cdot \frac{\partial p}{\partial x} = 0 \]

Momentum equation in Y-direction
\[ \frac{\partial q}{\partial t} + \frac{\partial}{\partial x} \left( \frac{q^2}{h} \right) + \frac{\partial}{\partial y} \left( \frac{pq}{h} \right) + q \cdot \frac{h}{\partial y} \left( l + H \right) + \frac{qNp^2+q^2}{c^2h^2} \cdot q + \kappa q - W_y + q \cdot \frac{\partial p}{\partial x} = 0 \]

where

- \( h \) = water level
- \( p \) = flux density in \( x \)-direction
- \( q \) = flux density in \( y \)-direction
- \( g \) = gravitational acceleration
- \( H \) = bottom elevation
- \( C \) = De Chezy resistance coefficient
- \( \kappa \) = Coriolis' coefficient, latitude dependant
- \( W_x \) = wind stress in \( x \)-direction
- \( W_y \) = wind stress in \( y \)-direction
- \( P \) = atmospheric pressure, water column
A choice has to be made concerning the type of element to be used. For example, when a triangle shaped element is chosen, the weighting function defined for nodal point \( k \), which is surrounded by the elements \( i-2, i-1, i, i+1 \) and \( i+2 \) can be written as (see also fig 7.2):

\[
\phi_k = \phi_{k,i-2} + \phi_{k,i-1} + \phi_{k,i} + \phi_{k,i+1} + \phi_{k,i+2} = \sum_{i=-2}^{2} \phi_{k,i} = \sum_{i=-2}^{2} \frac{A_{k,i}}{A_k}
\]

Where

- \( \phi_k \) = weighting function, defined in nodal point \( k \)
- \( i \) = subscript, indicating the relation to element \( i \)
- \( A \) = area or sub-area of the concerned element.

For the definition of \( \phi_{k,i-2}, \phi_{k,i-1}, \phi_{k,i}, \phi_{k,i+1}, \phi_{k,i+2} \), see Appendix B.

fig. 7.2 An example of a weighting function in a two dimensional case.
In this way, it is possible to construct for each basic equation as many equations as the number of nodal points. The principle remains the same as what is done in this report, only that the building up of the set of equations and the organization of the program is much more complicated. The weighting functions can be of a linear-type or quadratic-type etc. When other types of elements are used, they will give other weighting functions which can, again, be of a linear or quadratic type.
APPENDIX A

A.1 Defining the Validity of Equations

If the set of equations which describes a propagation problem is known, then the area in time and space, in which these equations are valid, must be defined. In general, a propagation problem will be described by three kinds of equations, while the area in space is subdivided into two parts:

- **domain** $D = \text{area in space except the boundaries}$
- **boundaries** $S = \text{borders of the system}$

while the domain and the boundaries together are called the **region**.

The three kinds of equations describing the problem are:

a.) Governing equations, defined within the domain

$$[L] \{a\} = \{f\} \quad \text{in } D \forall t > 0 \quad (A.1)$$

b.) Boundary data, also called boundary conditions, which describe what is happening on the boundaries themselves

$$[B] \{a\} = \{g\} \quad \text{on } S \forall t > 0 \quad (A.2)$$

c.) Initial data, which gives the starting values of time level zero

$$[I] \{a\} = \{g\} \quad \text{in } D \text{ on } S \forall t = 0 \quad (A.3)$$
where:

\[ [L], [B] \text{ and } [I] \] are matrices or differential operations

\[ \{a\} \] is a vector containing the dependent variables

\[ \{f\}, \{g\} \text{ and } \{h\} \] are vectors containing 'constant' terms

Usually the initial conditions are given as values, while the boundary conditions are, most of the time, either functions describing the dependent variables in time or values on fixed time levels. Although the initial conditions, as well as the boundary conditions are defined here as being equations (because of their completeness), they are normally given as values or functions.

### A.2 Interpolation Functions

An interpolation function is a function which enables us to describe a value over a certain region instead of only in a number of nodal points. A simple example of linear interpolation functions are the one dimensional linear Lagrange interpolation functions:

\[ \phi_1 = 1 - \frac{x}{\Delta x} \]

\[ \phi_2 = \frac{x}{\Delta x} \]

Using the interpolation functions \( \phi_1 \) and \( \phi_2 \) within the region the value of \( u \) is approximated by interpolation, only depending upon \( x \) :

\[ \hat{u}^c = \phi_1 \cdot u_1 + \phi_2 \cdot u_2 = \sum_{k=1}^{n} \phi_k \cdot u_k \]  \[ (A. 4) \]
Of course there exist many alternative interpolation functions, also for the two and three dimensional cases. Some of them are discussed in Appendix B.

A. 3 Trial Functions or Trial Solutions

A trial solution is a kind of approximation of the exact solution. When the exact solution will be defined over the whole region, then the trial solution must also be defined over the whole region. A trial solution can, in general, be written as

\[ \hat{\text{a}} = \sum_{r=1}^{\infty} \phi_r \cdot C_r \]  

(A. 5)

where

- \( \hat{\text{a}} \) is the trial solution, defined over the region
- \( \phi_r \) are the independent selected functions existing over the region (also on the boundaries)
- \( C_r \) are unknown parameters, related to the unknown dependent variables which are to be determined

The trial solution is usually not defined as vaguely as it is carried out above. Thus it can be said that:

- \( \hat{\text{a}} \) is a vector representing the trial solution over the region
- \( \phi_r \) are the interpolation functions which have just been discussed in A.2 and will be discussed further in Appendix B
- \( C_r \) are the unknown dependent variables in arbitrarily chosen points over the region and is written as \( \hat{a}_r = \hat{\alpha}_r \)

The symbol \( \hat{\cdot} \) on the dependent variable means that it is an approximated value and not the exact solution.

The condition for a trial solution, in general, is that it has to (or is supposed to) satisfy the initial conditions as well as the boundary conditions. It is not necessary that it satisfies the domain conditions. On the other
hand it is also possible (but not usual) that the trial solution satisfies the domain conditions as well as the boundary conditions, but not necessarily the initial conditions.

By introducing elements (Appendix B) with their interpolation function and considering only one element, the trial solution can be written as:

\[
\hat{\phi}_i = \sum_{k=1}^{ke} \phi_{k,i} \hat{\alpha}_k
\]

(A.6)

where

- \( \hat{\phi}_i \) = trial solution only defined in element \( i \)
- \( ke \) = number of nodal points of the element
- \( k \) = subscript indicating the related nodal point
- \( i \) = subscript indicating the related element
- \( \hat{\alpha}_k \) = approximated value of the dependent variable

As can be seen from Appendix B, \( \phi_{k,i} \) is only a function of the space coordinates \( x, y \), and \( z \) while, because \( \alpha_k \) is defined and so, also fixed in only the nodal points, \( \hat{\alpha}_k \) is only a function of the time. For only one element \( i \) the trial function can be written as:

\[
\hat{\phi}_i(x,y,z,t) = \sum_{k=1}^{ke} \phi_{k,i}(x,y,z) \cdot \hat{\alpha}_k(t)
\]

(A.7)

A.4 Interpolation functions and boundary conditions

In general, there are three types of boundary conditions:

- Boundary conditions where the dependent variables are given without derivatives (Dirichlet-type)
- Boundary conditions where the dependent variables are given as derivatives (Neumann-type)
- Boundary conditions where derivatives as well as non-derivatives of the dependent variables are given (mixed type boundary conditions)
In literature of finite element methods two main type of interpolation functions are mentioned:

- Lagrange interpolation functions
- Hermitian interpolation functions.

Because the last type of interpolation functions are not of any importance for the problems, discussed in this report, they are only mentioned. Also the boundary conditions which are discussed in this report are of the Dirichlet-type.

A.5 Example of a Trial Solution for the One Dimensional Case Using Langrange Shape Functions.

Because of their simplicity, linear Langrange shape functions are almost always used in practice. Also, the number of nodal points within an element is kept as small as possible. The simplest shape function for the one dimensional case is a linear interpolation function, defined in element i.

If \( \hat{a}_k \) and \( \hat{a}_{k+1} \) are the unknown approximated values of the dependent
variables in the nodal points \( k \) and \( k + 1 \), then:

\[
\begin{align*}
\hat{a}_k &= \phi_{k,i} \cdot a_k + \phi_{k+1,i} \cdot \hat{a}_{k+1} = \sum_k \phi_{k,i} \cdot \hat{a}_k \\
\hat{a}_i &= \begin{bmatrix} \phi_{k,i} & \phi_{k+1,i} \end{bmatrix} \begin{bmatrix} \hat{a}_k \\
\hat{a}_{k+1} \end{bmatrix}
\end{align*}
\]

(A. 8)

So for each element an approximated \( \hat{a}_i \) can be derived from the unknown approximated nodal values and the shape functions defined over the element. Note that the interpolation functions are only dependent on the shape and the size of the elements. The trial function over the whole region (\( NF \) elements) can be written as:

\[
\hat{a} = \sum_{L=1}^{NF} \left( \sum_{k=1}^{ke} \phi_{h,i} \cdot \hat{a}_k \right)
\]

(A. 9)

where:

\( \hat{a} \) = trial function, defined in every point of the region

\( NF \) = number of elements within the region

\( ke \) = number of nodes within the element \( i \)

\( \phi_{h,i} \) = interpolation function with unit value in node within element \( i \)

\( \hat{a}_k \) = approximated value of \( a \) in nodal point \( k \)

This is made visual for a one dimensional case with \( NE \) nodal points and so \( NF \) nodal points and \( NF = NE - 1 \) elements, using linear Langrangian interpolation functions. See fig. A. 4.
Interpolation functions are not always used in this way. For example, in the one dimensional case, 'hat functions' can be introduced. The interpolation remains the same, only the interpolation functions are written in a somewhat different way.
From the equation:

\[ \hat{a} = \sum_{k=1}^{n_e} \phi_k \cdot \hat{a}_k \]

follows:

\[ (\phi_{i_1} \cdot \hat{a}_1 + \phi_{i_2} \cdot \hat{a}_2 + \phi_{i_3} \cdot \hat{a}_3 + \ldots + \phi_{i_{k-1}} \cdot \hat{a}_{k-1} + \phi_{i_k} \cdot \hat{a}_k + \ldots + \phi_{n_e} \cdot \hat{a}_{n_e}) = \sum_{k=1}^{n_e} (\phi_k \cdot \hat{a}_k). \quad \text{(A.10)} \]

fig A.5 Pyramid- or hat function interpolation for the one dimensional case.
These kind of interpolation functions are often used in linear one dimensional problems.

A. 6 Curve Fitting

Curve fitting is a well known mathematical principle that tries to describe an arbitrary curve. This is done, for example, in the harmonic methods where a curve is followed, using harmonic functions. In some finite element methods, curve fitting is also used.

In describing a curve there is always a difference between the describing curve and the curve which must be fitted. The absolute value of this difference must be bent to zero.
APPENDIX B

ELEMENTS AND THEIR INTERPOLATION FUNCTIONS

B. 1 Kinds of Elements

In finite element methods, elements are subregions of the total region considered, which can be subdivided in two ways:

- by the dimension
  - one-, two-, or three-dimensional elements
- by the use of linear or curved-linear (iso-parametric) elements, depending on whether the boundaries of the elements are straight or curved. A curved linear element for the one-dimensional case can be two- or three-dimensional (see fig. B.1 to B.6).

In the following, only curved-linear elements are mentioned. For a further outline of curved-linear elements, refer back to the literature [12],[16],[17].

The great advantage of curved-linear elements is that they easily fulfill almost every shape of region. The disadvantage of it is, however, that the description of element shape and the interpolation functions is rather complex. Another disadvantage is that non-uniqueness is possible because of violent distortion in the relationship between the cartesian and the new coordinates (which describe the curved-linear elements) ((see fig. B19 and B20))

For every element a local set of coordinates is introduced. For straight linear elements local cartesian coordinates are used. For curved-linear elements, however, another type of coordinates must be introduced, depending upon the shape of the element. Now there must be an interrelation between the set of coordinates, defined over the whole region, and the local set of coordinates in each element. The set of coordinates for the region is not necessarily cartesian, but the local set of coordinates must be.

B. 2 The One-dimensional Element

The one-dimensional element with linear interpolation functions needs only two nodal points.
fig B.1 Linear one dimensional element

fig B.2 Curved linear one dimensional element

fig B.3 Linear two dimensional elements

fig B.4 Curved linear two dimensional elements

fig B.5 Linear three dimensional elements

fig B.6 Curved linear three dimensional elements
Any value of the dependent variable $a$ can be derived from the interpolation between the nodal values $a_k$ and $a_{k+1}$.

$$\phi_{k,i} = \frac{\Delta - x}{\Delta}$$  \hspace{1cm} (b. 1)

$$\phi_{k+1,i} = \frac{x}{\Delta}$$  \hspace{1cm} (b. 2)

$$a^e_i = \phi_{k,i} \cdot a_k + \phi_{k+1,i} \cdot a_{k+1} = \sum_k \phi_{k,i} \cdot a_k$$  \hspace{1cm} (b. 3)

where:

- $i =$ subscript for the element
- $k =$ subscript for the nodal point
- $e =$ superscript indicating that the expression is only valid within the considered element

The one-dimensional element with quadratic interpolation functions needs three nodal points (an extra internal nodal point).

$$\phi_{k-1,i} = \left(\frac{\Delta - 2x}{\Delta}\right) \left(\frac{\Delta - x}{\Delta}\right)$$  \hspace{1cm} (b. 4)

$$\phi_{k,i} = \frac{4x}{\Delta} \left(\frac{\Delta - x}{\Delta}\right)$$  \hspace{1cm} (b. 5)

$$\phi_{k+1,i} = \frac{x}{\Delta} \left(\frac{2x - x}{\Delta}\right)$$  \hspace{1cm} (b. 6)
In the same way, third order interpolation functions can be created, but they are seldom used because of their complexity.

B 3 The Two-dimensional Element

The most well known two-dimensional elements are:

a.) the rectangular

b.) the triangle

a. 1) The rectangular element with linear interpolation functions

First a linear interpolation in x direction on y-level j and j+1 is used.

\[ a_{x,i}^{j} = \phi_{k,i} \cdot a_{k,i} + \phi_{k+1,i} \cdot a_{k+1,i} \]  \hspace{1cm} (b. 8)

Next the interpolation is introduced in the y-direction with the x-interpolated \( a_{x,i}^{j} \) and \( a_{x,i}^{j+1} \) as nodal values.

The interpolation over the x-y plane now becomes:

\[ a_{k}^{j} = \phi_{k-1,l} \cdot a_{k-1,l} + \phi_{k,l} \cdot a_{k,l} + \phi_{k+1,l} \cdot a_{k+1,l} = \sum_{k-1}^{k+1} \phi_{k,l} \cdot a_{k}. \]  \hspace{1cm} (b. 7)
\[ a_i^e = a_x, y, i = \phi_j, i \cdot a_x, j, i + \phi_{j+1}, i \cdot a_x, j+1, i \]  

(b. 10)

or

\[ a_x, y, i = \phi_k, i(x) \cdot \phi_{j, i}(y) \cdot a_{k, j} + \phi_{k+1, i}(x) \cdot \phi_{j, i}(y) \cdot a_{k+1, j} + \phi_k, i(x) \cdot \phi_{j+1, i}(y) \cdot a_{k, j+1} + \phi_{k+1, i}(x) \cdot \phi_{j+1, i}(y) \cdot a_{k+1, j+1} \]  

(b. 11)

or

\[ a_{x, y, i} = \sum_j \sum_k \phi_k, i \cdot \phi_{j, i} \cdot a_{k, j} \]  

(b. 12)

where

\[ \phi_k, i = \frac{\Delta x - x}{\Delta x} \]  

(b. 13)

\[ \phi_{k+1, i} = \frac{x}{\Delta x} \]  

(b. 14)

\[ \phi_{j, i} = \frac{\Delta y - y}{\Delta y} \]  

(b. 15)

\[ \phi_{j+1, i} = \frac{y}{\Delta y} \]  

(b. 16)

Note that the \( x \) and \( y \) coordinates are dependent of the place of the element.

a. 2) The rectangular with quadratic interpolation functions

In the case of \( a_1 \), we needed for interpolation four nodal points in each element. Here we need nine nodal points \((3 \times 3)\) so we have to introduce an internal nodal point \((k, j)\).

In principle the interpolation is the same as before:
\[
\alpha_i^e = \sum_{k=1}^{K_c} \sum_{k=1}^{K_c} \phi_{k,i} \cdot \phi_{k,i} \cdot a_{k,i} = (b. 17)
\]

\[
= \phi_{k-1,i} \cdot \phi_{k-1,i} \cdot a_{k-1,i-1} + \phi_{k,i} \cdot \phi_{k-1,i} \cdot a_{k,i-1} + \phi_{k+1,i} \cdot \phi_{k-1,i} \cdot a_{k+1,i-1} + \phi_{k,i} \cdot \phi_{k,i} \cdot a_{k+1,i} + \phi_{k,i} \cdot \phi_{k-1,i} \cdot a_{k,i-1} + \phi_{k+1,i} \cdot \phi_{k-1,i} \cdot a_{k+1,i} + \phi_{k,i} \cdot \phi_{k,i} \cdot a_{k+1,i} + \phi_{k,i} \cdot \phi_{k-1,i} \cdot a_{k,i-1} + \phi_{k+1,i} \cdot \phi_{k-1,i} \cdot a_{k+1,i} + \phi_{k,i} \cdot \phi_{k,i} \cdot a_{k+1,i} + \phi_{k,i} \cdot \phi_{k-1,i} \cdot a_{k,i-1} + \phi_{k+1,i} \cdot \phi_{k-1,i} \cdot a_{k+1,i} + \phi_{k,i} \cdot \phi_{k,i} \cdot a_{k+1,i}.
\]

\[
\phi_{k-1,i} = \frac{(\Delta - x)(\Delta - x)}{\Delta^2} \quad (b. 19)
\]
\[
\phi_{k} = \frac{4x(\Delta - x)}{\Delta^2} \quad (b. 20)
\]
\[
\phi_{k+1} = \frac{x(2x - \Delta)}{\Delta^2} \quad , \Delta = \Delta x
\]
\[
\phi_{k-1} = \frac{(\Delta - 2y)(\Delta - y)}{\Delta^2} \quad (b. 21)
\]
\[
\phi_{k} = \frac{4y(\Delta - y)}{\Delta^2}
\]
\[
\phi_{k+1} = \frac{y(2y - \Delta)}{\Delta^2} \quad , \Delta = \Delta y
\]

Although higher order interpolation functions are possible, they will be very complex and hardly useable.

b. ) The triangle

Instead of line coordinates, here, area coordinates are introduced. In an arbitrary point \((x, y)\) within the triangle, the interpolated value of \(a_i^e\) can be written as:

\[
a_i^e = \phi_{k-1,i} \cdot a_{k-1,i} + \phi_{k,i} \cdot a_{k,i} + \phi_{k+1,i} \cdot a_{k+1,i}
\]

\[
\phi_{k, i} \text{ is written now as a function of the area of the triangle:}
\]

\[
\phi_{k-1,i} = \frac{A_{k-1,i}}{A_i}, \quad \phi_{k,i} = \frac{A_{k,i}}{A_i}, \quad \phi_{k+1,i} = \frac{A_{k+1,i}}{A_i}
\]
with $A_{k,i} =$ area, enclosed by $(x, y), (x_{k+1}, y_{k+1})$ and $(x_{k-1}, y_{k-1})$.

$A_i =$ area of the triangle with the number $i$.

In practice this kind of notation can create problems. In such a case, it is usual to introduce "local coordinates" as was done with the discussion of rectangulars etc.

In this case you cannot consider the whole region at one time, but element by element. (see fig. B.12)

The relation between the local coordinates $(x, y)$ is given by:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_2 & x_3 \\ 1 & y_1 & y_2 & y_3 \end{bmatrix} \cdot \begin{bmatrix} \phi_{1,i} \\ \phi_{2,i} \\ \phi_{3,i} \end{bmatrix}$$ (b. 26)

$$2A = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}$$ (b. 27)

or

$$2A_{1,i} = \begin{bmatrix} 1 & x & y \\ 1 & x_1 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} ; 2A_{2,i} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x & y \\ 1 & x_3 & y_3 \end{bmatrix} ; 2A_{3,i} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x & y \end{bmatrix}$$ (b. 28)

$$\phi_{1,i} = \frac{x_2\cdot y_3 - x_3\cdot y_2 + x_3\cdot y - x\cdot y_3 + x\cdot y_2 - x_2\cdot y_1}{x_2\cdot y_3 - x_3\cdot y_2 + x_3\cdot y_1 - x_1\cdot y_3 + x_1\cdot y_2 - x_2\cdot y_1}$$, cycl. (b. 29)

$$\phi_{1,i} = a_1 + \phi_{2,i} + a_2 + \phi_{3,i}$$ (b. 30)

The interpolated value of $a_1^e$ within the triangle is written as:

$$a_1^e = \phi_{1,i} \cdot a_1 + \phi_{2,i} \cdot a_2 + \phi_{3,i} \cdot a_3$$ (b. 31)
Note that $\phi_{1,i}$, $\phi_{2,i}$, and $\phi_{3,i}$ are dependent on each other:

$$\phi_{1,i} + \phi_{2,i} + \phi_{3,i} = 1$$

so what happens in fact is that the two independent variables $x$ and $y$ are transformed into two other independent variables $\phi_1$ and $\phi_2$ or $\phi_2$ and $\phi_3$ or $\phi_3$ and $\phi_4$.

Note

In finite element methods, integrals are used. By using rectangulars (line elements), these integrals are rather easy to solve, but by using triangles, this is more complicated. Therefore, here the general shape and solution of these integrals is given:

$$\int_A \phi_1 \cdot \phi_2 \cdot \phi_3 \cdot dA = \frac{p! \cdot q! \cdot r!}{(p+q+r+2)!} \cdot 2A$$  \hspace{1cm} (b. 33)

in which $p$, $q$ and $r$ are exponents.

b.) Triangle with internal nodes

Using quadratic interpolation functions for a higher accuracy, triangles with internal nodes are introduced. (see fig. B.13)

The interpolated dependent variable can be written as:

$$a_i^e = \sum_{k=1}^{6} \phi_{k,i} \cdot a_k$$  \hspace{1cm} (b. 34)

$$\phi_{1,i} = A_1 \cdot (2A_1 - 1)$$  \hspace{1cm} (b. 35)

$$\phi_{2,i} = A_2 \cdot (2A_2 - 1)$$  \hspace{1cm} (b. 36)

$$\phi_{3,i} = A_3 \cdot (2A_3 - 1)$$  \hspace{1cm} (b. 37)

$$\phi_{4,i} = 4 \cdot A_1 \cdot A_2$$  \hspace{1cm} (b. 38)

$$\phi_{5,i} = 4 \cdot A_2 \cdot A_3$$  \hspace{1cm} (b. 39)

$$\phi_{6,i} = 4 \cdot A_1 \cdot A_3$$  \hspace{1cm} (b. 40)
Of course, for a higher interpolation accuracy, much more complicated triangles can be used, but then the interpolation functions become too complicated. In such a case, it is better to use more elements.

c.) Other types of two-dimensional element shapes

The two most used types of elements are the rectangular and the triangle. More types are possible, however, they are not used here. Only the quadrilateral is mentioned.

Of course it is possible to construct interpolation functions for it. It is easier, however, to create two triangles from it, which have less complex functions.

![The quadrilateral divided into triangles.](image)

**Fig A.14** The quadrilateral divided into triangles.

---

**B 4 The Three Dimensional Element**

Only the two basic elements are mentioned here, because all non-curved linear elements can be built up from these two:

a.) the rectangular prism

b.) the tetrahedron

a.1) The rectangular prism with linear interpolation functions

The local origin is in the centre of the prism. The interpolated value of \( a_i^e \) in an arbitrary point within the prism:

\[
a_i^e = \sum_{k=1}^{8} \phi_{k,i} \cdot a_k
\]

where

\[
\phi_{k,i} = \frac{1}{8} \cdot (1 + \frac{X}{\Delta X})(1 + \frac{Y}{\Delta Y})(1 + \frac{Z}{\Delta Z})
\]

(\( k=1, 2, \ldots, 8 \))

![Rectangular prism using linear interpolation functions](image)
a. 2) The rectangular prism with quadratic interpolation functions

Because of their complexity, the interpolation functions are not mentioned. The trial function within the element can be written as

\[ a_i^e = \sum_{k=1}^{26} \phi_{k,i} \cdot a_k \] (b.43)

For practical use, this type of element with middle nodes is too complex. It may be better in such cases to use more linear elements.

b.) The tetrahedron

According to what is done with the triangle, we can write

\[ a_i^e = \sum_{k=1}^{4} \phi_{k,i} \cdot a_k \] (b.44)

\( \phi_{k,i} \) is written now as a function of the volume:

\[ \phi_{k,i} = \frac{V_{k,i}}{V_i} \quad (k = 1, 2, 3, 4) \] (b.45)

\( V_k \) = volume of the part of the tetrahedron, formed by the connection of \((x,y)\), \((x_2,y_2)\), \((x_3,y_3)\) and \((x_4,y_4)\), cycl.

The relation between the local coordinates \((1, 2, 3 \text{ and } 4)\) and the cartesian coordinates \((x,y)\) is given by

\[
\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix} =
\begin{bmatrix}
  x_1 & x_2 & x_3 & x_4 \\
  y_1 & y_2 & y_3 & y_4 \\
  z_1 & z_2 & z_3 & z_4 \\
  1 & 1 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
  \phi_1 \\
  \phi_2 \\
  \phi_3 \\
  \phi_4
\end{bmatrix}
\] (b.45)

The volume of a tetrahedron can be calculated from:
For the quadratic tetrahedron (with middle nodes) we get:

\[ \phi_{k,i} = \left( 2 \frac{V_{k,i}}{V_i} - 1 \right) \frac{V_{k,i}}{V_i} \quad (k=1, 2, 3, 4) \]  
\[ \phi_{5,i} = 4 \frac{V_{1,i}}{V_i} \cdot \frac{V_{2,i}}{V_i} \quad \text{etc.} \]  

At last the general shape and solution of the integral will be given in the same way as done for the triangle:
\[
\frac{1}{V^4} \cdot \int \left( V_1^{p} \cdot V_2^{q} \cdot V_3^{r} \cdot V_4^{s} \right) dx \cdot dy \cdot dz = \\
\frac{p! \cdot q! \cdot r! \cdot s!}{(p+q+r+s+3)} \cdot 6V
\]  
(b. 51)

c.) A third possible shape could be the combination of triangles and rectangulars = the triangle prism element. Because this one can be simply built up from simple langrange functions (rectangle) and simple linear triangle functions, it is only mentioned.
# REFERENCES

<table>
<thead>
<tr>
<th>Reference</th>
<th>Author(s)</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>(8)</td>
<td>Grootenboer, H. J. and J. L. Reijnen</td>
<td>Toepassing van de elementenmethode op tweedimensionale isotrope warmtediffusie. Rijkswaterstaat, Nota D. I. V. 7201</td>
</tr>
<tr>
<td>(10)</td>
<td>Praagman, N. and A. Segal</td>
<td>Application of the finite element method to initial value problems. Technische Hogeschool Delft, Report NA-10, 1974</td>
</tr>
<tr>
<td>(15)</td>
<td>Segal, A.</td>
<td>Kollegedictaat numerieke analyse B III. Technische Hogeschool Delft</td>
</tr>
</tbody>
</table>