An inquiry into the limits of applicability of the refraction - diffraction equation .

Thesis in partial
fulfillment of the
requirements of the
ingenieurs examination.

D.O. van Nieuwenhuijze

Delft aug 1981

Contents

SAMENVATTING

In deze studie zijn twee modellen met elkaar vergeleken welke elk de waarden voor een golfpotentiaal voorspellen (numeriek).
Het eerste model, " Side View Model ", is gebaseerd op de vergelijking van Laplace en beschrijft in een verticaal vlak. Het tweede model, " Top View Model ", geeft een beschrijving in een horizontaal vlak (het oppervlak) en is gebaseerd op de Refractie-Diffractie vergelijking, zoals gepresenteerd door Berkhoff (1976).

Doel van het onderzoek was: het komen tot een inzicht ten aanzien van de gevoeligheid voor de verandering van de bodenhelling, in het Top View Model.
Voor de studie is een computerprogramma gemaakt dat, voor elk der twee modellen, de waarde van de golfpotentiaal aan het oppervlak berekent, voor een gekozen bodemconfiguratie. Het computerprogramma is gebaseerd op de eindige elementen methode en gebruikt het in Delft ontwikkelde pakket "AFEP" .
Als bodemconfiguratie is gekozen: een trapeziumvormige dam op een vlakke bodem; de hoogte van de dam, de helling en de lengte van het talud van de dam zijn gevarieerd en de voorspelde waarden zijn voor beide modellen vergeleken voor de identieke gevallen.

Het is gebleken dat er geen duidelijke directe correlatie is tussen de vergeleken waarden (van de beide modellen) en een verandering van de bodemhelling bij de gebruikte toetsingsparameter (welke afgeleid is van de door Berkhoff gepresenteerde toetsingsparameter). Uitgaande van de hier gepresenteerde resultaten zou verdergaand onderzoek tot meer inzicht kunnen leiden.

→

1 Introduction

Wave phenomena have been studied with great interest throughout the ages, due to the importance of knowledge of this matter for various human activities (such as commerce: shipping ; habitation: shore protection).

A number of models have been designed, " equations ", which predict the changes in the characteristics of water waves as a result of the variations of the water-depth over which the wave travels.

One of these models, the " refraction - diffraction equation ", has been published by Berkhoff (1976).
The equation describes a wavefield with a free surface condition over an impermeable rigid bottom, which is flat or 'nearly' flat.

The requirement of the bottom to be at most 'nearly' flat follows from the way the equation has been derived:
Using a known solution for the case of a flat and horizontal bottom (Lamb,1975), Berkhoff has assumed that this solution might also be used as an approximative description for a 'nearly' flat bottom, with 'rather gentle' (in quantitative sense) slopes.

This study has been undertaken in order to find out what is to be understood by " rather gentle " slopes.
That is, it is an inquiry into the limits of applicability of the refraction - diffraction equation.

In this study, the refraction - diffraction equation, as well as the set of equations from which it has been derived, are used to predict the values of the wave potential at the water surface. The computation is done numerically, using A Finite Element Package (" AFEP ") which has been developed at Delft University (Segal, 1975).

The predictions by the two models are compared for identical cases and a correlation between the slope of the bottom and the discrepancy in the predictions is sought.

Only simple linear waves are considered in this study.

This thesis is made up as follows:

Chapter 2 contains the presentation of the analytical core of the study: the set of equations which describe the wave field over a rigid impermeable bottom with a free surface condition, the refraction - diffraction equation which is derived from it and the relation between the two.

Two models are created based on these descriptions, respectively: " Side view model " and " Top view model " of which the predictions are to be compared for identical cases.

Chapter 3 presents the numerical form of these two models as they are to be used in AFEP.

Chapter 4 gives a description of the computer program.

The correctness of the predictions of the computermodel are verified, for the case of a flat and horizontal bottom, in chapter 5.

The experimentation, described in chapter 6, involves the selection of the configuration of the bottom and a discussion of the characteristic parameters of the problem. This leads to the selected combinations of parameters for the experimentation.

Chapter 7 presents the output of the computer program: the generated mesh; the form of presentation of the predicted values of the wave potential and the results.

The results are interpreted in chapter 8 and discussed in chapter 9, where the influence of all the parameters on the difference in the predictions of the two models is investigated.

These investigations show that no direct correspondence can be found between the variation of the characteristic parameters and the difference in the predictions of the two models. These conclusions are described in chapter 10.

Chapter 11 contains a number of recommendations for further studies which follow from this study.

Chapter 12 contains acknowledgements.

2 Analytical Description

2.00 Introduction

This chapter starts with a brief summary of potential formulation, using vector notation.

The concepts are then used to describe a potential flow and boundary conditions of the fluid field: a free surface condition and impermeable rigid bottom.

The equations are presented which describe the same field in linearised form, in a vertical plane, for a simple harmonic wave. (" Side view model ").

The analytical solution of these equations is shown as it has been derived for the case of a flat bottom.

It is shown, briefly, how this solution is used by Berkhoff (1976) to reduce the initial set of equations to the refraction - diffraction equation, describing the wave characteristic (the potential) at the water surface. (" Top view model ").

The Side View Model and the Top View Model are compared and a simple setup is presented which may be used to indicate the difference in predictions by the two models.

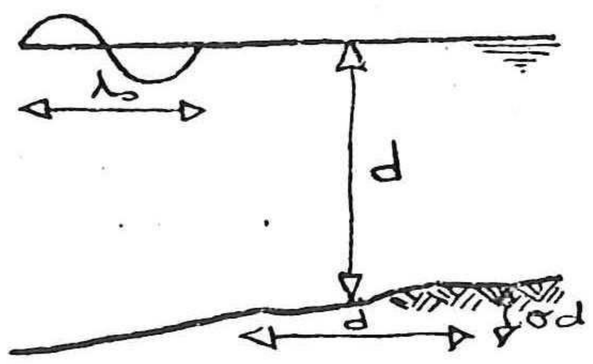
This study deals only with simple harmonic linear water waves. The fluid field is assumed to be free of currents; to be characterised by the Laplace equation.

Ignored are the effects of viscosity, compressibility, surface tension.

The wave potential φ is described in the complex plane:

$\varphi = \varphi_1 + i\,\varphi_2$ (i : imaginary unit)

Dimensionless parameters are used for the description.



d : mean water depth

T : wave period

$\lambda_0 = g \div \omega^2$

σ : mean bottom slope

2.01 Concepts of vector analysis

The description of the waves will be presented using the wave potential as the characterising parameter.
For this reason the neccessary concepts from vector analysis are presented, leading to the description of the potential.

- a vectorfield may be defined by $r_3 : (x,y,z)$
- a gradient is the vector, at location (x,y,z) which shows the direction of largest increment of a scalar-field "Φ"; the length of the vector indicates the size of this largest increment.

$$\text{grad}\,\phi = \nabla\phi = \left\{ \frac{\partial\phi}{\partial x} , \frac{\partial\phi}{\partial y} , \frac{\partial\phi}{\partial z} \right\} \qquad (2.01.1)$$

- the divergence of a vectorfield is a scalar which, at a location (x,y,z), indicates the strength of the source (or: sink) at that point of the vectorfield.

$$\text{div}\,\phi = \nabla\cdot\vec{v} = \left(\frac{\partial v_1}{\partial x} + \frac{\partial v_2}{\partial y} + \frac{\partial v_3}{\partial z} \right) = \lim_{vol\to 0} \frac{1}{vol} \oint \vec{v}\, dn \qquad (2.01.2)$$

- the rotation of a vectorfield is a vector which, at a location (x,y,z), indicates the intensity and direction of the change of orientation of the vectorfield.

$$\text{rot}\,\phi = \nabla * \vec{v} = \lim_{area\to 0} \frac{1}{area} \oint \vec{v}\cdot ds = \left\{ \frac{\partial v_3}{\partial y} - \frac{\partial v_2}{\partial z} ; \frac{\partial v_1}{\partial z} - \frac{\partial v_3}{\partial x} ; \frac{\partial v_2}{\partial x} - \frac{\partial v_1}{\partial y} \right\} \qquad (2.01.3)$$

- for a field without sources or sinks:

$$\nabla \cdot \vec{v} = 0 \qquad (2.01.4)$$

- for a field without rotation:

$$\nabla * \vec{v} = 0 \qquad (2.01.5)$$

- " nabla " is the vector operator:

$$\nabla \doteq \left\{ \frac{\partial}{\partial x} , \frac{\partial}{\partial y} , \frac{\partial}{\partial z} \right\}$$

2.02 Potential flow

A fluid field may be characterised by its velocity vectorfield. More convenient than the description as a vectorfield is a description using a scalarfield, " Potential ", using equation (2.01.1):

$$\vec{v} =: \nabla \Phi$$

here $\vec{v}$ is: the velocity vectorfield.

For a velocity field which is expressed as the gradient of the potential the condition of a rotation free field, (2.01.5), will always be fulfilled:

$$\nabla * \vec{v} = \nabla * (\nabla \Phi) \doteq \left\{ \frac{\partial}{\partial y} \cdot \frac{\partial \Phi}{\partial z} - \frac{\partial}{\partial z} \cdot \frac{\partial \Phi}{\partial y} ; \frac{\partial}{\partial x} \cdot \frac{\partial \Phi}{\partial z} - \frac{\partial}{\partial z} \cdot \frac{\partial \Phi}{\partial x} ; \frac{\partial}{\partial x} \cdot \frac{\partial \Phi}{\partial y} - \frac{\partial}{\partial y} \cdot \frac{\partial \Phi}{\partial x} \right\} =$$

$$\equiv \vec{0}$$

For a velocity field which is expressed as the gradient of a potential and which is free of sources and sinks the condition will be:

$$\nabla \cdot \nabla \phi = 0$$

That is: a fluid field without sources or sinks is characterised by:

$$\nabla \cdot \nabla \phi = \nabla^2 \phi = 0$$

the " Laplace equation ".

This description is used for the fluid field of this study:

The velocity vectorfield is described using a " potential ":

$$\vec{v} = \nabla \Phi \qquad \text{(2.02.1)}$$

The fluid field is assumed to be free of sources and sinks :

$$\nabla^2 \phi = 0 \qquad \text{(2.02.2)}$$

This is the field condition.

The next section presents the boundary conditions.

2.03 Boundary conditions of the field

Any field is determined by field condition and the boundary conditions.

Disregarding (untill section 2.09) the 'side' boundaries, the boundary conditions are:

the - linearised - free surface condition (see appendix 1):

$$\frac{\partial \Phi}{\partial z} + \frac{1}{g} \frac{\partial^2 \Phi}{\partial t^2} = 0 \qquad \text{at} \quad z = 0 \qquad (2.03.1)$$

and the rigid impermeable bottom condition:

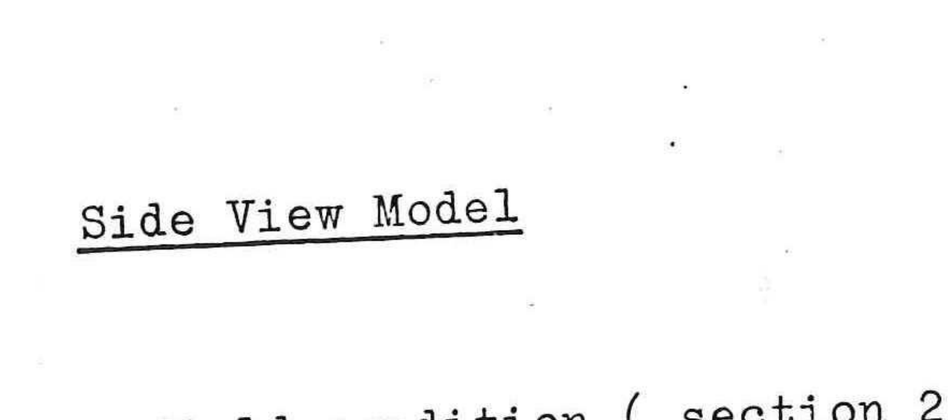
$$\frac{\partial \Phi}{\partial n} = 0 \qquad \text{at} \quad z = d \qquad (2.03.2)$$

d = waterdepth

2.04 Side View Model

Combining the field condition (section 2.02) with the boundary conditions of the field (section 2.03) and disregarding the variation in the " y-direction " a set of equations is obtained which will be called the " Side View Model " :

visualise,



(2.04.1)

where: $\nabla = \left\{ \frac{\partial}{\partial x}, \frac{\partial}{\partial z} \right\}$;

$\varphi = \varphi \langle x, z, t \rangle$

2.05 Flat bottom solution

An analytical solution is known for the Side View Model for the case of a flat horizontal bottom. ($\frac{\partial \varphi}{\partial z} = 0$).

In deriving this solution it has been assumed that the potential is time-space periodical.

Lamb (1975) shows that this solution is:

$$\varphi = \frac{g\,a}{\omega} \frac{\cosh (k (h+z))}{\cosh (kh)} \cdot e^{i(\omega t - kx)} \qquad (2.05.1)$$

where:
- g : gravitational gradient intensity
- a : amplitude
- ω : wave time frequency: " wave frequency "
- k : wave space frequency: " wave number "
- h : waterdepth
- z : depth below the water surface
- t : time
- i : unit of the imaginary number set

The displacement at the surface is:

$$\eta = i\,a \cdot e^{i(\omega t - kx)} \qquad (2.05.2)$$

There is a relation between the wave time frequency and the wave space frequency determined by the "elasticity" of the surface:

$$\omega^2 = g\,k \cdot \tanh (kh) \qquad (2.05.3)$$

The potential " φ " has three main components:

1) $\frac{g\,a}{\omega}$ an amplitude factor depending on ω (2.05.4)

2) $\frac{\cosh(k(h+z))}{\cosh(kh)}$ a distribution factor (over the vertical) (2.05.5)

3) $e^{i(\omega t - kx)}$ a phase factor (2.05.6)

This knowledge will be used in the formulation of the Top View Model.

2.06 Top View Model

In case the bottom variation is 'very small' (as assumed by Berkhoff) the solution for the case of a flat bottom (2.05.1) will be approximately valid:

$$\varphi\langle x,y,z,t\rangle = \varphi_o\langle x,y\rangle \cdot \frac{\cosh(k(h+z))}{\cosh(kh)} \cdot e^{i(\omega t - kx)} \qquad (2.06.1)$$

where: $\varphi_o\langle x,y\rangle$ is a first order approximation of $\varphi\langle x,y,z\rangle$ in case of a nearly flat bottom.

Berkhoff (1976) shows that the set of equations (2.04.1) then reduces to: (see appendix 3 for a short derivation):

$$\nabla \cdot (cc_g \nabla \varphi_o) + k^2 cc_g \varphi_o = 0 \qquad (2.06.2)$$

where: c : wave celerity

c_g: wave group celerity

φ_o : wave potential at the surface (z=0)

k : wave number

$\nabla = \left\{ \frac{\partial}{\partial x} , \frac{\partial}{\partial y} \right\}$ and,

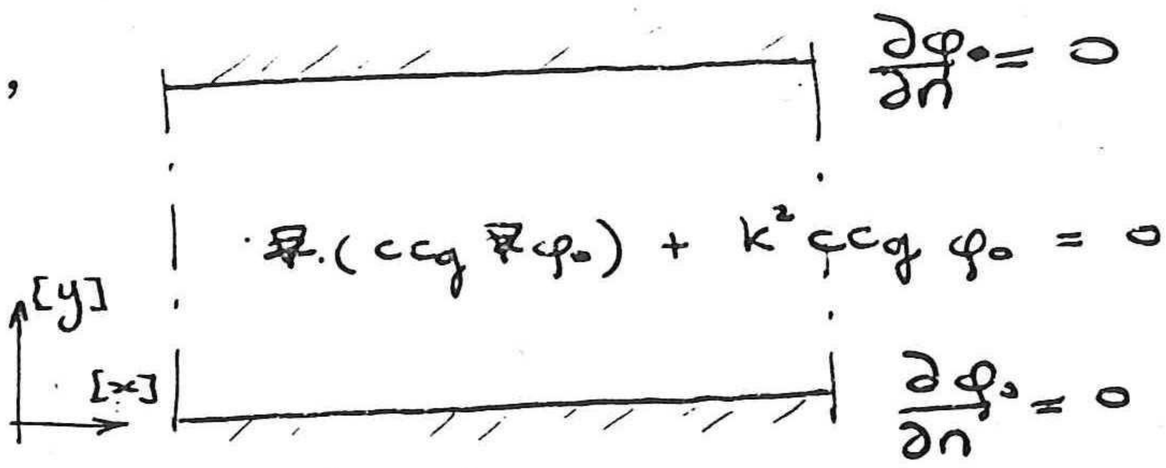
$$c \doteq \frac{\omega}{k} \qquad (2.06.3)$$

$$c_g = \tfrac{1}{2}\left(1 + 2\,\frac{kh}{\sinh(2kh)}\right) * c \qquad (2.06.4)$$

$$\omega^2 = gk \cdot \tanh(kh) \qquad (2.06.5)$$

Field equation (2.06.2), together with the boundary conditions of the field, will be called the " Top View Model " :

visualise,



(2.06.6)

where: $\nabla = \left[\frac{\partial}{\partial x} \cdot \frac{\partial}{\partial y}\right]$

2.07 Comparison

Two methods are now available for the prediction of the values of the potential throughout the wave field:

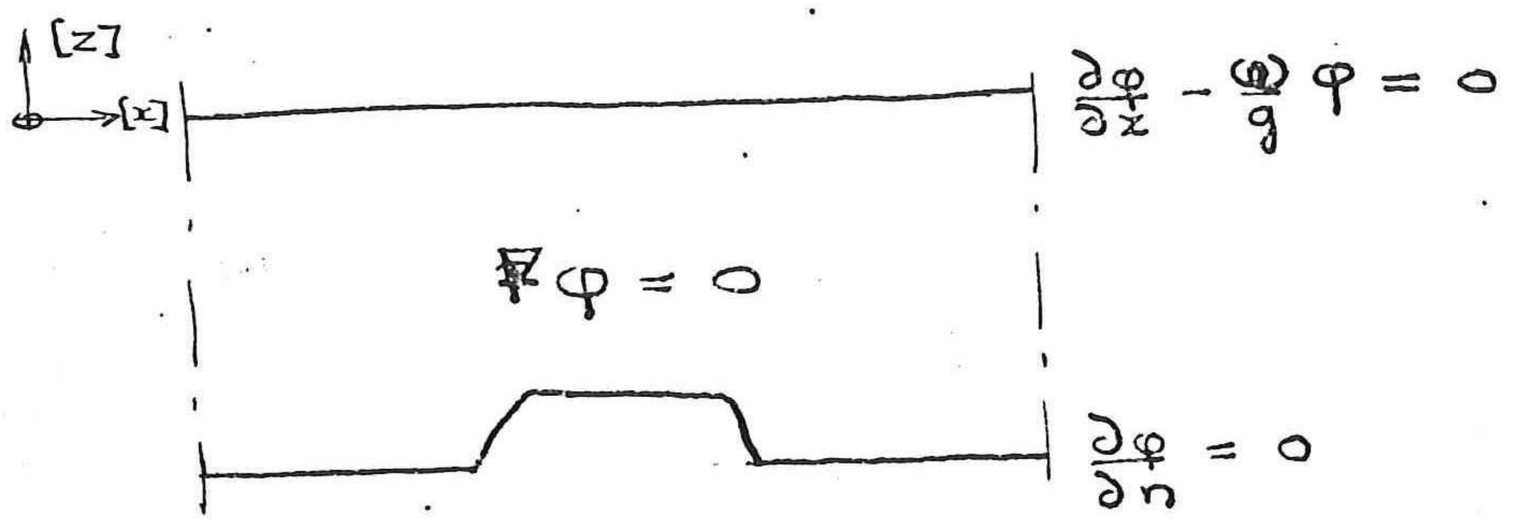
- (2.04.1) : the Side View Model, based on the Laplace equation.
- (2.06.6) : the Top View Model, based on the refraction - diffraction equation.

Since the time displacement factor, $e^{i(\omega t)}$, is known and the same for both models, this factor may be isolated and filtered out of the description:

The factor $e^{i(\omega t)}$ is further disregarded.

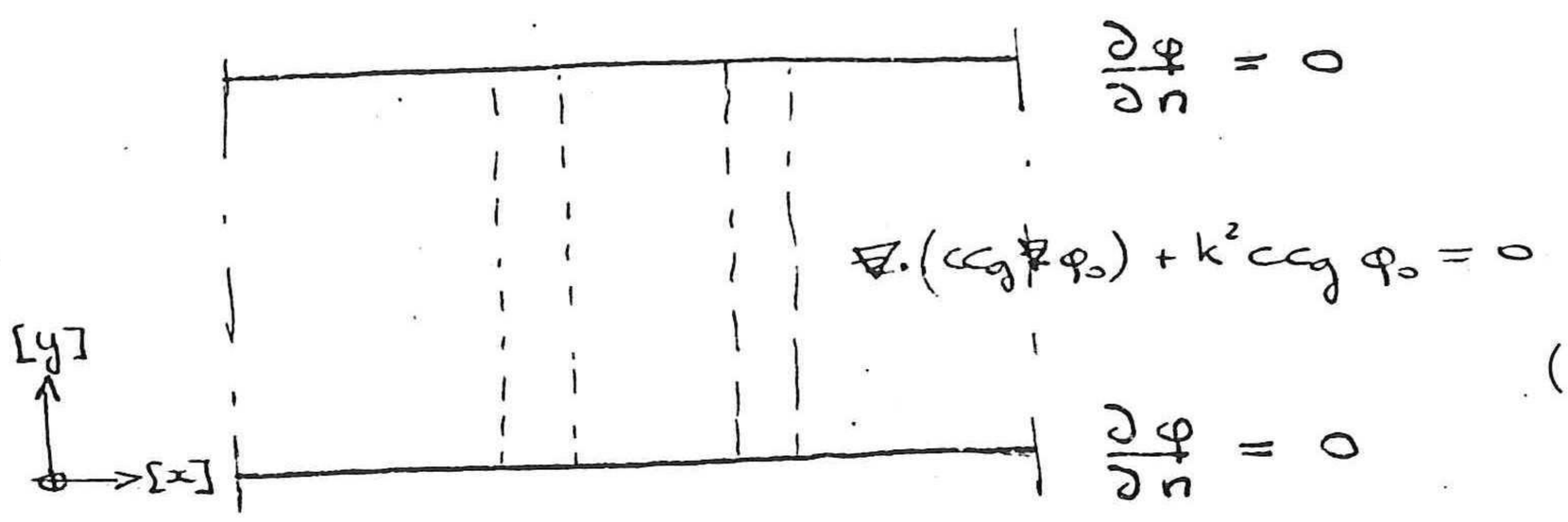
The two models which will be compared are:

Side View Model:



(2.07.1)

Top View Model:



(2.07.2)

These two models should both be able to predict the values of the wave potential in the field.

The Side View Model and the Top View Model differ in the following aspects:

- The Side View Model provides the values of the wave potential in a vertical plane whereas the Top View Model provides them for a horizontal plane: the water surface.
  This has the practical consequence that, for the comparison, only the common surface values have to be compared.

- The derivation of the refraction - diffraction equation (see appendix 3.) is based on the assumption that the variation of the bottom will be 'very small' . The Top View Model is based on this equation and thus may be sensitive to the bottom slope in its predictions; the Side View Model works for any bottom slope.
  This difference forms the basis of this study which is meant to provide insight in this particular characteristic of the Top View Model.
- The Top View Model is capable of dealing with variations in the plane of the horizontal which the Side View Model can not. (A 3-D model could.)
  This has the practical consequence that, for the comparison, variations to the side of the common axis are disregarded. (The Top View Model is thus used essentially one-dimensional This is the consequence of the aim to use the model for investigations of variations in the horizontal plane after succesful completion of the study. Unfortunately this has not been possible and, in retrospect, a one-dimensional model would have sufficed.)
- The Top View Model is based upon a distribution of the potential over the vertical as described by (2.06.1);
  The distribution of the potential over the vertical has to be prescribed for the Side View Model.
  This has the practical consequence that, for the comparison, at the influx boundary the potential of the Top View Model is also prescribed for the Side View Model.

Practical aspects of the comparison:

- The bottom configurations must be the same for both models per tested prediction of the potential.
- The potential distribution over the vertical is made the same at the influx boundary of both models.
- The influx boundaries should be sufficiently far removed from the bottom section with the variation of the bottom slope.
- Predictions of the potential at the surface will be compared.

2.08 A model for comparative calculation

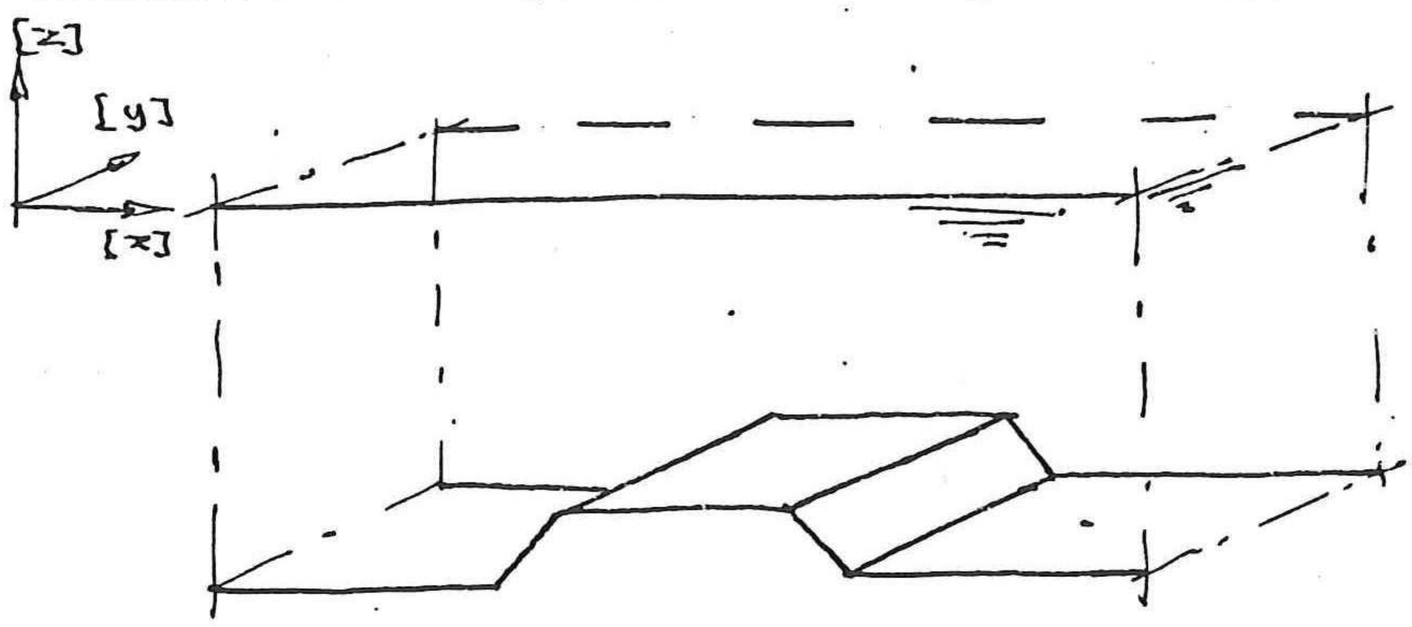
The important difference between the Side View Model and the Top View Model is the implied dependency on the bottom slope of the Top View Model.

The bottom contour which is to be used for the comparison of both models must therefore have a section characterised by the bottom slope which may be varied.

A very simple bottom contour has been selected which fits this description:

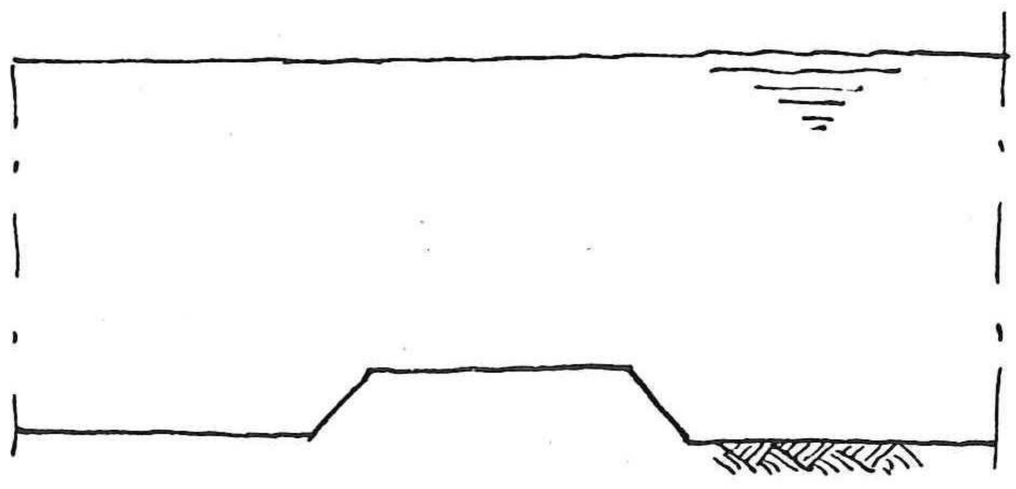
A trapezoidal submerged barrier on a flat, horizontal bottom.

Visualise the trapezoidal submerged barrier:



(2.08.1)

The Side View Model uses:



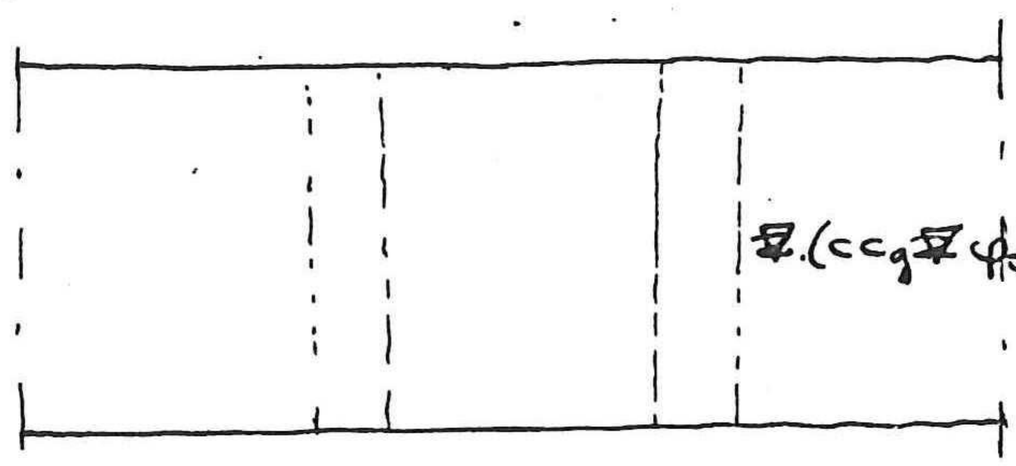
$$\frac{\partial}{\partial z}\varphi - \frac{\omega^2}{g}\varphi = 0$$

$$\nabla^2 \varphi = 0$$

$$\frac{\partial}{\partial n}\varphi = 0$$

(2.08.2)

The Top View Model uses:



$$\frac{\partial \varphi_o}{\partial n} = 0$$

$$\nabla.(cc_g \nabla \varphi_o) + k^2 cc_g \varphi_o = 0$$

$$\frac{\partial \varphi}{\partial n} = 0$$

(2.08.3)

where the values of c, c_g, and k have to be provided throughout the field.

For the side boundaries we have $\frac{\partial \varphi_o}{\partial n} = 0$, because variation in the y-direction is not considered.

The influx and efflux boundaries still have to be considered. This will be done next.

2.09 The influx and efflux boundaries

Up to this point two boundaries have not yet been considered: the influx and efflux boundaries, where the incoming and outgoing waves are to be described.

Both the Side View Model and the Top View Model require this information.

In this study only simple harmonic waves are considered.

These are described by:

$$\varphi = \hat{\varphi} \cdot e^{i(\omega t - kx)} \qquad (2.09.1)$$

(Recall that the time variation is disregarded.)

outgoing wave

Any wave coming from within the field must be able to pas freely through the boundary.

At any place within the field:

$$\frac{\partial \varphi}{\partial x} = -ik\hat{\varphi} \cdot e^{i(\omega t - kx)} \qquad (2.09.2)$$

that is:

$$\frac{\partial \varphi}{\partial x} = -ik\varphi \qquad (2.09.3)$$

This should also be the case for the outgoing waves at the boundary.

By using $\frac{\partial \varphi}{\partial n}$ instead of $\frac{\partial \varphi}{\partial x}$ the orientation of the efflux is in all cases correctly accounted for in the equations:

$$\frac{\partial \varphi}{\partial n} = -ik\varphi \qquad (2.09.4)$$

This is called the " radiation condition ": $\frac{\partial \varphi}{\partial n} = -ik\varphi$.

With the radiation condition imposed at the boundary waves can freely leave the field.

The radiation condition holds for both the Side View Model and the Top View Model.

incoming wave

It is assumed that a wave enters the field at the left-hand boundary. (See (2.08.2) and (2.08.3)).
At that boundary, at any time, two waves may exist together: an incoming wave and an outgoing wave:

$$\varphi = \varphi_i + \varphi_r \qquad (2.09.5)$$

where: φ_i : incoming wave
φ_r : outgoing wave

The outgoing wave, φ_r, can in the computation be determined only from:

$$\varphi_r = \varphi - \varphi_i \qquad (2.09.6)$$

This outgoing wave has to fulfill the radiation condition (2.09.4):

$$\frac{\partial \varphi_r}{\partial n} = -ik\varphi_i$$

$$\Rightarrow \frac{\partial \varphi}{\partial n} - \frac{\partial \varphi_i}{\partial n} = -ik\varphi + ik\varphi_i$$

$\frac{\partial \varphi_i}{\partial n}$ may be determined from the radiation condition for the incoming wave (for which the direction of travel is opposite to the normal at the boundary) :

$$\frac{\partial \varphi_i}{\partial n} = +ik\varphi_i$$

$$\Rightarrow \frac{\partial \varphi}{\partial n} = -ik\varphi + 2ik\varphi_i \qquad (2.09.7)$$

Taking $\hat{\varphi}_i = 1$ the influx condition becomes,

for the Top View Model:

$$\frac{\partial \varphi}{\partial n} = -ik\varphi + 2ik \qquad (2.09.8)$$

for the Side View Model:

$$\frac{\partial \varphi}{\partial n} = -ik\varphi + 2ik \frac{\cosh[k(h+z)]}{\cosh[kh]} \qquad (2.09.9)$$

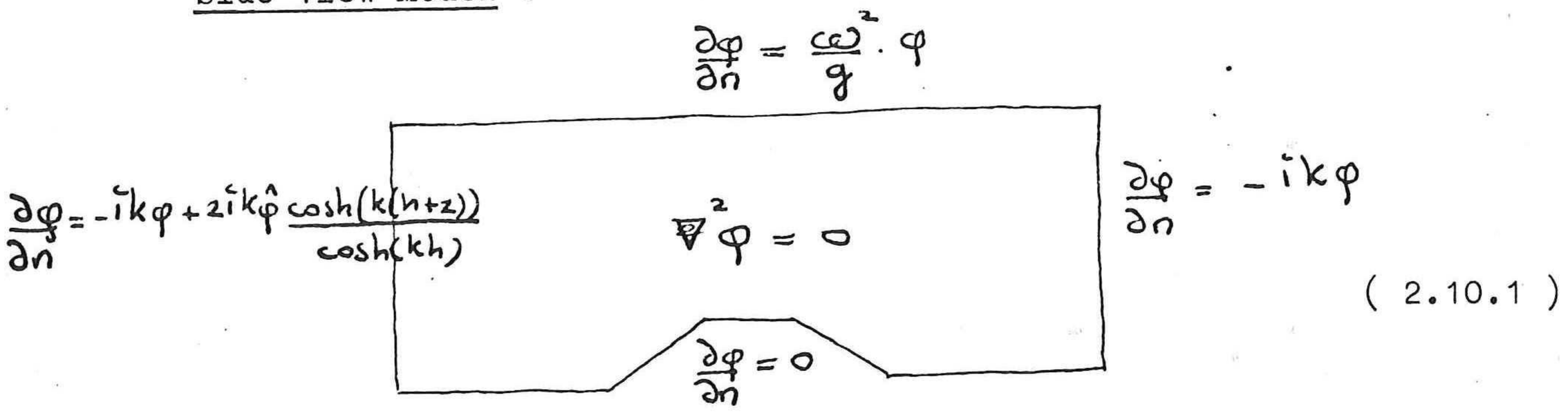
(As has been stated in section 2.07 the Side View Model has to be given the same distribution of the potential over the vertical as is implied for the Top View Model, at the influx boundary. At this boundary the bottom must be horizontal.)

The choice of $\hat{\varphi}_i = 1$ does not affect the generality of the description due to the linearity of the description.

2.10 Summary of the analytical description

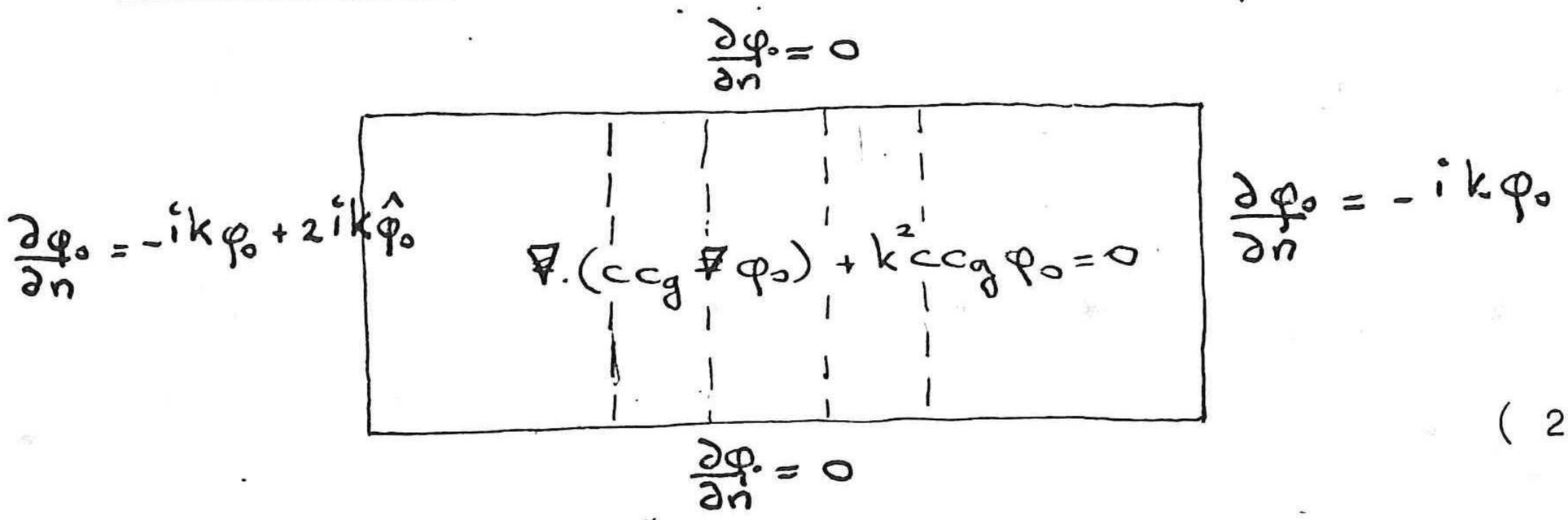
These are the two models which will be used for the comparative calculation:

Side View Model :



(2.10.1)

Top View Model :



(2.10.2)

(To ensure that the Top View Model uses the same bottom configuration as the Side View Model the values of c, c_g and k , as required for the Top View Model, will be determined using the values of the local depth in the Side View Model.)

3 Numerical Description

3.00 Introduction

The sets of equations (2.10.1), the Side View Model, and (2.10.2), the Top View Model, need to be written in numerical form, in order to prepare them for computation in the computer.

The translation from analytical to numerical form is done in this chapter.

The resulting sets of equations are: (3.07.1) and (3.07.2) .

The numerical form of the description is based on the Finite Element Package (" AFEP ") which has been developed at the T.H. Delft.

In order to provide some background information for the Finite Element Method, the chapter starts off with a verybrief general description of the concepts of the Finite Element Method, followed by a description of Lagrange Polynomials and the Galerkin Method, used in AFEP.

3.01 Finite Element Method

This section presents a very brief description of the concepts of the Finite Element Method.

For a more complete covering consult any of the many books on the topic, such as Zienkiewicz (1971), van Kan & Segal (1979), Strang & Fix (1973).

The Finite Element Method is based on " elements ", which are small subregions of the field in which the computation is to be done.
The elements are generated by 'covering' the field with a mesh.
The unknowns are determined in the nodes of the mesh.
By interpolating between the nodes, using an assumed interpolation function, an approximation of the continuous description can be achieved.

Depending on the total number of nodes and their distribution it is possible , within certain limits, to determine a balance between accuracy and laboriousness.

Basically the Finite Element Method works as follows (Segal, 1975):

- a finite continuous region is divided into subregions called " elements ".
- nodal points are selected within each element.
- it is indicated which unknowns are associated with each nodal point. It is not neccessary that these be the same at each nodal point.
- an approximate solution generating algorithm using the selected unknowns is provided for each element.
- the approximate solution, expressed in the selected unknowns, is determined for each element.
- the contributions from each element to the system of equations are stored in an element matrix resp. element vector.
- the system of equations is constructed by 'adding' the contributions per element together.
- the system of equations is solved for the unknowns, thereby providing the solution.

The essential part of this procedure is the description of the algorithm for the approximate solution per element. In doing this use can be made of a method of weighted residuals (Finlayson, 1972) in combination with interpolation functions.
In AFEP Lagrange polynomials of the first kind are used for the interpolation but also as weighting functions. This kind of method of weighted residuals is called the " Galerkin Method ". Both the Galerkin Method and the Lagrange polynomials will now be described.

3.02 The Galerkin Method

The Galerkin Method is a method of weighted residuals which uses the same functions for the weighting as for interpolation.

The method of weighted residuals aims at reducing the residual which results when an operation is performed in discrete space instead of in continuous space.

Let the operation in continuous space be :

$$L.\varphi = 0 \qquad (3.02.1)$$

where: L : linear operator.

In discrete space this will generally be:

$$L\,\tilde{\varphi} = R \qquad (3.02.2)$$

where: $\tilde{\varphi}$: the approximation of φ

R : residual

In order to minimalise the residual (that is: to obtain a better approximation of the operation in the discrete form) a weighting function is introduced in the operation.

A requirement for this weighting function is that it nullifies the residual, at least averaged over an interval " F ":

$$\int_F R.W.dF = 0 \qquad (3.02.3)$$

where: W : weighting function.

The operation then becomes:

$$\int_F L.\tilde{\varphi}.W.dF \qquad (3.02.4)$$

In discrete form, continuous functions are described using interpolation functions between the nodes. The Galerkin Method uses the same function Φ_j as weighting function ($W =: \Phi_j$) and as interpolation function ($\tilde{\varphi} =: \varphi \cdot \Phi_i$).

Using the Galerkin Method the operation will be:

$$\int_F L.\tilde{\varphi}.\Phi_j.dF \qquad (3.02.5)$$

where: F : interval of the weighting operation
Φ : interpolation polynomial
j : nodal index
L : linear operator

In this study the Galerkin method is used in the translation of the analytical description of the models into numerical form.

The interpolation is done using Lagrange Polynomials.

3.03 Lagrange Polynomials

A continuous function can be approximated by a discrete function using nodal points (where the values of the unknowns are determined) and interpolation polynomials.

A general interpolation formula is the Lagrange Interpolation Formula (Abramowitz, 1970)

$$f\langle x\rangle = \sum_{o}^{n} l_i\langle x\rangle . f\langle x_i\rangle + R_n\langle x\rangle$$

where:
$$l_i = \frac{\pi_n\langle x\rangle}{(x-x_i)\pi_n\langle x_i\rangle}$$
$$= \frac{(x-x_o)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_o)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$
$$R_n = \pi_n\langle x\rangle . \frac{f^{n+1}\langle \xi\rangle}{(n+1)\ !} \quad : \text{the remainders}$$

In this study only the linear Lagrange Polynomial will be used:

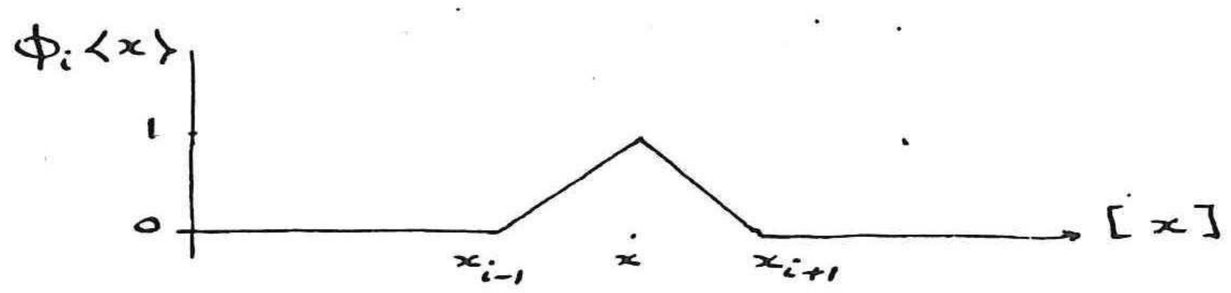
$$l_i\langle x\rangle = \frac{x-x_{i-1}}{x_i-x_{i-1}}$$

: the Lagrange Polynomial at point x_i.

Likewise for point x_{i-1} :

$$l_{i-1}\langle x\rangle = \frac{(x_i-x)}{(x_i-x_{i-1})}$$

Visualise the linear Lagrange Polynomials:

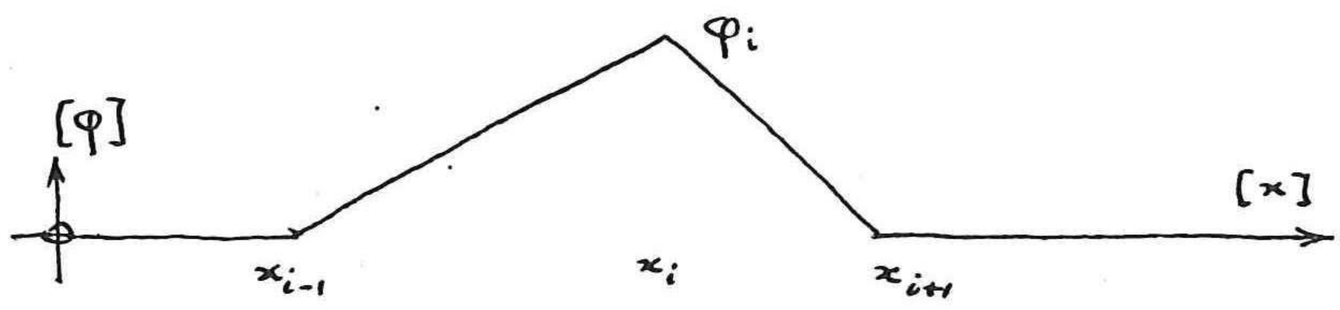


The interpolation function, x, is composed of linear Lagrange Polynomials.

For brevity $\Phi_i\langle x\rangle$ will be written as Φ_i , etc.

Describing a continuous function using the linear Lagrange Polynomials for the approximation is done as follows:

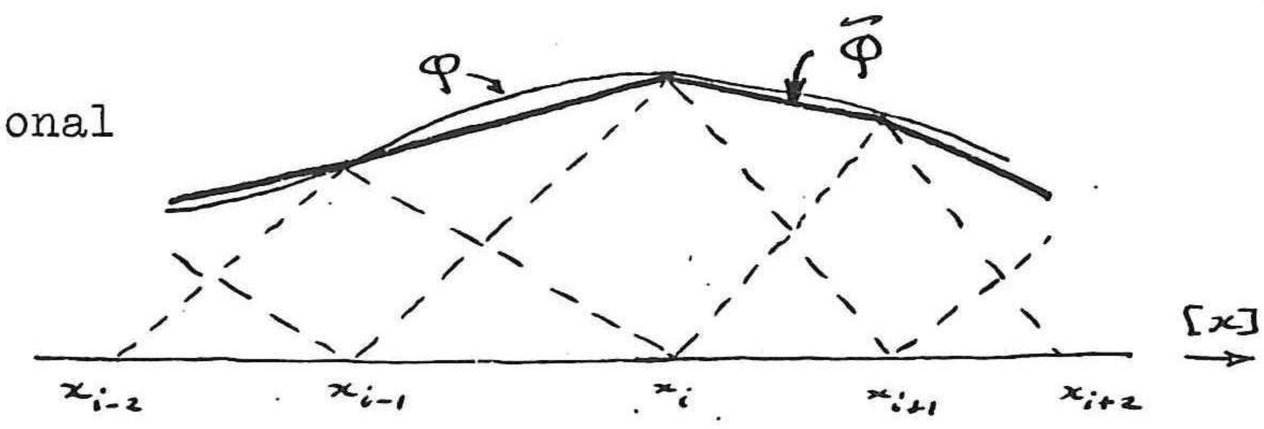
- at the nodes the value of the function is used.
- between the nodes interpolation is used, using the linear Lagrange Polynomials.
- the description over the interval will be composed of the local approximated descriptions using superposition.
- the local description will have the following form:



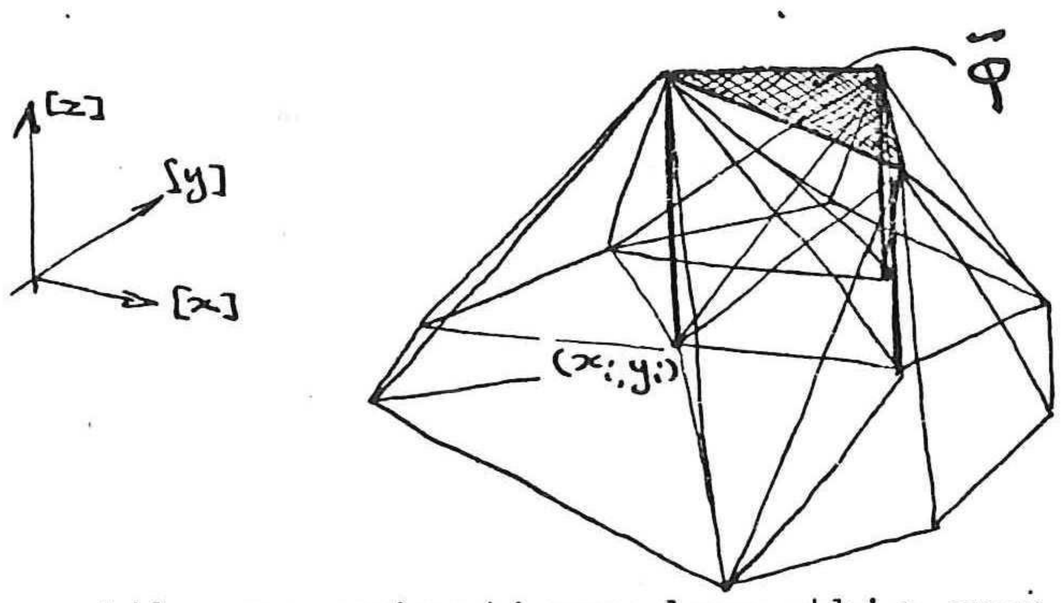
where: φ_i : the value of φ at point x_i ($x_i = \varphi\langle x_i \rangle$)

Visualise the result of an approximation done this way:

one dimensional



two dimensional (the local description uses $\Phi_i\langle x, y \rangle$)



In dealing with approximations done this way the generalised differential of the Sobolev space is to be used, due to non-differentiability of the function $\tilde{\varphi}$ at the

nodes and boundaries of the element. See Segal (1974) for a more detailed description.

3.04 Algorithm for the translation

The following is a brief outline of the method for rewriting the analytical equations ((2.10.1),(2.10.2)) into numerical form ((3.07.1),(3.07.2)):

1)- Rewrite the field equations using the Galerkin Method (3.02.5) using linear Lagrange Polynomials for the approximative description in discrete form.

$$L\varphi = 0 \quad \rightarrow \quad \int_F L\,\tilde{\varphi}\,\Phi_i\,dF$$

2)- Apply Gauss' theorem:

$$\text{e.g.:} \int_F \nabla^2\tilde{\varphi}\,\Phi_j\,dF \quad \Leftrightarrow \quad -\int_F \nabla\tilde{\varphi}\,\nabla\Phi_j\,dF + \int_\Gamma \nabla(\tilde{\varphi}\,\Phi_j)\,d\Gamma$$

3)- Apply Greens' theorem:

$$\int_\Gamma \nabla(\tilde{\varphi}\,\Phi_j)\,d\Gamma = \int_\Omega \frac{\partial\tilde{\varphi}}{\partial n}\,\Phi_j\,d\Omega$$

4)- Substitute the boundary conditions. (In this case these are of the general form: $\frac{\partial\varphi}{\partial n} = -\dot{a}\dot{\varphi} + \dot{q}$).

5)- Provide the computer with an algorithm for the evaluation of the integral.

6)- Store, separately, the contributions from each element to the system of equations (as described in section 3.01) to the element matrix resp. the element vector.

Due to the difference between field elements and boundary elements the contributions from the field elements and the boundary elements are treated differently.

7)- Assemble all the contributions to the matrix and the vector and solve the system $\vec{\vec{A}}\,\vec{\varphi} = \vec{g}$

Standard methods may be used for this purpose.

These steps will be applied to the equations (2.10.1) and (2.10.2).

3.05 Describing the equations in numerical form

In the following the Side View Model and the Top View Model will be considered together so as to show the similarities between the operations performed on the two models.

The field equations are:

Side View Model: $$-\nabla^2 \varphi = 0$$

Top View Model : $$-\nabla.(cc_g \nabla \varphi_0) - k^2 cc_g \varphi_0 = 0$$

(The negative sign is chosen in order to adhere to AFEP usage.)

1)- Apply Galerkins Method:

S.V.M. $$-\varphi_i \iint_F \nabla^2 \Phi_i \, \Phi_j \, dF = 0$$

T.V.M. $$-\varphi_i \iint_F \{ \nabla . (c c_g \nabla \Phi_i) - k^2 c c_g \Phi_i \} \Phi_j \, dF = 0$$

In this description use has been made of the property of interpolation by linear Lagrange Polynomials:

$$\tilde{\varphi} = \varphi \langle x_i \rangle . \Phi_i =: \varphi_i \Phi_i \qquad \text{(summation being implied)}$$

From now on ∇^2 is written as ∇ , the two dimensionality already being implied in the integration sign.

2)- Apply the Gauss' theorem and Green's theorem.

3)-

S.V.M. $$+\varphi_i \iint_F (\nabla \Phi_i)(\nabla \Phi_j) \, dF - \oint_\square \frac{\partial}{\partial n}(\varphi_i \Phi_i) \, \Phi_j \, d\square = 0$$

T.V.M. $$+\varphi_i \iint_F (c c_g \nabla \Phi_i \nabla \Phi_j - k^2 c c_g \Phi_i \Phi_j) dF - \oint_\square c c_g \frac{\partial}{\partial n}(\varphi_i \Phi_i) \Phi_j \, d\square = 0$$

where: $\square$: boundary

4)- Substitute the general boundary conditions

$$\frac{\partial \varphi}{\partial n} = - a \varphi + g$$

S.V.M. $$+\varphi_i \iint_F (\nabla \Phi_i)(\nabla \Phi_j) \, dF \qquad - \Sigma \int_{\square_k} (-a \varphi \Phi_i + g) \Phi_j \, d\square = 0$$

T.V.M. $$+\varphi_i \iint_F \{ c c_g (\nabla \Phi_i . \nabla \Phi_j - k^2 \Phi_i \Phi_j) \} dF \qquad - \Sigma \int_{\square_k} c c_g (-a \varphi \Phi_i + g) \Phi_j \, d\square = 0$$

(The boundary may be considered in parts and described in the summation in the equations.)

Separating the Real and Imaginary parts of the complex numbers:

(The Real part is given index " 1 ", the Imaginary part has index " 2 " .)

S.V.M.

$$\begin{cases} + \varphi_{1i} \oint_F (\nabla\Phi_i)(\nabla\Phi_j)\,dF \quad - k\sum \int_{\square_k} (-a_{1i}\varphi_{1i}\Phi_i + a_{2i}\varphi_{2i}\Phi_i + g_{1i})\ \Phi_j\, d\square \\ \& \\ + \varphi_{2i} \oint_F (\nabla\Phi_i)(\nabla\Phi_j)\,dF \quad - \sum \int_{\square_k} (-a_{2i}\Phi_{1i}\Phi_i - a_{1i}\varphi_{2i}\Phi_i + g_{2i})\,\Phi_j\, d\square \end{cases}$$

T.V.M.

$$\begin{cases} + \varphi_{1i} \oint_F cc_g\,(\nabla\Phi_i\nabla\Phi_j - k^2\Phi_i\Phi_j)\,dF \quad - \sum \int_{\square_k} cc_g(-a_{1i}\varphi_{1i}\Phi_i + a_{2i}\varphi_{2i}\Phi_i + g_{1i})\,\Phi_j\, d\square \\ \& \\ + \varphi_{2i} \oint_F cc_g\,(\nabla\Phi_i\nabla\Phi_j - k^2\Phi_i\Phi_j)\,dF - \sum \int_{\square_k} cc_g(-a_{2i}\varphi_{1i}\Phi_i - a_{1i}\varphi_{2i}\Phi_i + g_{2i})\,\Phi_j\, d\square \end{cases}$$

Rewriting these sets of equation using vector description and the algorithm:

$$\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

S.V.M.

$$\begin{bmatrix} \oint_F \nabla\Phi_i\nabla\Phi_j\,dF + \sum \int_{\square} a_{1i}\Phi_i\Phi_j\,d\square & -\sum \int_{\square_k} a_{2i}\Phi_i\Phi_j\,d\square \\ \sum \int_{\square_k} a_{2i}\Phi_i\Phi_j\,d\square & \oint_F \nabla\Phi_i\nabla\Phi_j\,dF + \sum \int_{\square_k} a_{1i}\Phi_i\Phi_j\,d\square \end{bmatrix} \begin{bmatrix} \varphi_{1i} \\ \varphi_{2i} \end{bmatrix} = \begin{bmatrix} \sum \int_{\square} g_{1i}\Phi_j\,d\square \\ \sum \int_{\square} g_{2i}\Phi_j\,d\square \end{bmatrix}$$

T.V.M.

$$\begin{bmatrix} \oint_F cc_g(\nabla\Phi_i\nabla\Phi_j - k^2\Phi_i\Phi_j)dF + \sum\int_{\square_k} cc_g(a_{1i}\Phi_i\Phi_j)d\square & -\sum\int_{\square_k} cc_g a_{2i}\Phi_i\Phi_j d\square \\ \sum\int_{\square_k} cc_g a_{2i}\Phi_i\Phi_j d\square & +\oint_F cc_g(\nabla\Phi_i\nabla\Phi_j - k^2\Phi_i\Phi_j)dF + \sum\int_{\square_k} cc_g a_{1i}\Phi_i\Phi_j d\square \end{bmatrix} \begin{bmatrix} \varphi_{1i} \\ \varphi_{2i} \end{bmatrix} = \begin{bmatrix} \sum\int_{\square_k} cc_g g_{1i}\Phi_j d\square \\ \sum\int_{\square_k} cc_g g_{2i}\Phi_j d\square \end{bmatrix}$$

It is convenient to use the property of superposition to separate the matrices into parts.

This also has the advantage of allowing a separation into matrices describing the different aspects:

the field condition versus the boundary conditions.

S.V.M.

$$\begin{bmatrix} \oint_F \nabla\Phi_i\nabla\Phi_j\,dF & 0 \\ 0 & \oint_F \nabla\Phi_i\nabla\Phi_j\,dF \end{bmatrix} \begin{bmatrix} \varphi_{1i} \\ \varphi_{2i} \end{bmatrix} + \qquad (3.05.1)$$

$$+ \begin{bmatrix} \sum\int_{\square_k} a_{1i}\Phi_i\Phi_j\,d\square & -\sum\int_{\square_k} a_{2i}\Phi_i\Phi_j d\square \\ \sum\int_{\square_k} a_{2i}\Phi_i\Phi_j d\square & \sum\int_{\square_k} a_{1i}\Phi_i\Phi_j d\square \end{bmatrix} \begin{bmatrix} \varphi_{1i} \\ \varphi_{2i} \end{bmatrix} = \begin{bmatrix} \sum\int_{\square_k} g_{1i}\Phi_j d\square \\ \sum\int_{\square_k} g_{2i}\Phi_j d\square \end{bmatrix}$$

T.V.M. (3.05.2)

$$\begin{bmatrix} \oint_F cc_g(\nabla\Phi_i\nabla\Phi_j - k^2\Phi_i\Phi_j)dF & 0 \\ 0 & \oint_F cc_g(\nabla\Phi_i\nabla\Phi_j - k^2\Phi_i\Phi_j)dF \end{bmatrix} \begin{bmatrix} \varphi_{1i} \\ \varphi_{2i} \end{bmatrix} +$$

$$+ \begin{bmatrix} \sum\int_{\square_k} cc_g a_{1i}\Phi_i\Phi_j d\square & -\sum\int_{\square_k} cc_g a_{2i}\Phi_i\Phi_j d\square \\ \sum\int_{\square_k} cc_g a_{2i}\Phi_i\Phi_j d\square & \sum\int_{\square_k} cc_g a_{1i}\Phi_i\Phi_j d\square \end{bmatrix} \begin{bmatrix} \varphi_{1i} \\ \varphi_{2i} \end{bmatrix} = \begin{bmatrix} \sum\int_{\square_k} cc_g g_{1i}\Phi_j d\square \\ \sum\int_{\square_k} cc_g g_{2i}\Phi_j d\square \end{bmatrix}$$

It can be seen that for both models the first section is a (symmetrical) matrix describing the field condition in the element.
The second part is a (non symmetrical) matrix for the boundary elements.
The last part, the right hand side of the equation, is the vector of the boundary elements.
The three parts are called, respectively:

- the element matrix of the field,
- the element matrix of the boundary,
- the element vector of the boundary.

Note the similarity between the two models.

5)- Evaluate the integrals.

Equations (3.05.1) and (3.05.2) contain integrals. That is an operation which has to be defined for the computer.
This can be done either by providing an algorithm which computes the value of the solution, or, in the case of a simple integration, by providing the solution directly. Because of the simplicity of the integration involved in this study the latter method has been used.
It is demonstrated how this is done using the boundary element. For the field element the integration is done analogously; this is not demonstrated.

Lagrange Polynomials are used for the description of the approximation of the unknowns in the element.

For a two-point boundary element the Lagrange Polynomial will have either of two forms:

The nodes have been labeled ① resp ②.

In an integral a multiple of combinations of Φ_i may exist,

e.g.: $\int_{\square} c\, c_g\, a_{1i} \Phi_k \Phi_j \, d\square = \int_{\square} c\, c_g\, a_1 (\Phi_k \Phi_i \Phi_j)\, d\square$

(a_{1i} has been rewritten using Lagrange Polynomials.)

In the integration i,j,k may each take either value ① or ②. Thus a number of possible combinations of Φ_1, Φ_2 may exist in the integration, depending upon the nodal point from which the computation is done (① or ②) and which combination is made to the other nodes within the element (① or ②).

As an example it is demonstrated how this is done for the evaluation of the integral $\int_{\square} c\, c_g\, a_1 (\Phi_k \Phi_i \Phi_j)\, d\square$.

A)-

For $\Phi_k =: \Phi_1$; $\Phi_i =: \Phi_1$; $\Phi_j =: \Phi_1$

$$\int_{\square} cc_g a_1 (\Phi_1)^3 \, d\square = \int_0^H cc_g a_1 \left(\frac{x}{H}\right)^3 dx = cc_g a_1 \frac{x^4}{4H^3} \Big|_0^H$$

$$= cc_g a_1 \cdot \frac{H}{4}$$

B)-

For $\Phi_k =: \Phi_1$; $\Phi_i =: \Phi_2$; $\Phi_j =: \Phi_2$

$$\int_0^H cc_g a_1 \left(\frac{x}{H}\right)\left(1-\frac{x}{H}\right)^2 dx = cc_g a_1 \left(\frac{x^2}{2H} - \frac{2x^3}{3H^2} + \frac{x^4}{4H^3} \right) \Big|_0^H =$$

$$= cc_g a_1 \cdot \frac{H'}{12}$$

Following this method:

$$\int_{\square} \Phi_i \, d\square = \frac{H}{2}$$

$$\int_{\square} \Phi_i \Phi_j \, d\square = \frac{H}{3} \quad (i=j)$$

$$= \frac{H}{6} \quad (i \neq j)$$

Per element the matrix description of φ_{1i} and φ_{2i} may be expanded.

i = 1,2,3 (field element) produces a 6 x 6 matrix,

i = 1,2 (boundary element) produces a 4 x 4 matrix.

(The right hand part of the equation for the boundary element will be a 1 x 4 matrix: " the element vector ".) evaluating the integrals in the manner described above while expanding the matrix over the number of nodes per element the equation will take the form indicated below:

The general form of the matrix of the boundary element (compare with the second term in equations (3.05.1) and (3.05.2).) :

$$\frac{H}{12} \begin{Bmatrix} 3d11+d12 & -(3d21+d22) & d11+d12 & -(d21+d22) \\ 3d21+d22 & 3d11+d12 & d21+d22 & d11+d12 \\ d11+d12 & -(d21+d22) & 3d11+d12 & -(3d21+d22) \\ d21+d22 & d11+d12 & 3d21+d22 & 3d11+d12 \end{Bmatrix}$$

The general form of the vector of the boundary element (compare with the third term in equations (3.05.1) and (3.05.2).) :

$$\frac{H}{6}\begin{Bmatrix} 2e11+\ e21 \\ 2e21+\ e22 \\ \ e11+2e12 \\ \ e21+2e22 \end{Bmatrix}$$

where: H : the length of the interval

d_{ij} : the coefficient of Φ_i in node "j".
e_{ij} : the coefficient of Φ_i in node "j".

$d_{ij} = a_{ij}$ for the Side View Model
$= a_{ij}.cc_g$ for the Top View Model

$e_{ij} = g_{ij}$ for the Side View Model
$= g_{ij}.cc_g$ for the Top View Model

(see equations (3.05.1) and (3.05.2).)

For the matrices of the field element the description is obtained in a similar way.

3.06 Computing element matrices and vectors

The subroutines (of the computer program) which compute the contributions of each element to the system of equations (following the method described in section 3.05) are called:

Side View Model:

ELM148 field element matrix

ELM149 boundary element matrix

ELR149 boundary element vector

Top View Model:

ELM301 field element matrix

ELM302 boundary element matrix

ELR302 boundary element vector

That is:

ELM148 describes the Laplace equation

ELM301 describes the refraction - diffraction equation

The boundary condition: $\frac{\partial \varphi}{\partial n} = -a\varphi + g$

is described by:

ELM 149 and ELR149 (for the Sido View Model)

ELM 302 and ELR302 (for the Top View Model)

A listing of these subroutines is presented in Appendix 7.

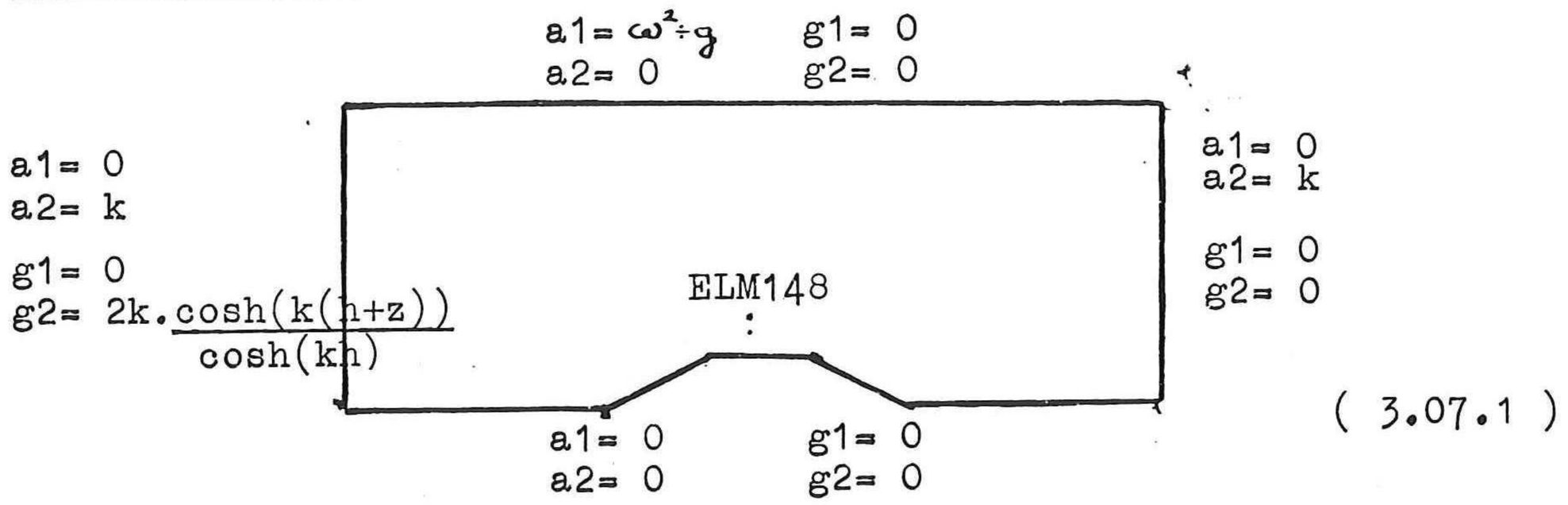
Note that the boundary elements all require input: $\acute{a}$, $\acute{g}$, and all the elements of the Top View Model require the input: cc_g (for the field elements: kcc_g).

These data have to be provided in the program for computation.

3.07 Summary of the numerical description

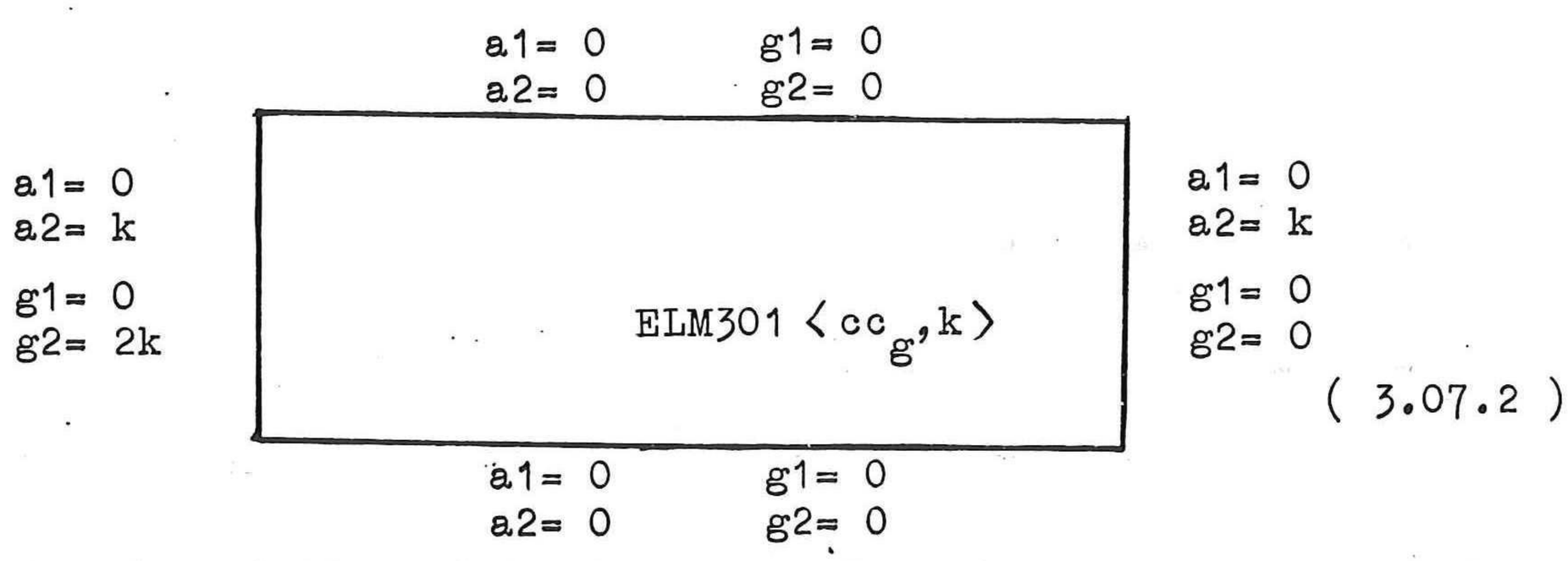
The numerical form of the two models which will be used for the comparative calculation is:

Side View Model :



(3.07.1)

Top View Model :



(3.07.2)

For the description of the boundary elements,

the Side View Model uses: ELM149 and ELR149

the Top View Model uses: ELM302 and ELR302.

The required input data are indicated above.

4 Description of the computation

4.00 Introduction

This chapter presents the general structure of the computer program (and the way AFEP is used) for the comparison of the two models: the Side View Model and the Top View Model.

The comparison is done by, for each model, computing the values of the potential of the wave field for identical configurations of the bottom profile and with identical incoming wave.

Equations (3.07.1) and (3.07.2) form the basis of the computer program.

The listing of the program and an introduction to understanding the subroutines can be found in Appendix 4.

4.01 Basic structure of the program

The program contains the complete description of all the actions to be performed by the computer for the achieving of the desired goal: obtaining predictions of the potentials of the wave field for the two models: the Side View Model and the Top View Model. (These predictions are then compared.)

For both models the line of action is similar:

1) - Preparing the computer for the job
2) - Generate elements for the finite element method
3) - Prescribe the equations for the approximate solution per element (ELM148,ELM149,...)
4) - Provide the required input for " 3) "
5) - Compute the contribution from each element to the system of equations.
6) - Assemble the large matrix " $\vec{\vec{A}}$ " and the vector " $\vec{g}$ " from the local contributions.
7) - Solve "$\vec{\varphi}$" from the system of equations: $\vec{\vec{A}}\vec{\varphi} = \vec{g}$.
8) - Present the solution.
9) - Signal the computer that the job is done.

The steps 1 through 9 are described in the following paragraphs. The program can be found in Appendix 4.

4.01.1 Preparing the computer for the job

A number of things have to be taken care of before the job can be executed by the computer. (the " job " is the set of statements which command the actions to be performed for the computation.) .

The administrative affairs of how the computer is to treat and execute the program is described in the Job Control Language ("JCL") section of the program:

- Assign a memory space for the job and a maximal time interval to be used.
- Indicate the account where the costs of the job are to be charged.
- Indicate which computer language and compiler are to be used.
- Specify all input and output datasets.
- Specify all libraries to be used (AFEP, ...).
- Provide the program of the job.
- Provide datacards as required by the job.
- Mark the end of the program.

AFEP forms an essential part of the structure of the program. Information concerning all AFEP subroutines is to be found in the AFEP User Manual.

The job program consists of a main program and a set of subroutines which are called by the main program.

In case the subroutines are not already available in one of the libraries linked to the program they have to be added after the main program.

Fortran IV is used as the computer language for the program of this study.
This requires that in every main program the following information is presented:

- the amount of memory space to be reserved for every array which is used.
- the code which is used for variables which are to be treated as IReal numbers.
- the variables which are to be used in "common blocks".

For more detailed information on Fortran IV consult any of the available books, e.g. Chirlian (1973).

4.01.2 Generate elements for the finite element method

The field is divided into subregions called " elements " using an AFEP subroutine: " MESB2D ". (See the AFEP Manual for details.)

This meshgenerator creates a mesh within vertices on the boundary of which the user provides the coordinates.
The user also has to specify how many nodal points are

to be made on each section of the boundary and how the spacing is to be done. (Segal, 1975)
The " meshgenerator " then uses this information to generate the mesh in the field.

Subroutine " INTAKE " provides the information the mesh-generator requires. (See Appendix 4.)

The form of the field (laid down in INTAKE) is varied in this study. The variation of INTAKE is done by the subroutine " VARY ".

4.01.3 Prescription of the equations

The equations to be used for the computations, for both field and boundaries, are indicated by their number in a datacard input section at the end of the program following AFEP convention.
(See (3.07.1), (3.07.2) and the AFEP User Manual).

4.01.4 Providing the input for the equations

As is described on page - 44 - the equations (3.07.1) and (3.07.2) require input.

This information is made available using arrays " USER " and " IUSER " which, following AFEP convention provide the input for the subroutines describing the equations (ELM148, ELM149, ELR149; ELM 301, ELM302, ELR302).

(See AFEP User Manual, Appendix 4 and Appendix 5).

Before the values of the required parameters of the input (a, c, c_g, k) can be stored they have to be calculated.

This is done in subroutine " DEEP " (Side View Model) resp. subroutine " LOCAL " (Top View Model).

Subroutine " AUSER " stores the calculated values in the arrays " USER " and " IUSER ". (Appendix 5).

4.01.5 Computing matrices and vectors per element

AFEP subroutine " COMMAT " does the job.

4.01.6 Assembling matrix " $\vec{\vec{A}}$ " and vector " $\vec{g}$ "

Done by AFEP subroutines " MATRIX " resp. " RHSIDE ".

4.01.7 Solving $\vec{\varphi}$ from $\vec{\vec{A}}\vec{\varphi} = \vec{g}$

AFEP subroutine " SOLVE " does the job. SOLVE is based on a direct method. Matrix $\vec{\vec{A}}$ is subjected to LU decomposition and used in the call of SOLVE in that form. SOLVE consists of 10 subroutines. A description can be found in the AFEP Programmers Guide.

4.01.8 Presenting the solution

The solution can be presented in several ways:

Subroutine	PRINRV	prints the values of the potential (φ_1, φ_2) at each nodal point.
	PLOTCN	makes a contourplot using the results of PRINRV. Input is required for PLOTCN; this is provided by subroutine INPLOT.
	GRAPH	prints a graph of the values along a specified boundary, drawing on the results of PRINRV.

In this study the last form of presentation has been used (using the surface boundary of the Side View Model and the corresponding side boundary of the Top View Model).

4.01.9 Signaling the end of the program

This is done by the statement " end " at the end of the program.

4.02 Using the computer program

The computer program has been constructed in such a way that equations (3.07.1) and (3.07.2) are each described completely in a program.

A link has been created between these two programs such that the parameters of the Top View Model (cc_g and k) are calculated using the values of the depth of the configuration of the Side View Model.

The position of the nodal points is ensured to be the same for both models (along the axis where the values of the predictions will be compared) by using the coordinates of the Side View Model (which have been generated by the mesh generator) for the Top View Model as well.

In this manner it is certain that the two models both compute the potentials for identical configurations.

A practical consequence for the use of the program is that the computation of the Side View Model always has to be done prior to the computation of the Top View Model for the same configuration.

The computer program, as it has been designed, readily permits variation of:
(See Appendix 4.)

- the bottom configuration (" ITEST ", " RATIO ")
- the waterdepth (" SCD ")
- the wavefrequency (" OMEGA ")

The next chapter describes how the program is tested.

Chapter 6 describes how the program is used.

5 Testing the program

5.00 Introduction

The computer program which has been described in chapter 4 (the program can be found in Appendix 4) has to be tested. This is done by using the program to predict the values of the potential ϕ for a configuration for which there exists another method of determining the values of the potential.

This chapter shows which configuration has been used for this purpose and how the predictions are verified.

5.01 Configuration for the verification

A flat and horizontal bottom has been selected as the configuration for the verification.

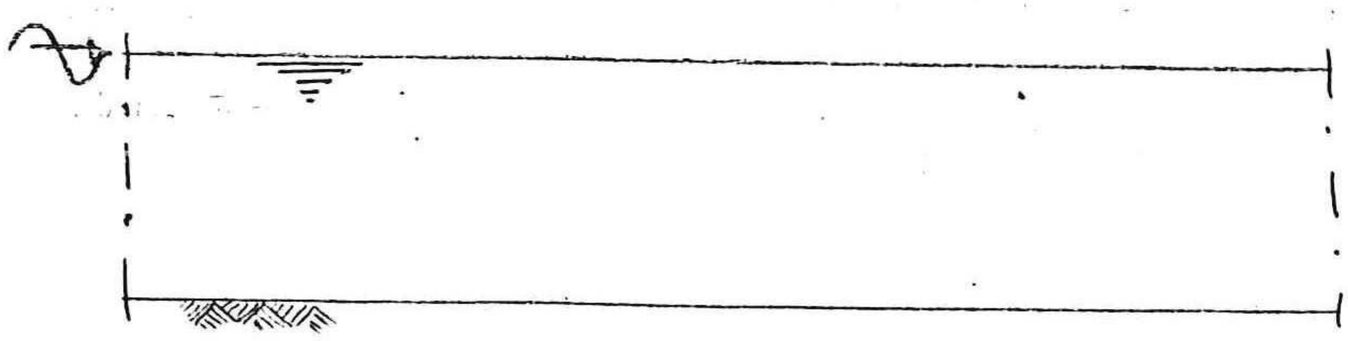


Figure 5.01.1 Sketch of the selected bottom configuration for the testing of the program.

The wave potential can be determined analytically for this case .

5.02 Verifying the predictions

There exist two ways for verifying the computed values of the wave potential φ :

1) The wavelength must be correctly predicted,

2) The magnitude of the potential, $\|\varphi\|$, must be invariant.

(A description of the relationship between the wave potential and its magnitude is provided in section 6.05)

5.02.1 Wavelength

The wavelength has been determined for two values of the waterdepth. These two values are the same as have been used in the computations for the configurations described in the next chapter.

To determine the wavelength for the flat and horizontal bottom case using analytical methods is a simple matter, described in any book on wave mechanics (e.g. Lamb, 1975).

To determine the wavelength for the same case from the computer results is also rather straightforward but does require some further description of the interpretation of the results as presented by the computer (this is done in section 7.02) and the way the accuracy of the thus obtained values (which will be done next).

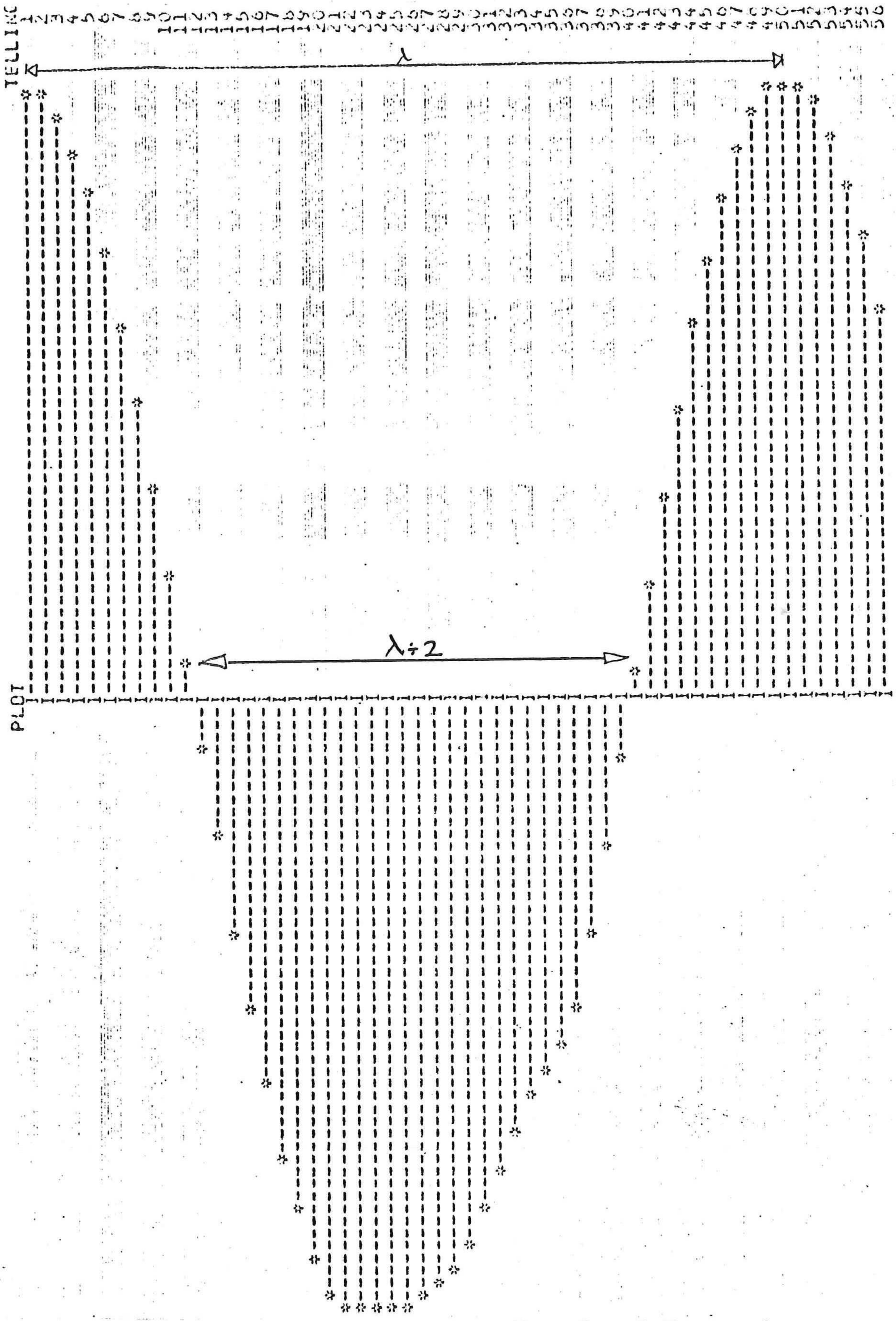


Figure 5.02.1.1

Example of the graph of the computed value of φ_1 from which the wavelength may be determined.

Figure (5.02.1.1) shows the computed values of the IReal part of the wave potential φ.

The figure clearly shows the wave pattern.

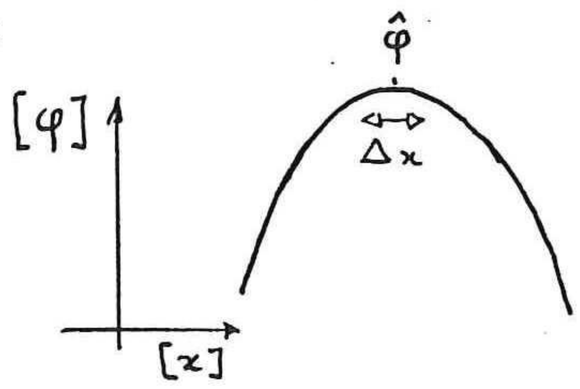
Observe that the depicted wave pattern is not exactly sinusoidal. This is caused by the fact that the steplength is not the same throughout the field. (The stepsize has been made smaller in the area where the greatest change in the value of the potential would be expected: over the dam. See figures (7.01.1) and (7.01.2).)

(A discussion on the numerical aspects of the chosen step-length can be found in Appendix 6.)

The exact location of each node can always be determined from the input data of the program.

Since the location of a node evidently does not have to coincide with the location of a maximum or minimum, an error will be made in determining the wavelength.

If the wavelength is determined from the location of the maxima (See Figure (5.01.1.1), the uncertainty of the value will be as great as the steplength, as follows from the following sketch:



$$x\langle\hat{\varphi}\rangle = x\langle\tilde{\hat{\varphi}}\rangle \pm \frac{\Delta x}{2}$$

$\hat{\varphi}$: max of φ, continuous function.

$\tilde{\hat{\varphi}}$: max of $\tilde{\varphi}$, approximate function.

Δx : steplength. (5.01.1.2)

(In the studied cases: $\Delta x = 0.11$)

A more accurate value of the wavelength may be obtained by interpolating for the location of the zero-crossings of the sinus function. In that case the uncertainty will be twice

as large as the round-off error of the value of the x-coordinate. (Which, for the studied cases is: $\pm$ 0.01 round-off .)

The results of the calculated and graphically determined values of the wavelength are compared:

Table 5.01.1.3

Comparison of the graphically determined wavelength versus the calculated wavelength for a horizontal flat bottom. (SVM= Side View Model; TVM= Top View Model)

Depth	wavelength SVM	wavelength TVM	calculated wavelength
0.4	(3.71 $\pm$ .02)	(3.71 $\pm$ .02)	(3.71 $\pm$ .01)
0.6	(4.39 $\pm$ .02)	(4.39 $\pm$ .02)	(4.38 $\pm$ .01)

These results show that the results of the computation do indeed correspond with the calculated values of the wavelength.

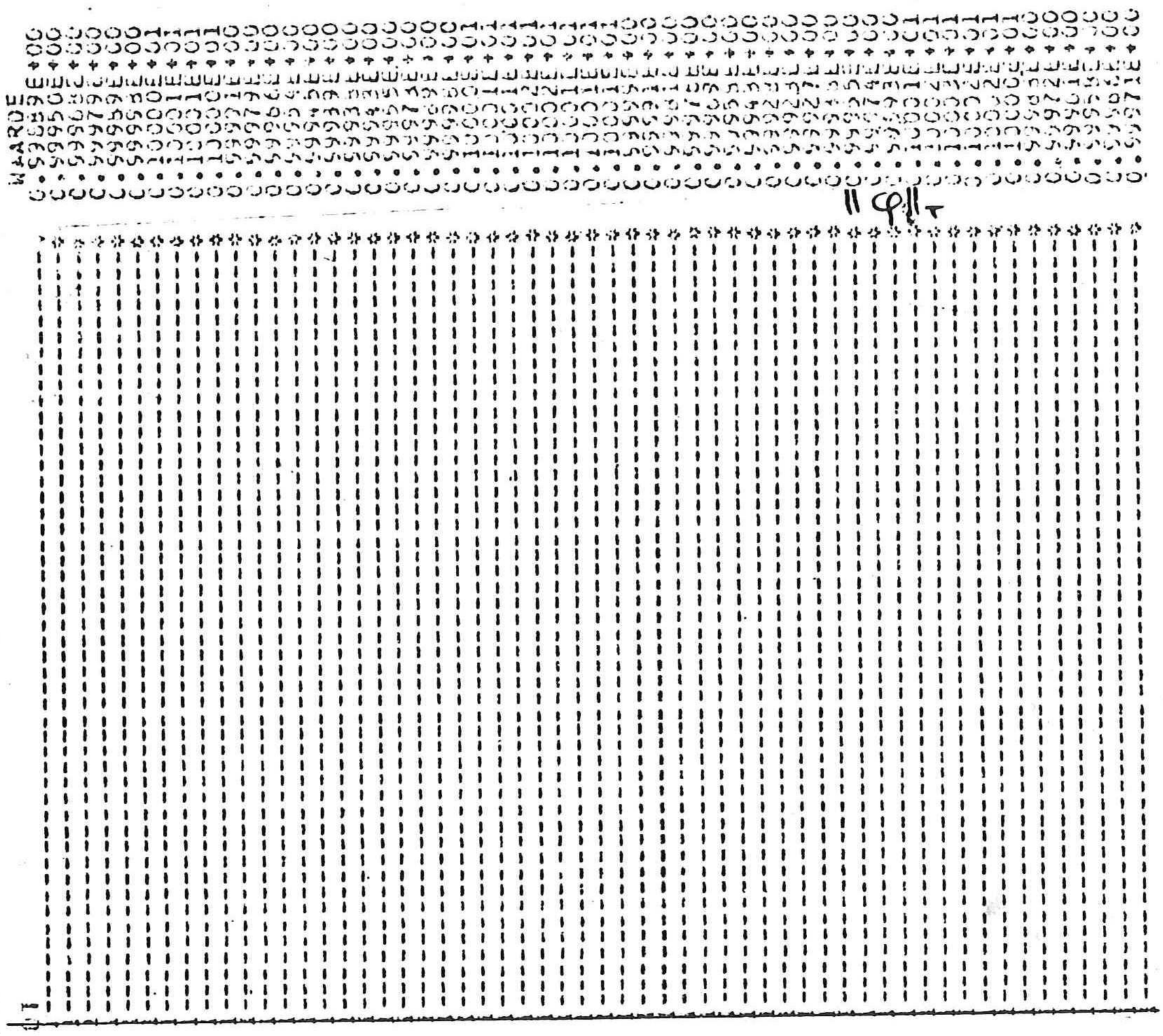
5.02.2 Magnitude of the potential: $\| \varphi \|$

Discussion of the relationship between the wave potential and its magnitude, as well as their interpretation, will be done in chapter 6.

At this moment it suffices to observe the behaviour of $\| \varphi \|$ for the case of a flat and horizontal bottom. In this case the value of $\| \varphi \|$ should be constant.

The values of $\| \varphi \|$ for both the Side View Model and the Top View Model are displayed in figures (5.02.2.2) and (5.02.2.3).

The two cases are the same as which have been used in section (5.02.1) for the verification of the computed wavelength.



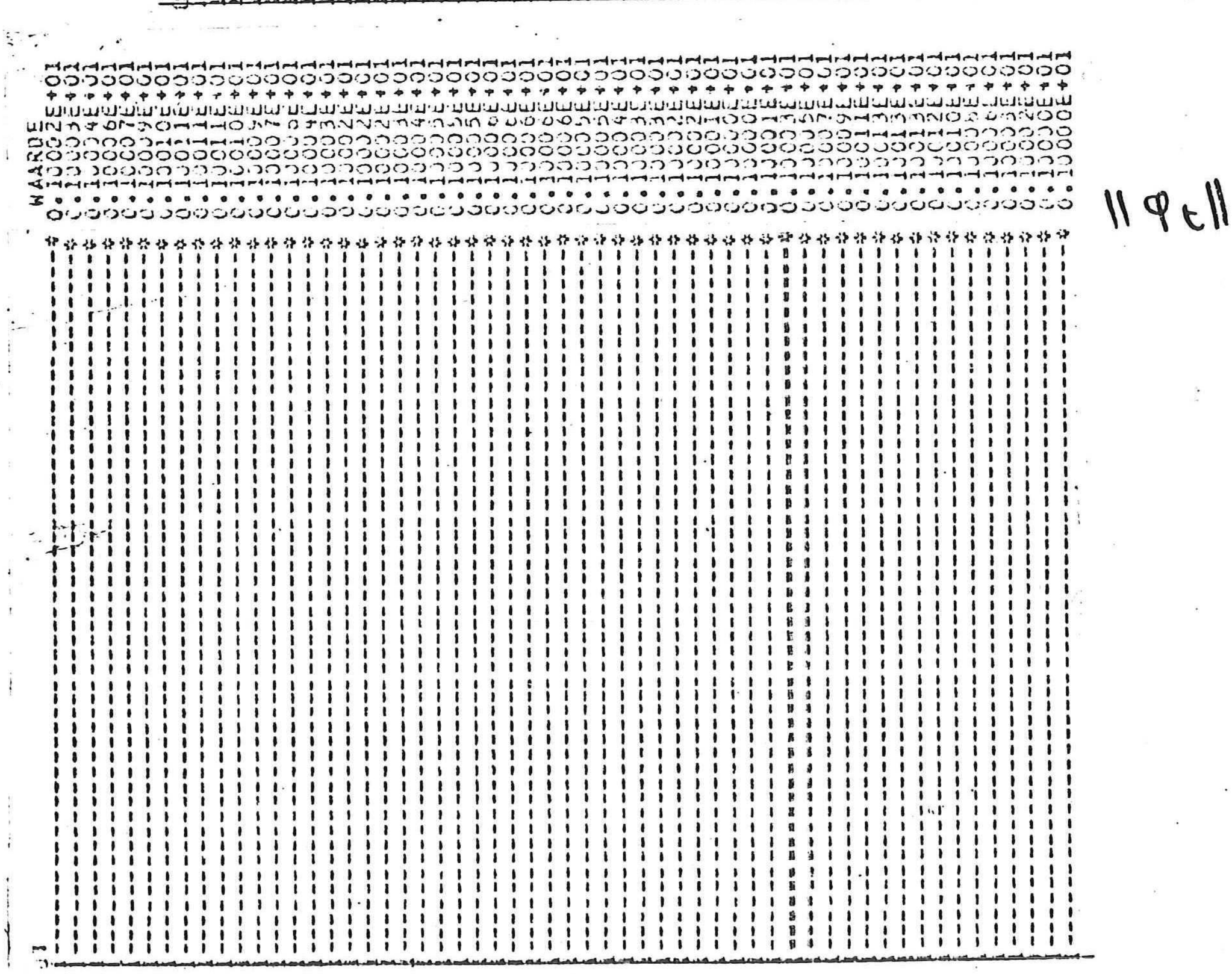
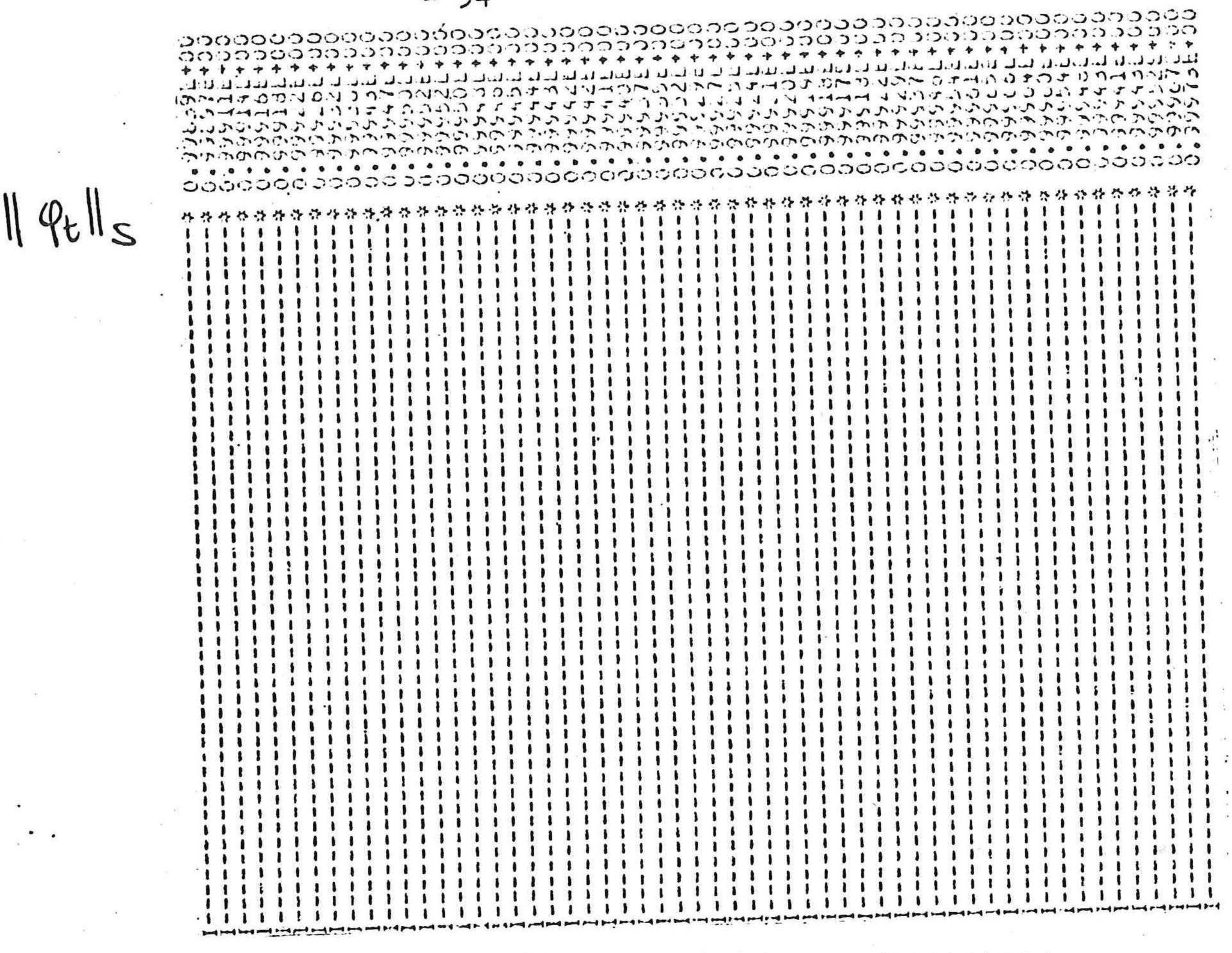


Figure 5.02.2.1 Predicted values of $\|\varphi_t\|_5$ and $\|\varphi_t\|_\tau$ for the case of a flat horizontal bottom, for CASE 1.



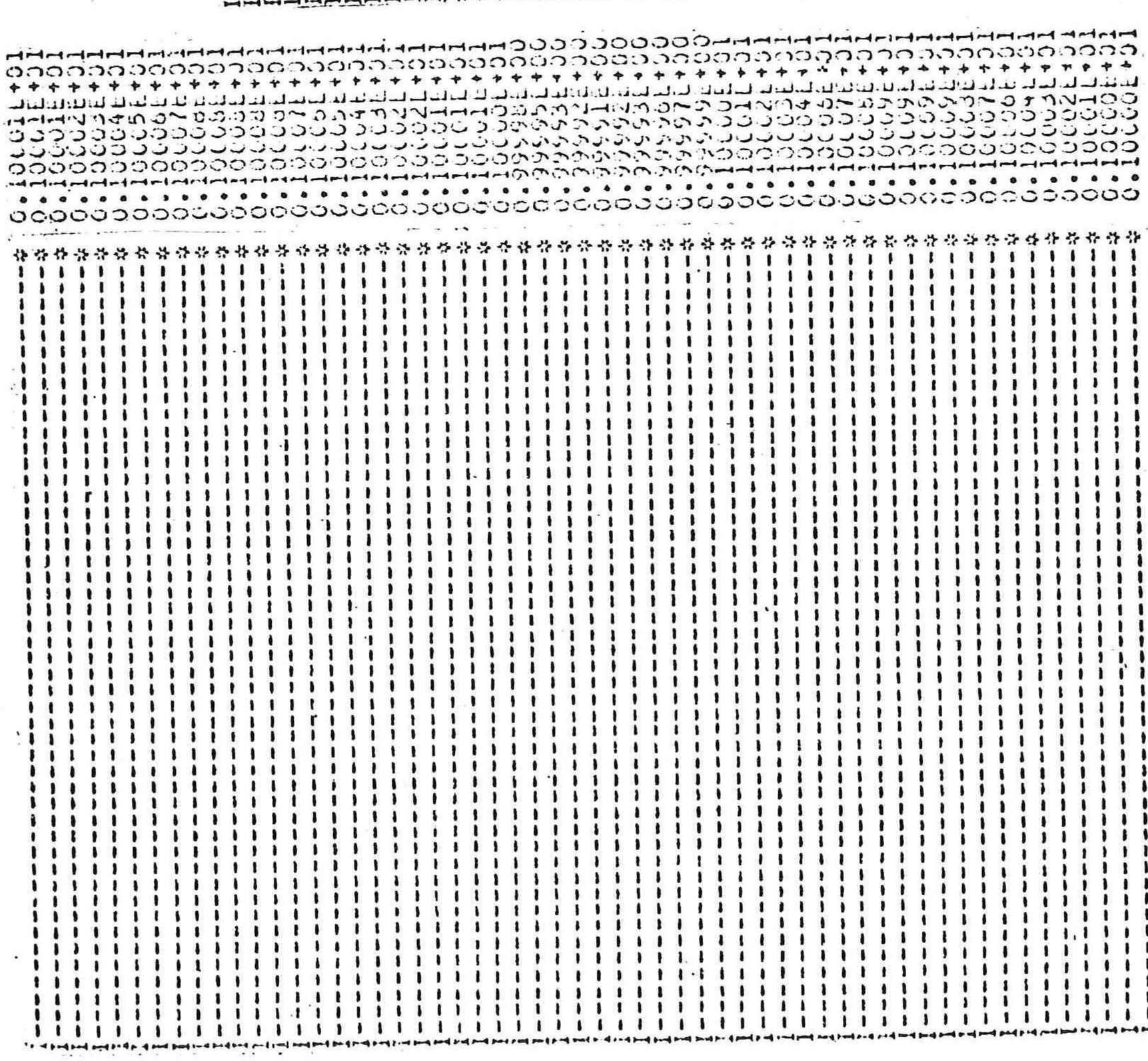


Figure 5.02.2.2 Predicted values of $\|\varphi_t\|_s$ and $\|\varphi_t\|_T$ for the case of a flat horizontal bottom, for CASE 2.

Inspection of the results shows that the predicted values of $\|\varphi\|$ are indeed constant for these cases:

Table 5.02.2.3

Predicted values of $\|\varphi\|$ for a flat horizontal bottom. (See figures (5.02.2.1) and (5.02.2.2)

Depth	Side View Model $\|\varphi\|$	Top View Model $\|\varphi\|$
0.4	(0.9996 $\pm$ 0.0004)	(1.0000 $\pm$ 0.0013)
0.6	(0.9994 $\pm$ 0.0003)	(1.0000 $\pm$ 0.0009)

5.03 Alternative verification

It has been possible to check the computations by the computer program for the Top View Model by comparing the results versus results obtained for identical cases using a one dimensional model bases on the Refraction-Diffraction equation. This model, made by N. Booy, produced identical results (for the compared cases) as the model which has been used in this study.

Appendix 6 shows the results of the model by N. Booy.

6 Experimentation

6.00 Introduction

The preceding chapters described how the Side View Model and the Top View Model have been derived and how a computer program has been made which, for both models, can compute the potentials of the wavefield for a selected bottom configuration.

This chapter shows how this program has been used for this study and which bottom configurations have been selected for the comparison.

Beginning with a summary of the parametric description for the general case of wave propagation and the specific case which is selected for this study, the chapter then continues with a description of the characteristic parameter which has been selected for the comparison. Finally a description of three main types of variations of the selected bottom configuration is presented.

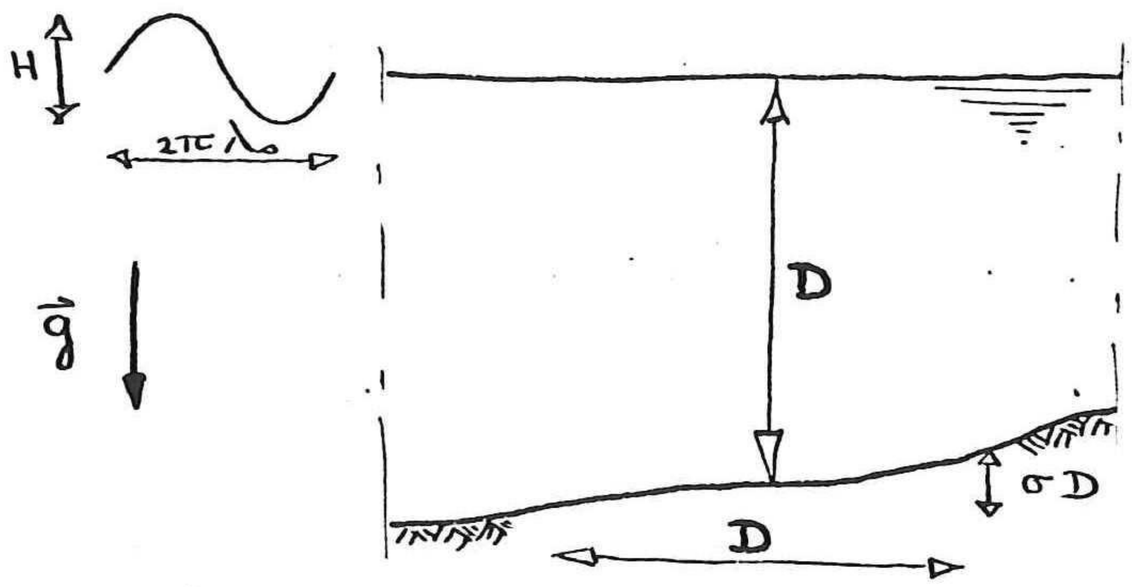
6.01 Goal of the test

The goal of the test is: to determine the difference in the prediction of the Side View Model and the Top View Model as a function of a difference in the bottom configuration. (cq to determine if such a difference does exist.)

6.02 Parameters of the wave field

A number of restrictions and assumptions underlie the mathematical models (2.07.1) and (2.07.2).

Figure 2.00.1, repeated here for convenience, shows the characteristic quantities and parameters which are important in wave propagation problems. (Berkhoff, 1976).



(6.02.1)

where: D : mean water depth

H : wave height

$\lambda_o = g \div \omega^2$

σ : mean slope of the bottom over a distance D.

For a linear model it is assumed that the wave steepness $\gamma = H \div \lambda_o$ is small.

In the case of a small waterdepth the ratio $\alpha = H \div D$ must also be small.

In the refraction - diffraction equation it is assumed that the variation of the bottom is moderately small and noticeable for the surface waves $(\frac{\sigma\lambda_o}{D} \ll 1)$.

The curvature of the wave amplitude is taken into account.

A two dimensional equation in the horizontal is obtained by integration over the waterdepth.

(See Appendix 2.)

These assumptions do not have to be made for the Laplace model.

6.03 Selected bottom configuration

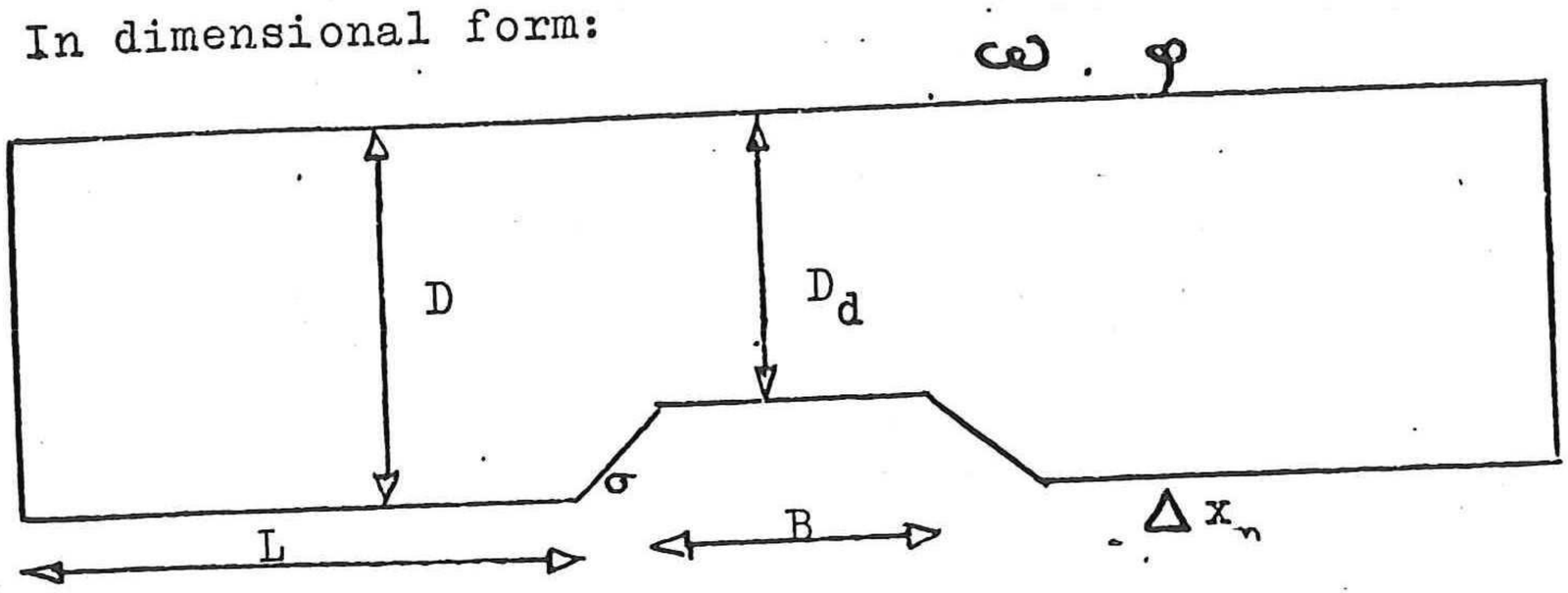
A trapezoidal dam has been selected because of the simplicity of it's contour and the clearly defineable bottom slope.

The selected setup is depicted in figure (6.03.1).

This figure shows the characteristic quantities and parameters which are important for the comparison of the Side View Model versus the Top View Model.

(Compare with (6.02.1))

In dimensional form:



(6.03.1)

where: ω : wave time frequency
λ_o : wave length at deep water
H : wave height
D : waterdepth
D_d : waterdepth over the dam
σ : slope of the incline of the dam
B : width of the crest of the dam
L : length of the zone left of the dam
φ : wave potential at the surface: $\varphi = \varphi_1 + i\varphi_2$
g : gravitational potential
Δx_n : numerical steplength of the computation

It is more convenient to use dimensionless parameters. The coordinates are made dimensionless using the free surface characteristic length $\lambda_o = g \div \omega^2$ (deep water wave length)

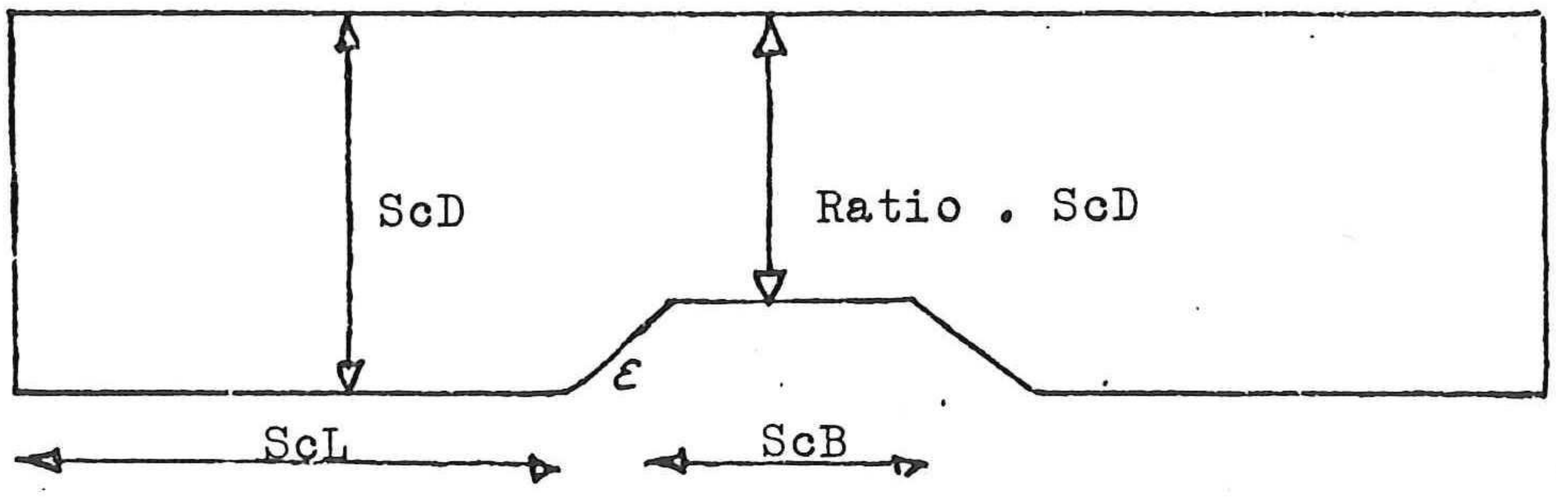
As Berkhoff (1976) has done, the gradient of the water-depth can be made of the order one by taking new horizontal coordinates (x', y') in the (dimensionless) waterdepth function $\frac{D}{\lambda_o}\langle x, y\rangle$:

$$\varepsilon \langle x'.y' \rangle = \frac{\sigma}{D/\lambda_o} \langle x.y \rangle$$

As a consequence of this the bottom slope is characterised by the parameter (see figure (6.03.3)) :

$$\varepsilon = \frac{\sigma \lambda_o}{D} \qquad (6.03.2)$$

In dimensionless form:



(6.03.3)

where: $ScD = D + \lambda_o$

$ratio = D_d + D$

$ScB = B + \lambda_o$

$ScL = L + \lambda_o$

$Sc\Delta x_n = \Delta x_n + \lambda_o$

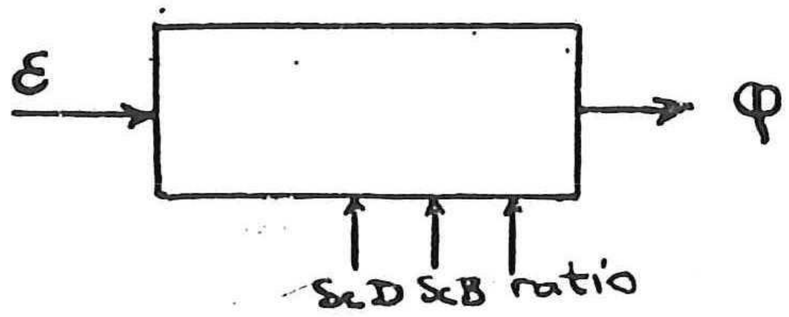
$\varepsilon = \sigma + (D + \lambda_o)$

Verder lineair model $\Rightarrow$ H met Lokamp?

From the derivation of the refraction - diffraction equation (Berkhoff, 1976) it may be expected that this parameter will be important for differences in predictions, by the two models, of the values of the wave potential.

Because of the importance of the assumption (in the derivation by Berkhoff) of the bottom variation being moderately small (see (6.03.3) and Appendix 3.), the computer program and the experimentation have been made to focus on the variation of ε.

For that reason ε has been varied in a number of cases where, per case, the other parameters remained constant, as depicted in the following system sketch:



(6.03.3)

System sketch. The computer program as a system for predicting values of φ for varying values of ε with a chosen setting of values of ratio, ScD, ScB.

6.04 Discussion of the parameters

In this section the parameters ScD, Ratio and ScB will be discussed.

Parameter ε has already been described in (6.03.2), and φ will be described in the next section.

The influence of parameters ΔL (length of the zone upwave of the dam) and of Δx_o (steplength) is described in Appendix 6 . These parameters do not have to be varied in the computational model once they have been suitably determined.

ScD : Dimensionless measure of the depth of the wavefield.

Values of ScD greater than 1.0 will not be useful: the effect of the bottom will not be noticeable.

For this reason four values have been selected:

ScD = 0.6 ; 0.5 ; 0.4 ; 0.25 .

Ratio : Dimensionless measure of the waterheight over the dam.

The value of Ratio is to be greater than zero, in order to be able to use the mesh generator. (See section 7.01).

In this study no troughs are considered, that is:
Ratio is not greater than 1.0 .

In this study values for Ratio have been selected in the interval (0.1 → 1.0).

ScB : Dimensionless measure of the width of the crest of the dam.

The value of ScB may in principle vary between zero and infinite. However, because of the requirement that the influx and efflux boundaries be sufficiently far removed from the barrier, there exists a maximum value of ScB which is determined by, amongst others, the length of the basin. (The length of the basin is restricted by the maximum number of nodes of the computation, i.e., the costs of the computation.)

This restriction has restricted the values of ScB in this study to a maximum of ScB ≙ 0.6 .

By letting the crest of the dam go to the downwave side (efflux) boundary, an additional value of ScB = ∞ is obtained.

6.05 Characteristic parameter of the wavefield

The computational model determines the value of the wave potential, φ, at the surface for the various combinations of the parameters ε, ScD, Ratio and ScB which determine the bottom profile.

The wave potential, φ, consists of three components:

φ_i : the incident wave

φ_r : the reflected wave

φ_t : the transmitted wave

The comparison of the predicted wave potentials of the Side View Model and the Top View Model will be done using the magnitudes of the potential: $(\|\varphi\| = (\varphi_1^2 + \varphi_2^2)^{\frac{1}{2}})$.

$\|\varphi_i\|$: magnitude if the incident wave

$\|\varphi_r\|$: magnitude of the reflected wave

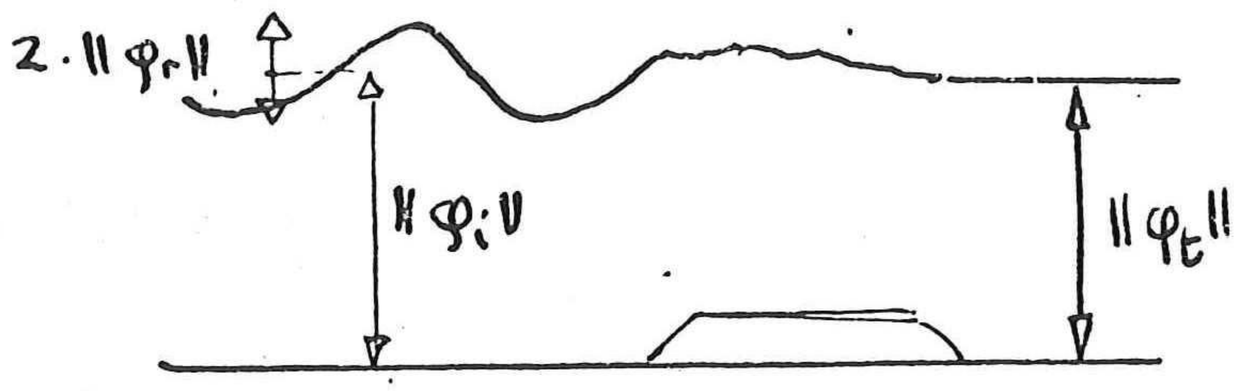
$\|\varphi_t\|$: magnitude of the transmitted wave

There is a fixed relationship between these three components (Lamb, 1975):

$$\|\varphi_i\|^2 = \|\varphi_r\|^2 + \|\varphi_t\|^2 \qquad (6.05.1)$$

With the aid of figure (6.05.2) it will be shown how, from a graph depicting the values of $\|\varphi\|$, the three

characteristic components may be determined.



(6.05.2)

where: $\|\varphi_i\| = \frac{1}{2}(\|\varphi\|_{max} + \|\varphi\|_{min})$

$\|\varphi_r\| = \frac{1}{2}(\|\varphi\|_{max} - \|\varphi\|_{min})$

In this study the value of the imposed $\|\varphi_i\|$ is equal to one.

The oscillations of $\|\varphi\|$ left of the barrier are caused by the interference of φ_i and φ_r ; the amplitude of the oscillations is the amplitude of the reflected component.

The average value of the oscillation is the value of the potential of the incoming wave.

The height of the potential downwave of the barrier is the magnitude of the potential of the transmitted wave.

Due to the chosen magnitude of the imposed wave, $\|\varphi_i\| = 1$, and because of the existence of the relationship between φ_i, φ_r and φ_t as described by equation (6.05.1), the characterisation of the wavefield may be done using either $\|\varphi_r\|$ or $\|\varphi_t\|$ as the characteristic parameter of the wave field.

Because of the greater ease of determining $\|\varphi_t\|$ (see figure 6.05.2), $\|\varphi_t\|$ has been selected as the characteristic parameter of the wavefield in the comparison of the two models.

Thus: the comparison of the Side View Model versus the Top View Model will be done by comparing the predicted values of $\|\varphi_t\|$ of the two models for various bottom configurations.

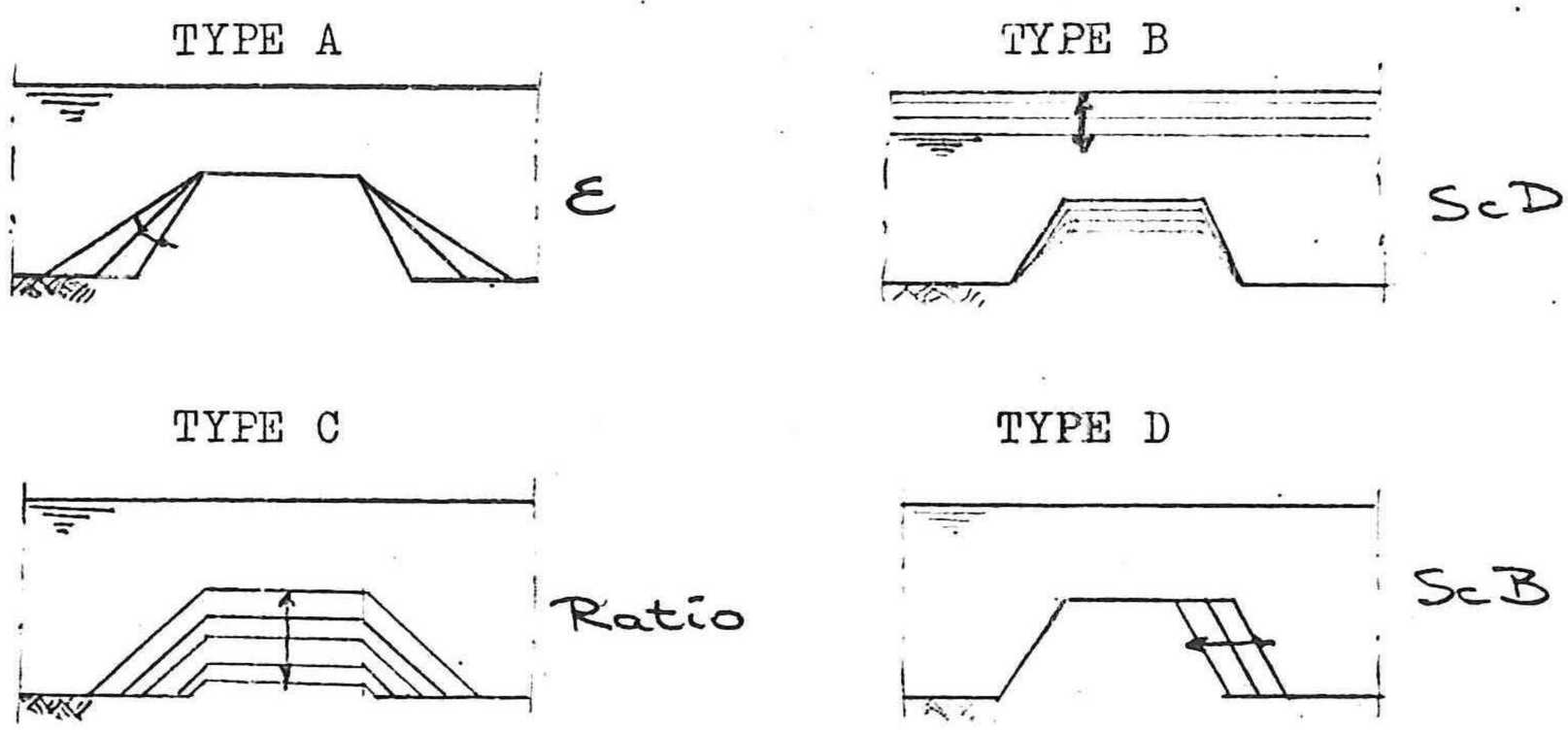
The predictions will be compared in relative form:

$$\frac{\|\varphi_t\|_{T.V.M.} - \|\varphi_t\|_{S.V.M.}}{\|\varphi_t\|_{S.V.M.}} = \frac{\|\varphi_t\|_T}{\|\varphi_t\|_S} - 1 \qquad (6.05.3)$$

: The results of the Top View Model (refraction - diffraction equation) are compared versus those of the Side View Model ('laplace equation') which is taken as reference.

6.06 Three main types of variations

From the descriptions of sections (6.03) and (6.04) it follows that there are four basic types of variations, associated with the parameters ε, ScD, Ratio and ScB :

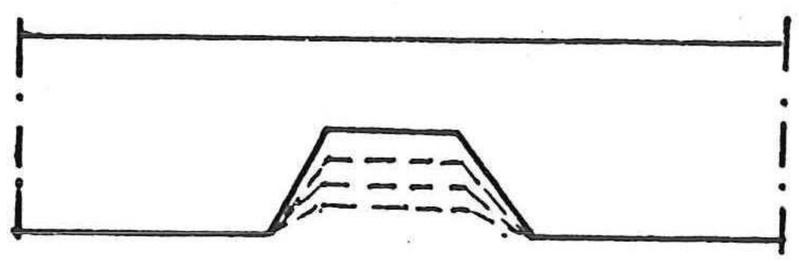


For the computations these four basic types have been combined into one type which allows for the variations of the four parameters: ITYPE 2 .

In order to allow for the case of ScB = ∞ (as described insection (6.04)) ITYPE 3 has been designed.
ITYPE 1 has been designed to provide direct insight into the effects of the variation of the damheight.

In each ITYPE the value of ε is varied for a chosen setting of the values of ScD, Ratio, ScB.

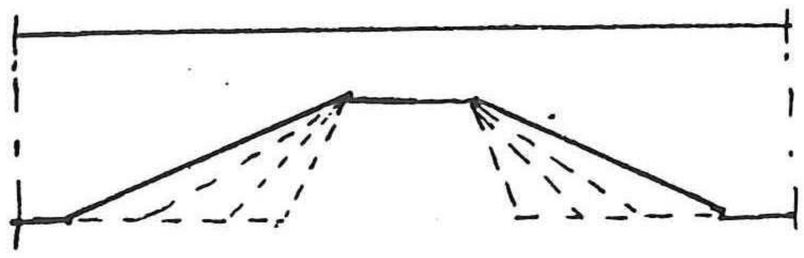
The three ITYPEs are:



ITYPE 1

The height of a symmetrical trapezoidal dam is varied in steps of 0.1*ScD.

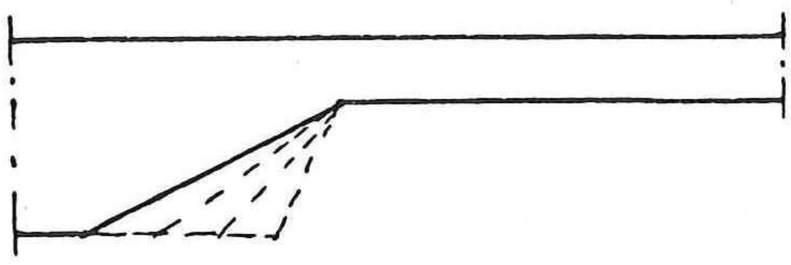
(6.06.1



ITYPE 2

The slope of the incline of the dam (of constant height) is varied by changing the length of the toe of the dam.

(6.06.2)



ITYPE 3

Identical to ITYPE 2 but for the infinite crestwidth.

(6.06.3)

6.07 Selected parameter combinations

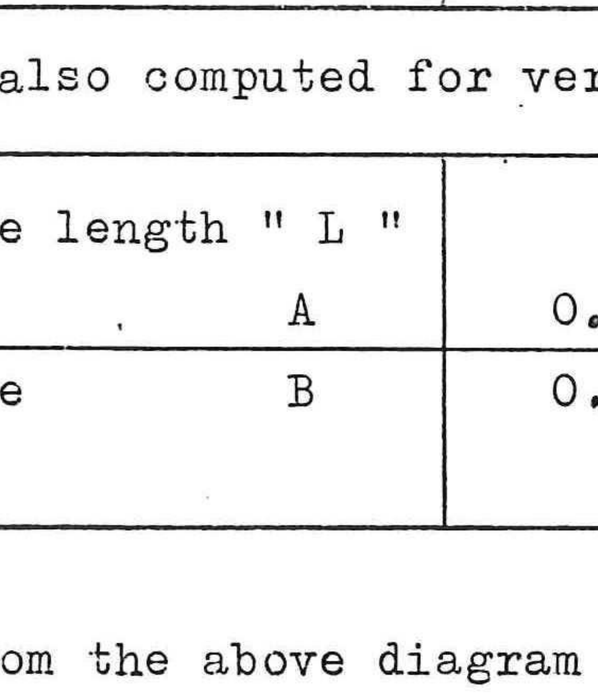
Diagram (6.07.1) presents the values of the parameters which have been used for the computations, for the three different ITYPEs.

(The description of the ITYPEs can be found in Appendix 4, and section 6.06.)

The results of the computations for these three ITYPEs are presented in the next chapter.

Diagram 6.07.1 Tested configurations				
ITYPE 1	CASE nr.	ScD	Ratio	ScB
	1	0.4	varies	0.6
	2	0.6	varies	0.6
ITYPE 2	3	0.6	1÷3	0.6
	4	0.5	1÷3	0.6
	5	0.4	1÷3	0.6
	6	0.25	1÷3	0.6
	7	0.6	1÷5	0.6
	8	0.6	1÷1.5	0.6
	9	0.6	1÷2	0.6
	10	0.6	1÷3	0.4
	11	0.6	1÷3	0.2
ITYPE 3	12	0.6	1÷3	∞
two cases are also computed for verification				
extended upwave length " L "	A	0.6	1÷3	0.6
double stepsize	B	0.6	1÷3	0.6

It can be seen from the above diagram that with respect to CASE 3 the influence of parameter " ScD " is investigated in cases 4 through 6; the influence of " Ratio " in cases 7 through 9 and the influence of " ScB " in cases 10 through 12:



Investigated variations of the parameters relative to CASE 3.

The variations which are already implied in the ITYPEs are the following: (see section 6.06) :

ITYPE 1 varies the height of the dam in steps of a tenth of the waterdepth.
The only required extra information is the length of the toe of the dam: " toe ".
(In this study Toe is kept constant: Toe = 0.2 in ITYPE 1.)

ITYPE 2 varies the length of the toe of the dam for a fixed damheight. (That is: the slope σ is made to vary by changing the run, keeping the rise constant.)

ITYPE 3 is identical to ITYPE 2 except for the crestwidth.
(See page - 67 - .)

Two cases are computed for the purpose of verification;

CASE A — This case is identical to CASE 3 except for the extension of the length "SCL" upwave of the dam (see figure (6.03.3)). This case acts as a test for the location of the influx boundary, which should be sufficiently far removed from the barrier.

CASE B — In order to test the stepsize the most critical case of CASE 3 is also calculated with double stepsize.

(Doubling stepsize means less nodes, thus a cheaper computation. If the results are acceptable this does provide sufficient information. See the discussion on this topic in Appendix 6.)

7 RESULTS

7.00 Introduction

In this chapter the results are presented of the comparative computation.

(In section (6.07) it has been shown which cases have been selected for the three ITYPEs.)

Typical examples are given of a mesh generated by the mesh generator and of the results as presented by the computer program.

A description is given of the manner in which the computer presented data are interpreted to provide the data required for the comparison.

The predicted values of the wave potentials are presented.

7.01 The generated Mesh

One of the results which the computer program produces is the mesh.

This mesh plays an essential role in the computation because by the mesh the location of the nodes where the computation is done is determined.
Thus the mesh also shows the steplength with which the computation is done.

The mesh is generated by AFEP's meshgenerator MESB2D, based on user provided characteristic nodes (vertices of the field, etc.).
In this study the mesh generator is made to generate a mesh which has a refinement in the region where it is expected that the potential changes most: over the barrier.

Figures 7.01.1 and 7.01.2 show, at different scales, typical examples of generated meshes for the Top View Model resp. the Side View Model.

It is possible to let the computer display the details of the generated mesh such as the sections of which the total mesh is composed and the nodal numbering.
An example of the sections of the mesh is shown in figure 7.01.3 .

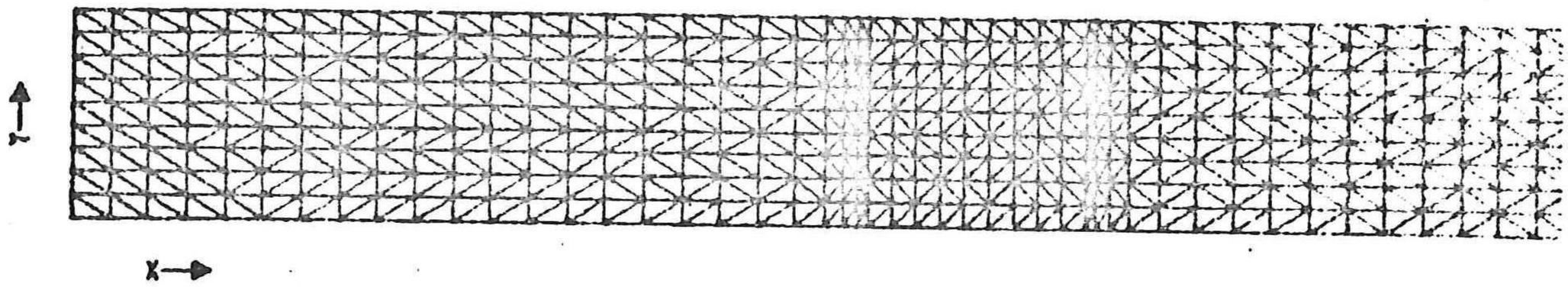


Figure 7.01.1 Example of a mesh generated in the Top View Model.

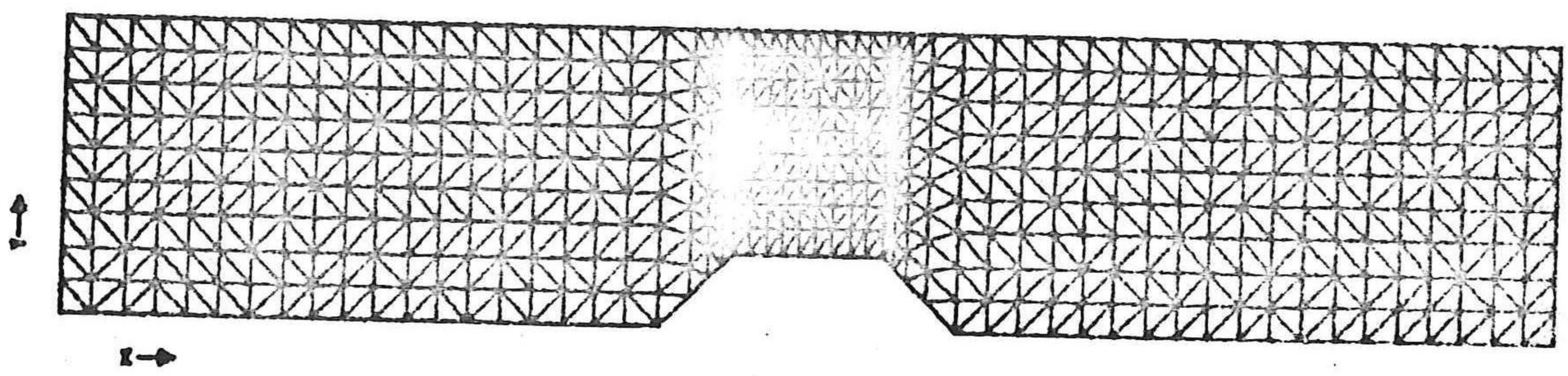


Figure 7.01.2 Example of a mesh generated in the Side View Model.

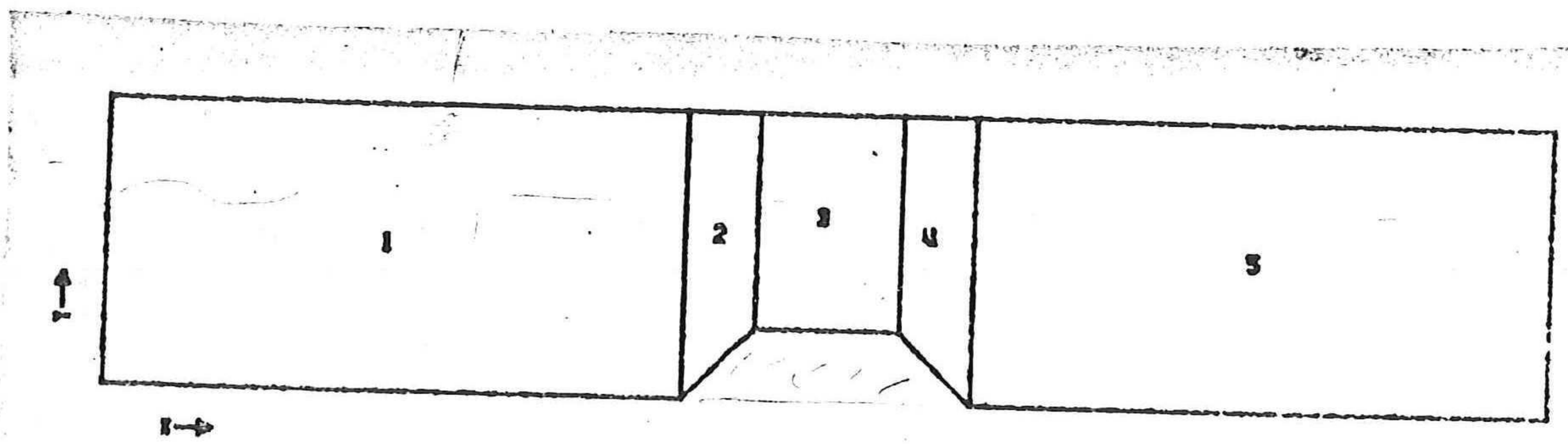


Figure 7.01.3 Example of the division of the field into subfields as done by the mesh generator of the Side View Model.

7.02 Form of presentation of the results

AFEP permits a number of forms of presentation, such as the drawing of contourlines of the potential (see for an example figure (7.02.2)) or a graph of the values of the potential along a selected interval. Both are made on the plotter.

In this study an alternative method has been used: the values of the potential along the surface boundary have been printed (conform the remarks made in section 2.07) and presented in the form of a graph of $\|\varphi\|$. In this way the results became more readily available and at smaller costs (as compared to the rather expensive plotter) .

An example of the presentation of the results can be found in figure (7.02.3).

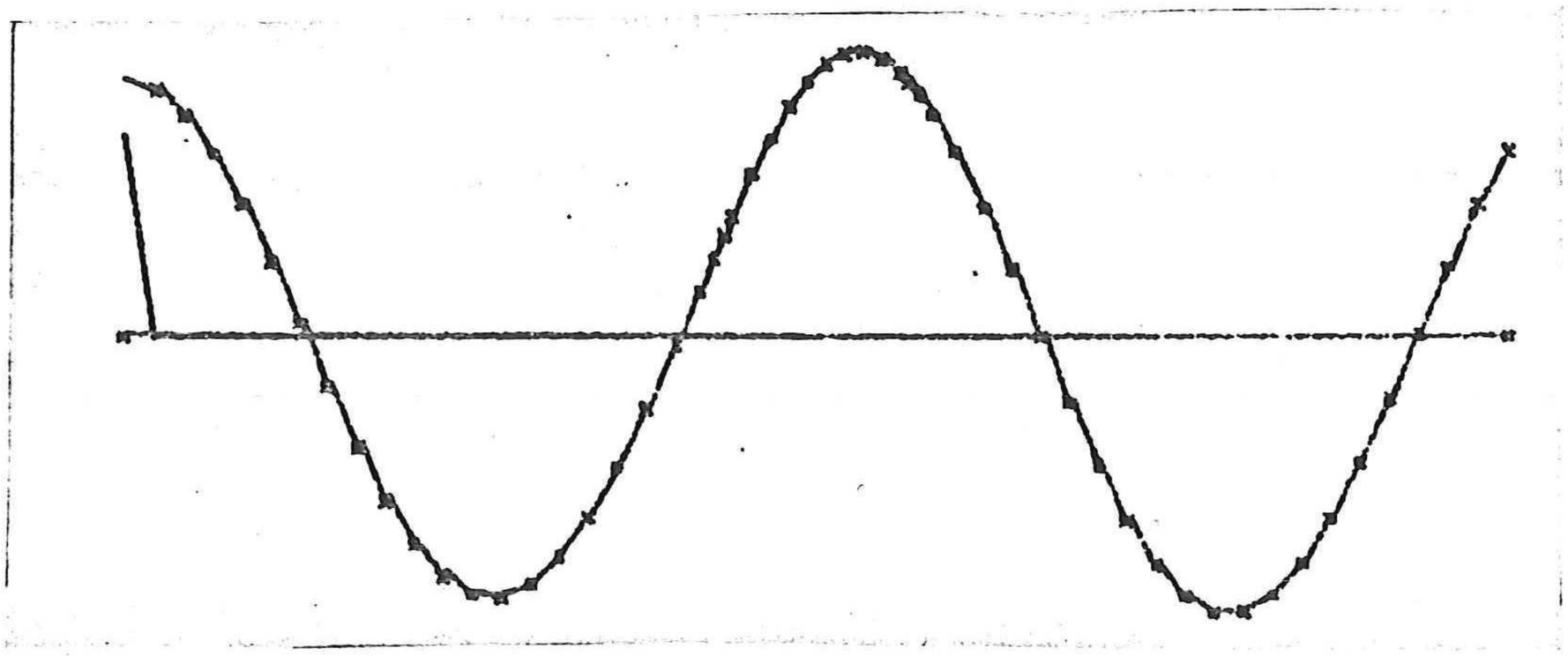


Figure 7.02.1 Example of the plotted values of the potential at the water surface (in this case: for a flat horizontal bottom.)

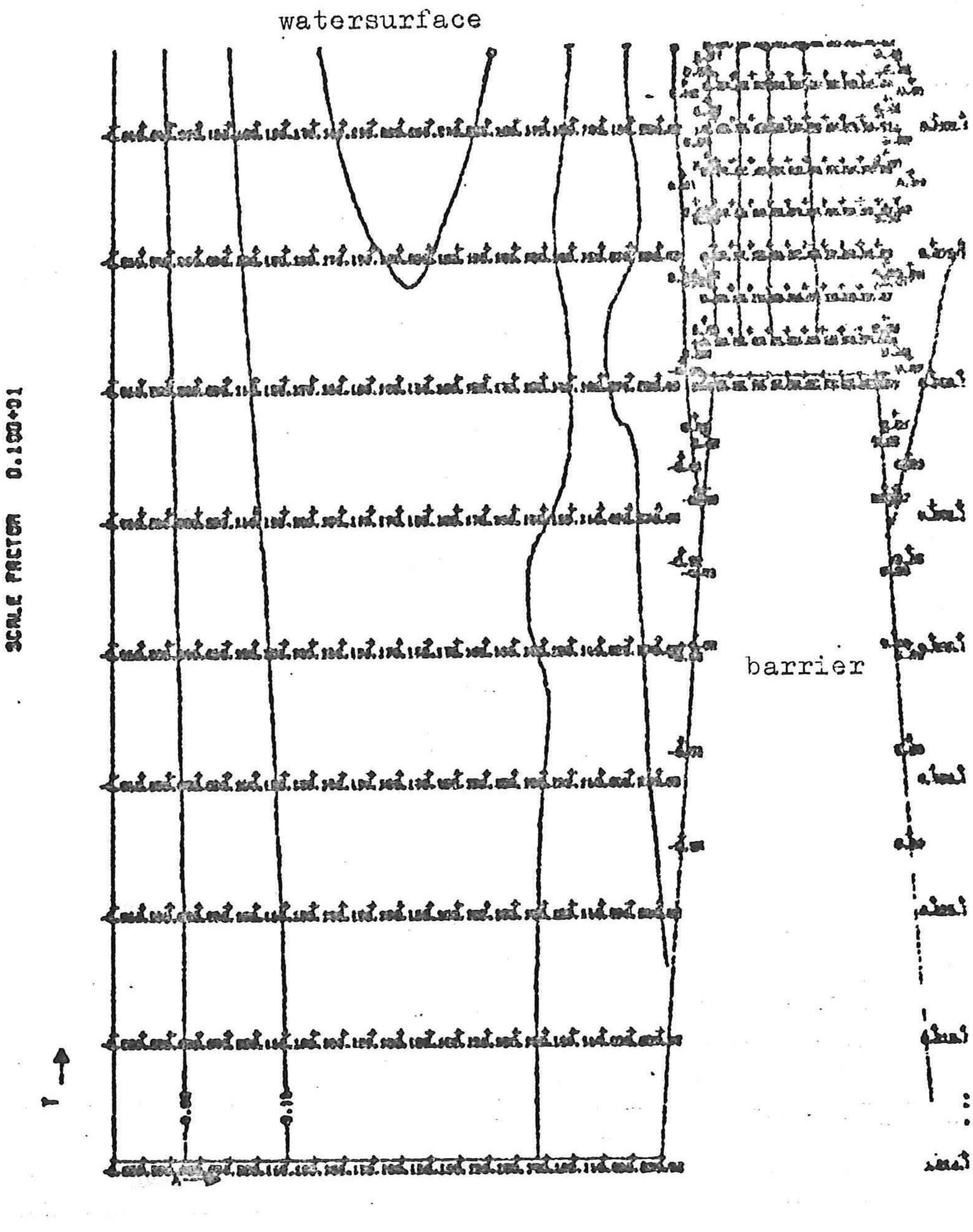


Figure 7.02.2 An example of a contourplot of the values of the potential (in this case: of a section of the field of the Side View Model with a barrier height which is 0.6 times the waterdepth.)

Presentation in compressed scale.

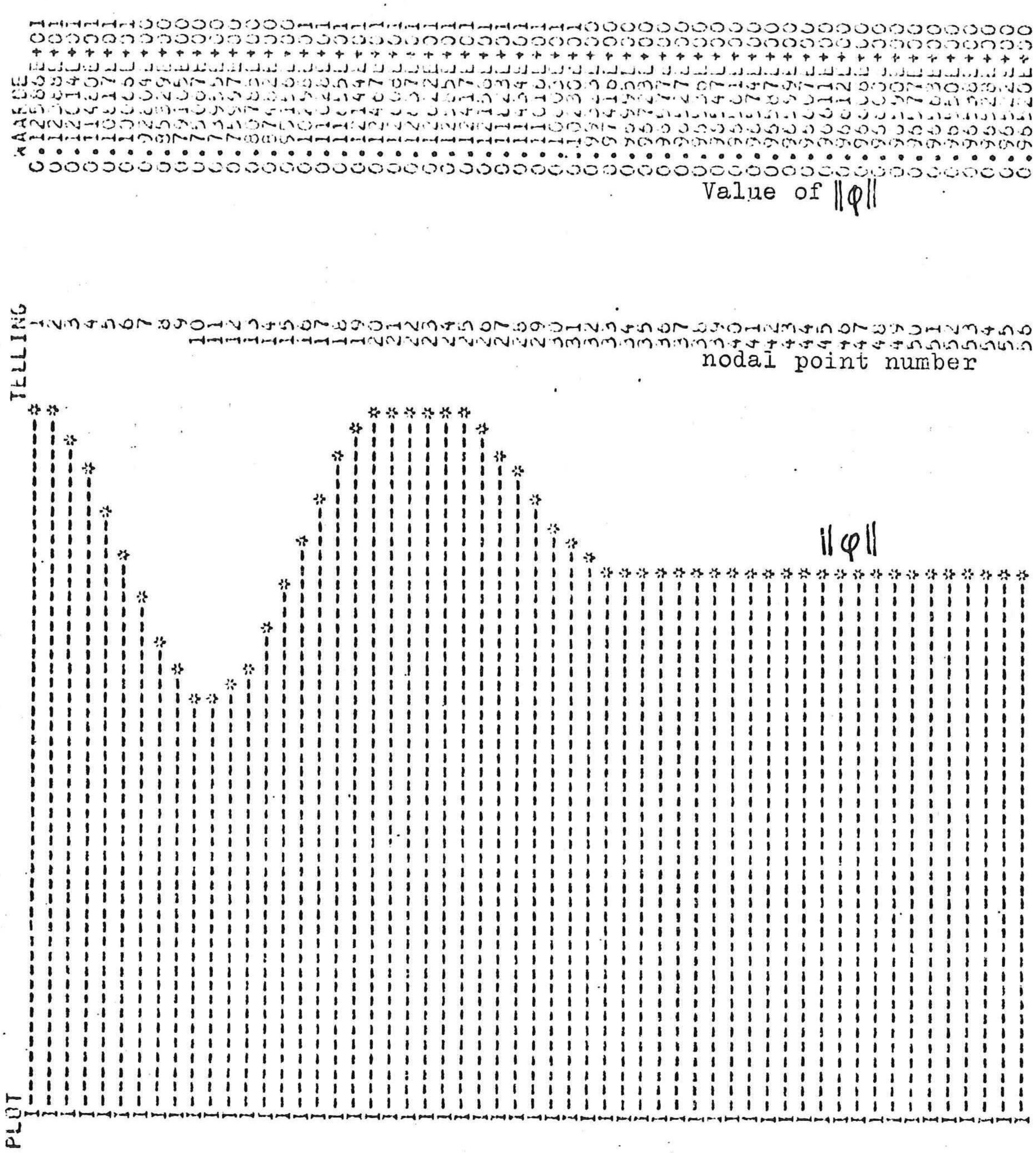


Figure 7.02.3 Example of the form of presentation used for this study.

Presentation of the values of $\|\varphi\|$ at the nodes along the surface and a representation of these values in a graph.

For a description of the meaning of the presented characteristic parameter in this graph consult section 6.05 .

7.03 Gathering data from the graphs

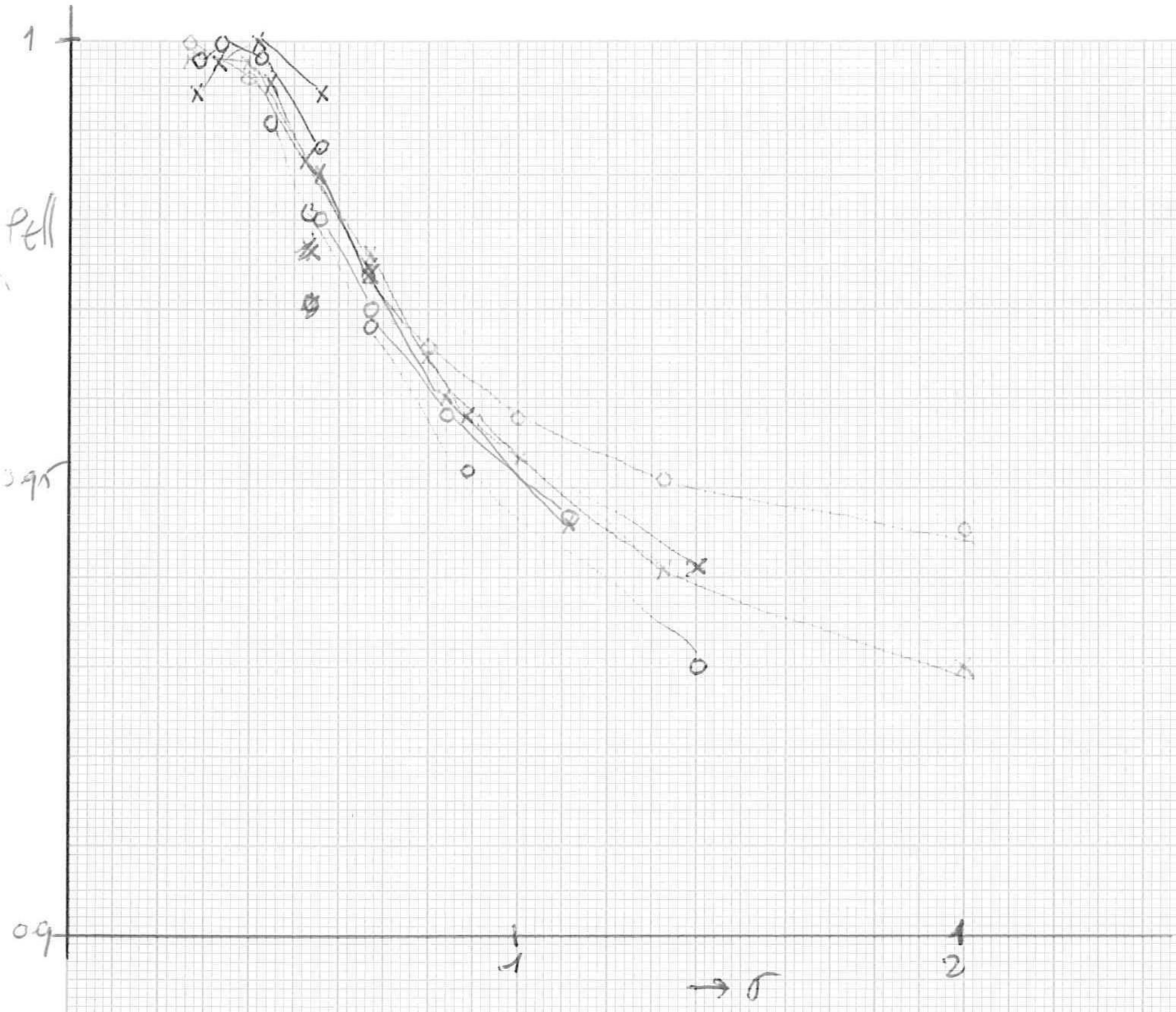
The results of the computations of the Side View Model and the Top View Model all are presented in graphs of the form presented in figure (7.02.3).

These graphs are interpreted as described in section 6.05 .

7.04 Results in tables

The following are the results of the computations for the cases described in section 6.07, interpreting the data as described in section 6.05 .

The numbering of the cases is the same as in section 6.07 .



ScD = 0.4 0.5 0.6 0.25

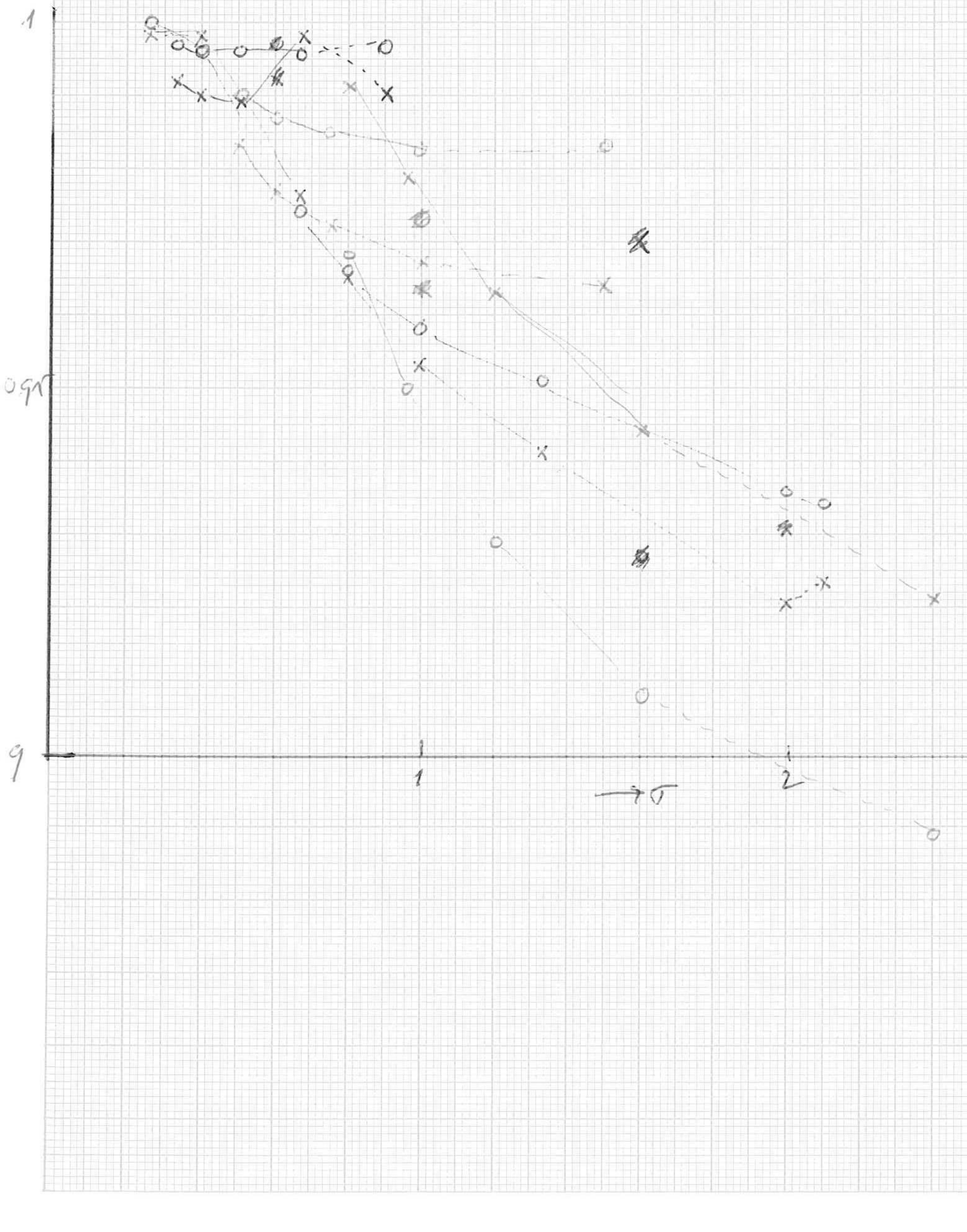
ratio = $\frac{1}{3}$

ScB = 0.6

x Fidu

o Fp

Undried ScD

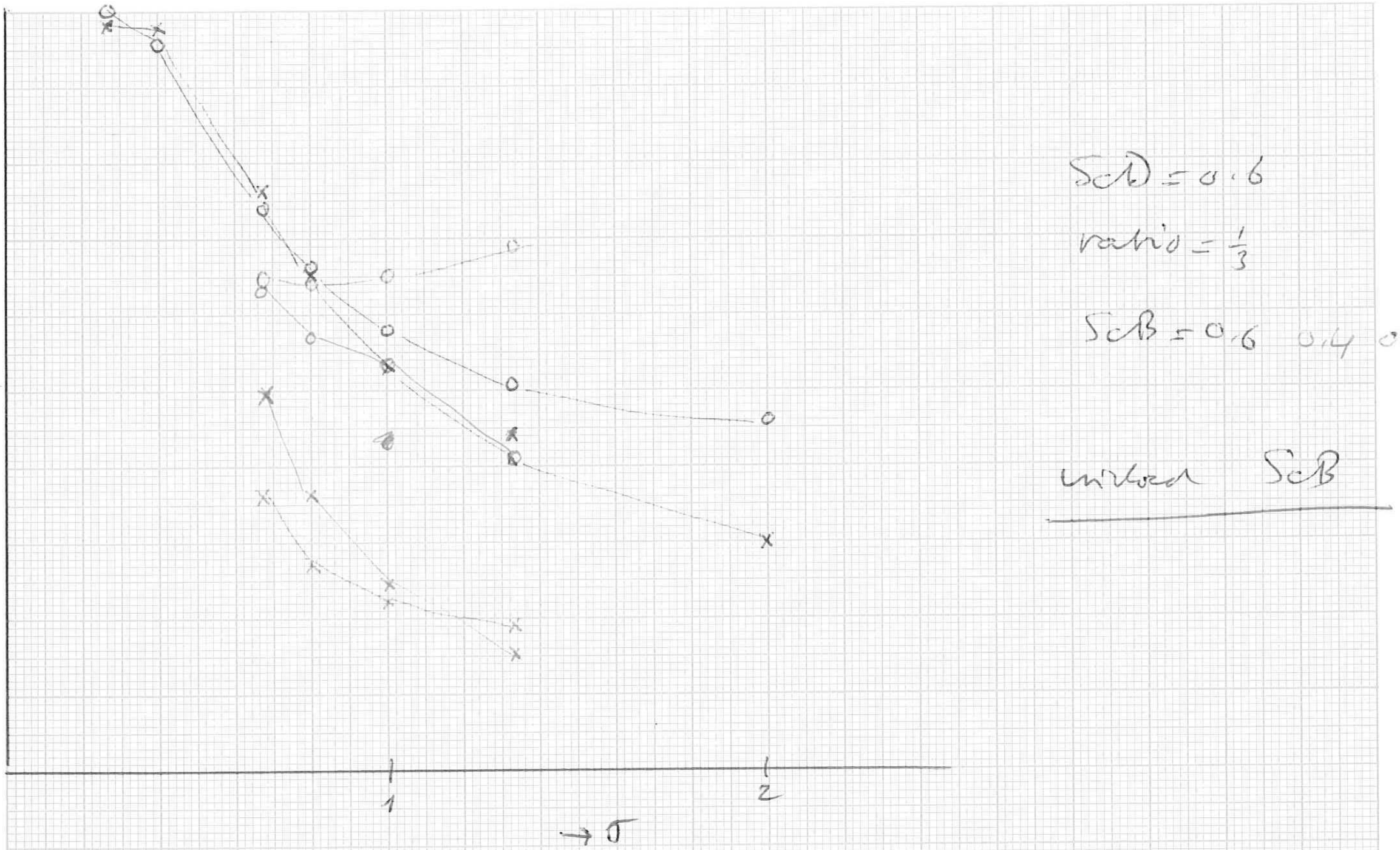


ScD = 0.6

ScB = 0.6

ratio = $\frac{1}{3}$ $\frac{1}{5}$ $\frac{1}{1.5}$ $\frac{1}{1}$

Undried "ratio"



SCD = 0.6

ratio = 1/3

SCB = 0.6 0.4 0.2

unfixed SCB

Table 7.04.1 The results of the computations, per CASE

CASE 1	ScD	Ratio	ScB	ΔD	Δx	sigma	epsilon	$\|\varphi_t\|_s$	$\|\varphi_t\|_T$	$\frac{\|\varphi_t\|_T}{\|\varphi_t\|_s} - 1$
	0.4	1.0	0.6	0.00	0.2	0.00	0.00	1.000	1.000	0.000
		0.9		0.04		0.20	3.14	.999	1.000	.001
		0.8		0.08		0.40	6.28	.995	.998	.003
		0.7		0.12		0.60	9.42	.988	.994	.006
		0.6		0.16		0.80	12.57	.978	.986	.008
		0.5		0.20		1.00	15.71	.963	.973	.010
		0.4		0.24		1.20	18.85	.948	.954	.006
		0.3		0.28		1.40	21.99	.941	.930	- .012
		0.2		0.32		1.60	25.13	.970	.927	- .044
		0.1		0.36		1.80	28.27	.883	.981	.111
CASE 2										
	0.6	1.0	0.6	0.00	0.2	0.00	0.00	1.000	1.000	0.000
		0.9		0.06		0.30	3.14	.999	1.000	.001
		0.8		0.12		0.60	6.28	.996	.999	.003
		0.7		0.18		0.90	9.42	.990	.997	.007
		0.6		0.24		1.20	12.57	.979	.992	.013
		0.5		0.30		1.50	15.71	.964	.983	.020
		0.4		0.36		1.80	18.85	.944	.964	.022
		0.3		0.42		2.10	21.99	.923	.934	.012
		0.2		0.48		2.40	25.13	.921	.889	- .035
		0.1		0.54		2.70	28.27	1.000	.929	- .071
CASE 3										
	0.6	1÷3	0.6	0.4	1.5	0.26	2.79	.998	1.000	0.002
					1.0	.40	4.19	.998	.996	- .002
					.6	.67	6.98	.976	.974	- .002
					.5	.80	8.38	.965	.966	.001
					.4	1.00	10.47	.953	.958	.005
					.3	1.33	13.96	.941	.951	.011
					.2	2.00	20.94	.930	.946	.017
CASE 4										
	0.5	1÷3	0.6	0.33	0.6	0.556	6.98	.985	.980	- .005
					0.5	.667	8.38	.974	.970	- .004
					0.4	.833	10.47	.960	.958	- .002
					0.3	1.111	13.96	.946	.947	.001
CASE 5										
	0.4	1÷3	0.6	0.266	0.6	.444	6.98	.995	.991	- .004
					0.5	.533	8.38	.987	.981	- .006
					0.4	.667	10.47	.974	.968	- .006
					0.3	.889	13.96	.958	.952	- .006
CASE 6										
	0.25	1÷3	0.6	0.16	0.6	0.278	6.98	.994	.998	.004
					0.5	0.333	8.38	.998	1.000	.002
					0.4	0.417	10.47	1.000	.998	- .002
					0.3	0.556	13.96	.994	.988	- .006

Table 7.04.1 The results of the computations, per CASE, continued

CASE	ScD	Ratio	ScB	ΔD	Δx	sigma	epsilon	$\|\varphi_t\|_S$	$\|\varphi_t\|_T$	$\frac{\|\varphi_t\|_S}{\|\varphi_t\|_T} - 1$
	0.6	1÷5	0.6	0.48	0.6	0.80	8.38	.991	.968	- .023
					0.5	0.96	10.05	.979	.950	- .030
					0.4	1.20	12.57	.963	.929	- .035
					0.3	1.60	16.76	.944	.908	- .038
CASE 8										
	0.6	1÷1,5	0.6	0.2	0.6	0.33	3.49	.992	.997	.005
					0.5	0.40	4.19	.990	.996	.006
					0.4	0.50	5.24	.989	.996	.007
					0.3	0.67	6.98	.998	.996	.008
CASE 9										
	0.6	1÷2	0.6	0.30	0.6	.50	5.24	.983	.990	.007
					0.5	.60	6.28	.977	.987	.010
					0.4	.75	7.85	.972	.985	.013
					0.3	1.00	10.47	.967	.983	.017
CASE 10										
	0.6	1÷3	0.4	0.4	0.6	0.667	6.98	.949	.963	.015
					0.5	0.80	8.38	.936	.957	.022
					0.4	1.00	10.47	.924	.953	.031
					0.3	1.333	13.96	.915	.951	.039
CASE 11										
	0.6	1÷3	0.2	0.4	0.6	0.667	6.98	.936	.965	.031
					0.5	0.80	8.38	.927	.964	.040
					0.4	1.00	10.47	.922	.965	.047
					0.3	1.333	13.96	.919	.969	.054
CASE 12										
	0.6	1÷3	'∞'	0.4	0.6	0.667	6.98	1.168	1.176	.007
					0.5	0.80	8.38	1.166	1.175	.008
					0.4	1.00	10.47	1.163	1.174	.009
					0.3	1.333	13.96	1.160	1.172	.010
CASE A										
	0.6	1÷3	0.6	0.4	0.3	1.333	extended upwave fieldlength .939			
CASE B										
	0.6	1÷3	0.6	0.4	0.3	1.333	double stepsize .934			

Discussion of these testcases is done in the next chapter.

8 Interpretation of the results

8.00 Introduction

The results of table (7.04.1) are interpreted in this chapter.

The predicted values of $\|\varphi_t\|$ are displayed in graphs for both the Side View Model and the Top View Model.

The predictions of the two models, compared using the parameter $\frac{\|\varphi_t\|_T}{\|\varphi_t\|_S} - 1$, are also displayed in graphs.

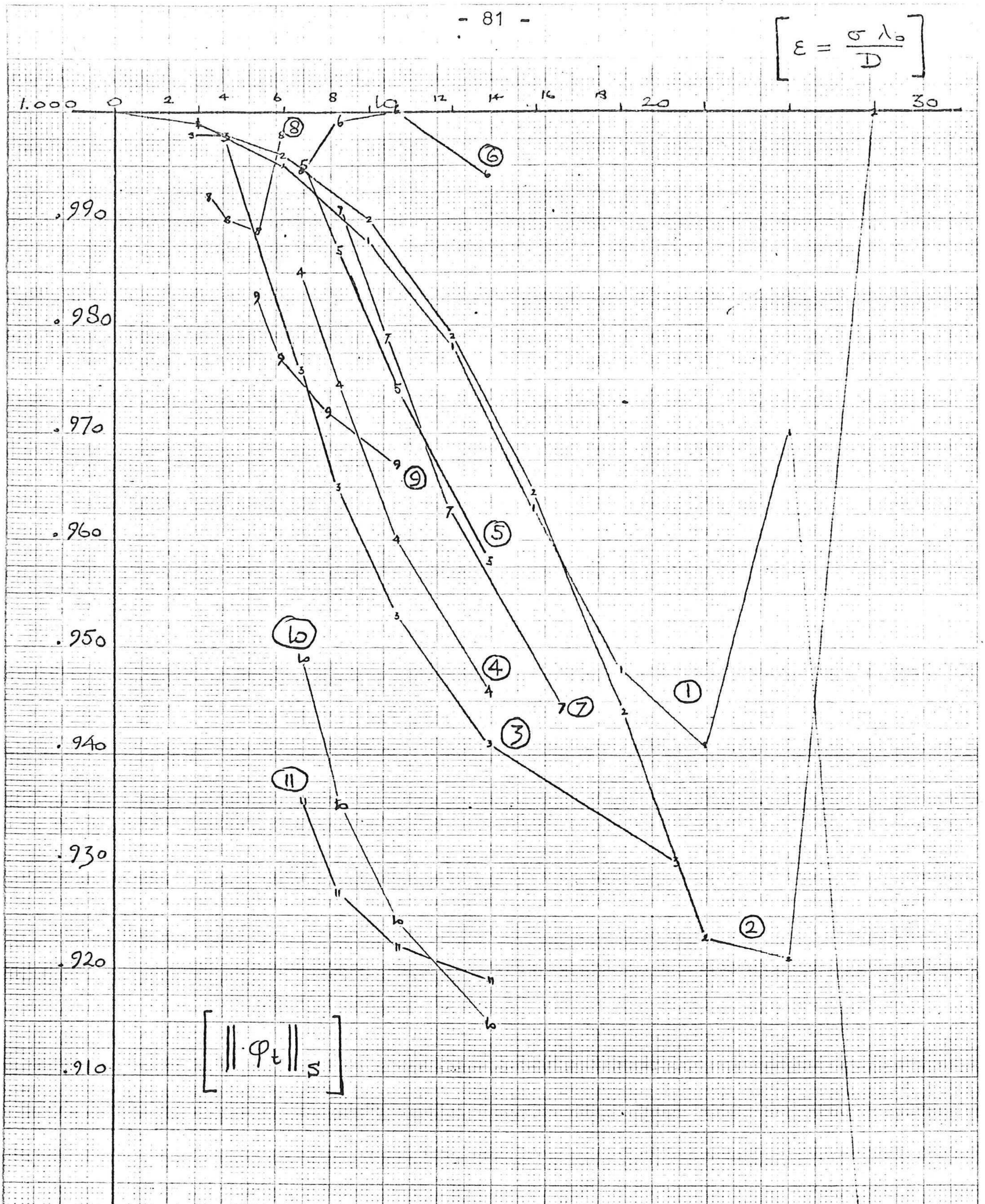
The effect of the variation of the parameters ScD, Ratio and ScB is investigated.

This leads to a discussion of the results in the next chapter.

8.01 Presentation per model and per CASE

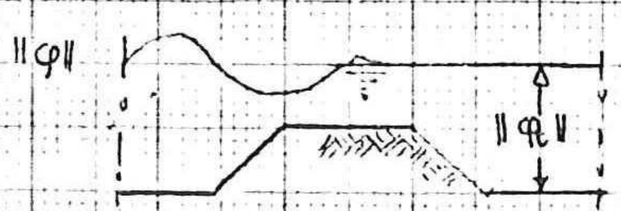
The predictions of $\|\varphi_t\|$ have been presented in graph (8.01.1) for the Side View Model and in graph (8.01.2) for the Top View Model.

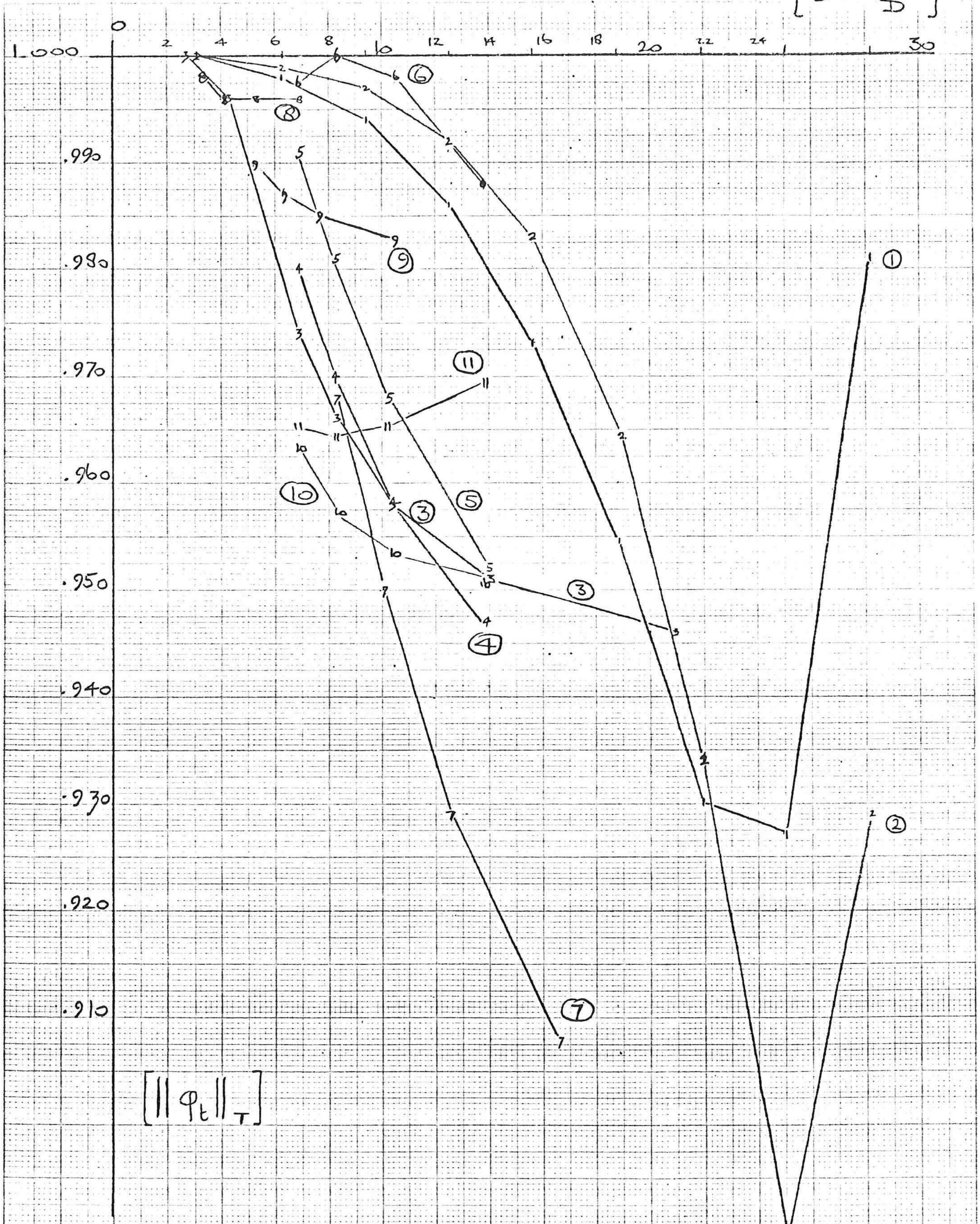
The results have also been grouped per CASE, as displayed in graph (8.01.3). For the ease of the comparison the curves have been untangled, providing better insight in the relative predictions per case.



Graph 8.01.1

Predicted values , by the Side View Model, of the potential of the transmitted wave for the cases 1 through 11.(see diagram 6.07.1)





Graph 8.01.2

Predicted values, by the Top View Model, of the potential of the transmitted wave.

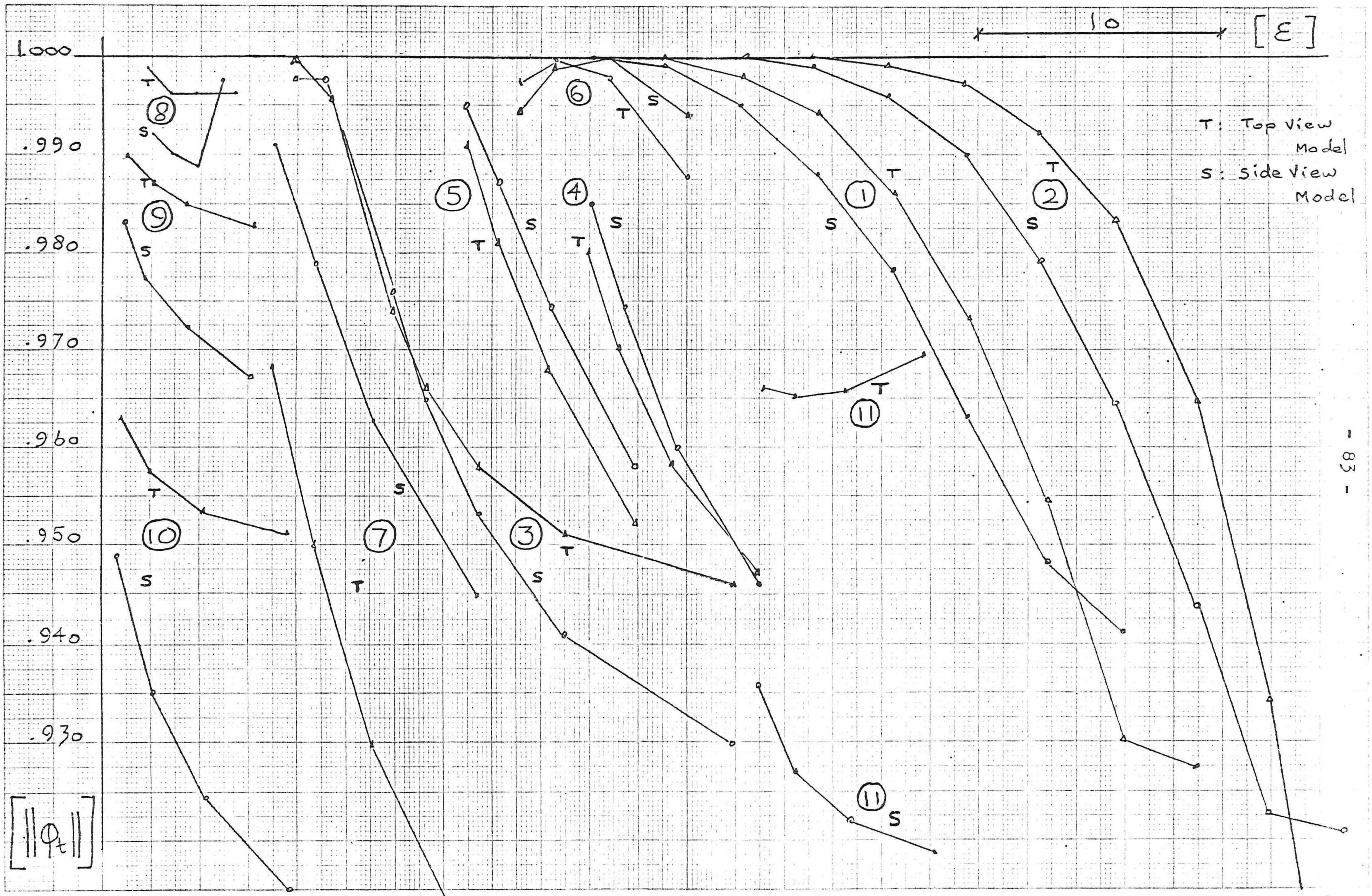


Figure 8.01.3 Comparative presentation of predictions per case.

8.02 Evaluation of the predictions of $\|\varphi_t\|$

From the data in graphs (8.01.1) through (8.01.3) insight might be obtained concerning the propagation of a wave over a trapezoidal dam. As can be seen in graph (8.01.3) remarkable differences exist between the predictions by the two models. Since relatively little information is available of the characteristics of the propagation of waves over a trapezoidal dam it requires a special study, outside of the scope of this study, to further evaluate this information.

Only for CASE 1 and 2 it may be concluded that the behaviour which is predicted by the computation is in accordance with the behaviour described in other studies. Mobarek (1974) has done hydraulic scale model experiments for the wave propagation over a trapezoidal dam and found that the coefficient of transmission varies with the parameter Ratio as depicted in figure (8.02.1).

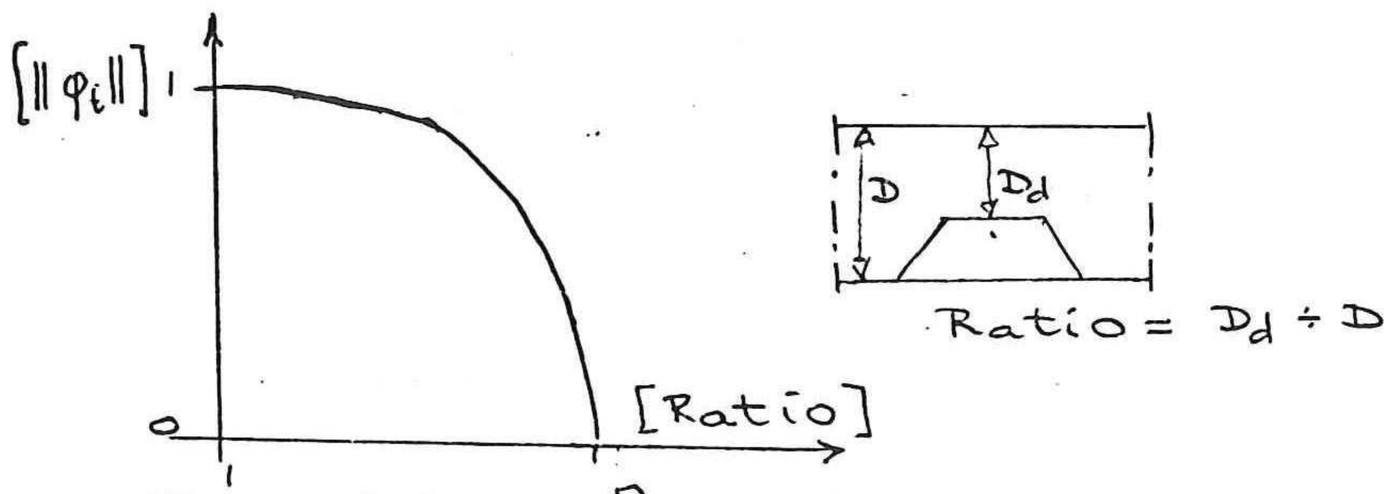


Figure 8.02.1 Schematic representation of the characteristic behaviour of transmission over a trapezoidal dam.

As can be seen in graph (8.01.3) this is indeed the form of the computed coefficients of transmission. Only for low values of Ratio do the predictions deviate from the

expected behaviour. (D_d less than 0.4 D, approximately) According to the data of Mobarek nonlinear behaviour will have to be taken into account for these values of Ratio, which means that the computer model could not be used for these cases.

8.03 Evaluation of the values of $\frac{\|\varphi_t\|_T}{\|\varphi_t\|_S} - 1$.

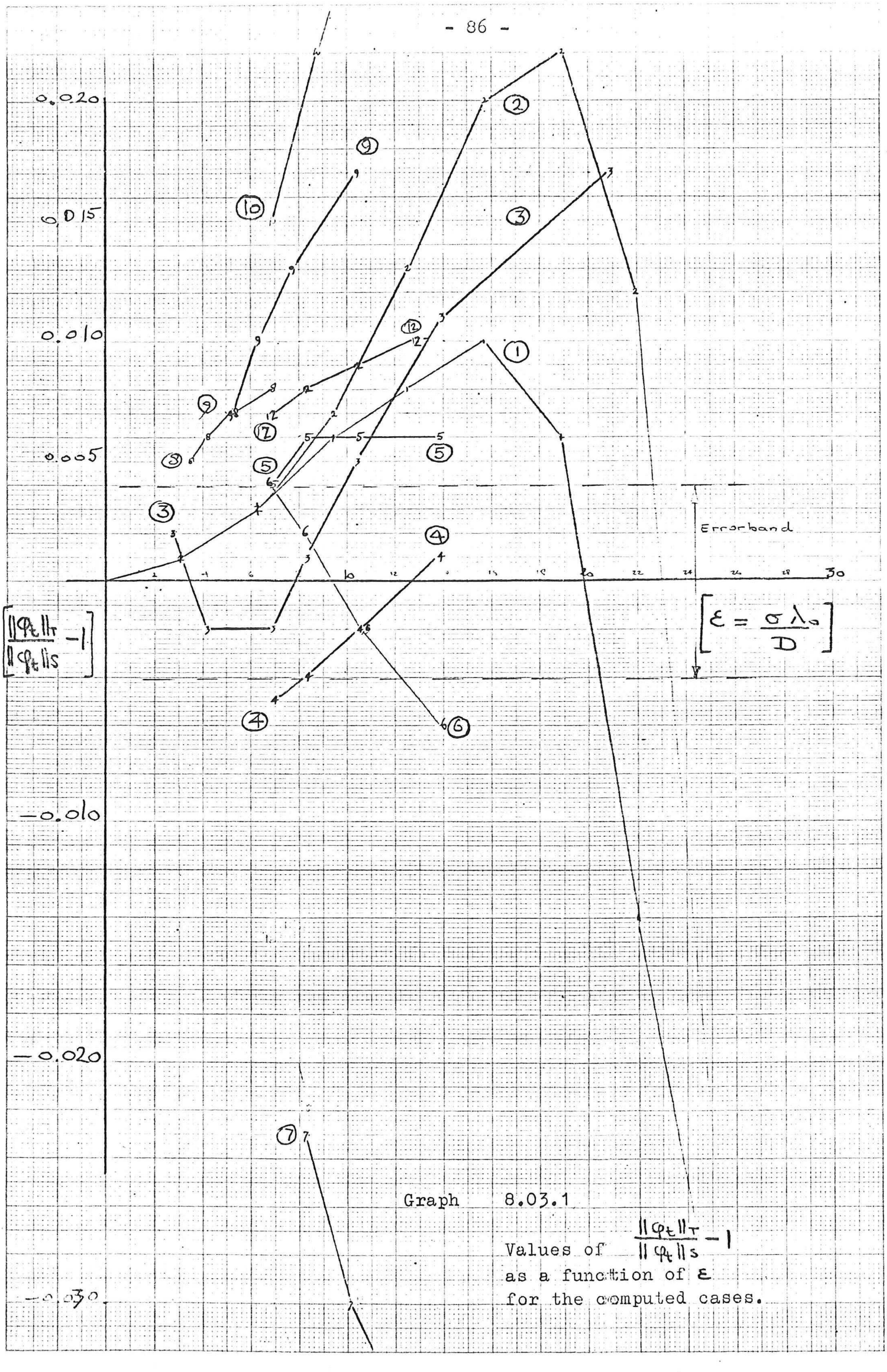
Graph (8.03.1) presents the comparison of the predicted values of $\|\varphi_t\|$ per model using the parameter $\frac{\|\varphi_t\|_T}{\|\varphi_t\|_S} - 1$ mapped versus ε.

The errorband in the graph has magnitude 4.10^{-3}.
(See Appendix 6.)

Within the errorband the results of the two models will be called " equal ".

It is clear from the results in graph (8.03.1) that there is no straightforward relationship between the parameter ε and the difference in the predictions by the two models.

The results of the various CASEs will be compared in the next chapter.



Graph 8.03.1

Values of $\frac{\|\varphi_t\|_T}{\|\varphi_t\|_S} - 1$ as a function of ε for the computed cases.

9 Discussion of the results

9.00 Introduction

In this chapter the results will be discussed.

In the derivation of the refraction - diffraction equation (see Appendix 3) the most important assumption made by Berkhoff was the ' near flatness ' of the bottom, for which the parametric requirement is:

$$\varepsilon = \frac{\sigma \lambda_0}{D} \qquad \text{must be small.}$$

The aim of this study is: to determine the limit of applicability of the refraction - diffraction equation.
For this purpose a computer program has been made by which the values of the wave potential at the water surface can be determined using the refraction - diffraction equation (Top View Model). Also the program computes the values of the wave potential using the set of equations from which the refraction - diffraction equation has been derived (The Side View Model).
This leads to the practical reinterpretation of the aim of this study: to determine when the predictions of the Side View Model and the Top View Model will be significantly different, whereby the expectation exists that the difference will be related to the value of ε.

The results (Graph 8.03.1) show that there is no direct relationship between the difference in the predictions and the value of ε.

In this chapter the influence of the other parameters (ScD, Ratio and ScB) will be investigated.
These influences will be compared with the results of CASE 3.

9.01 Results of CASE 3

A case in which the slope is the only parameter which is varied, is CASE 3 (see diagram 6.07.1).
The results for this case are for convenience repeated in graph (9.01.1).

The results of this case seem to confirm the expectations: with an increase of ε the difference in the predictions of the two models increases.

The difference becomes significant, using the criterion defined in Appendix 6 , for a value of $\varepsilon = 10$.

This result, however, is to be considered with great caution.
The results for the other cases (graph 8.03.1) show that this behaviour is not at all characteristic.

In the next section the influence of the other parameters will be studied, for ε having values of approximately $\varepsilon = 10$.

$\left[\frac{\|\varphi_t\|_T}{\|\varphi_t\|_S} - 1\right]$ ↑

0.008

0.006

0.004

0.002

5 10 15 20 [ε] →

-0.002

-0.004

Error band

CASE 3

ScD = 0.6

Ratio = 1÷3

ScB = 0.6

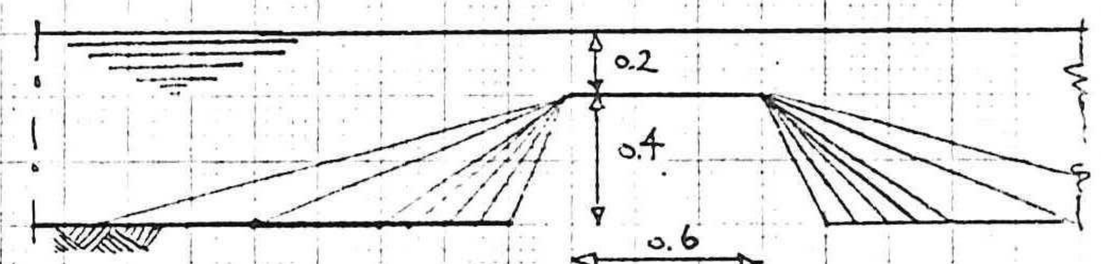


Figure 8.01.1 Presentation of the results of CASE 3

9.02 Influence of the parameters ScD, Ratio and ScB

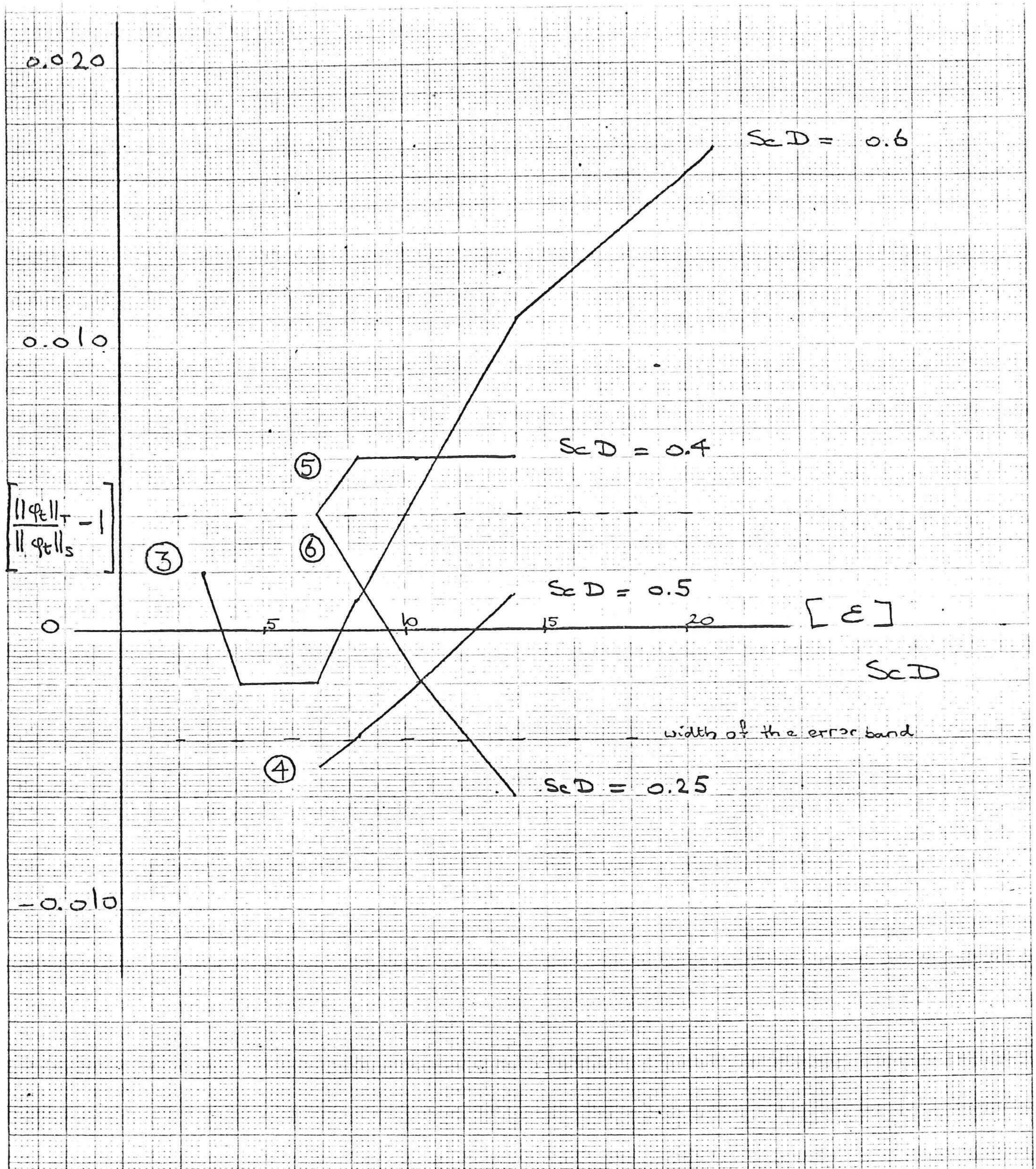
The cases which differ only in the value of ScD are presented in graph (9.02.1).

It is evident that variation of ScD greatly influences the results and that from the results in graph (9.02.1) no regularity can be discovered in the way the variation of the ScD affects the results.

The same is the case for those CASEs which only differ in the value of Ratio (graph 9.02.2) respectively in the value of ScB (graph 9.02.3).

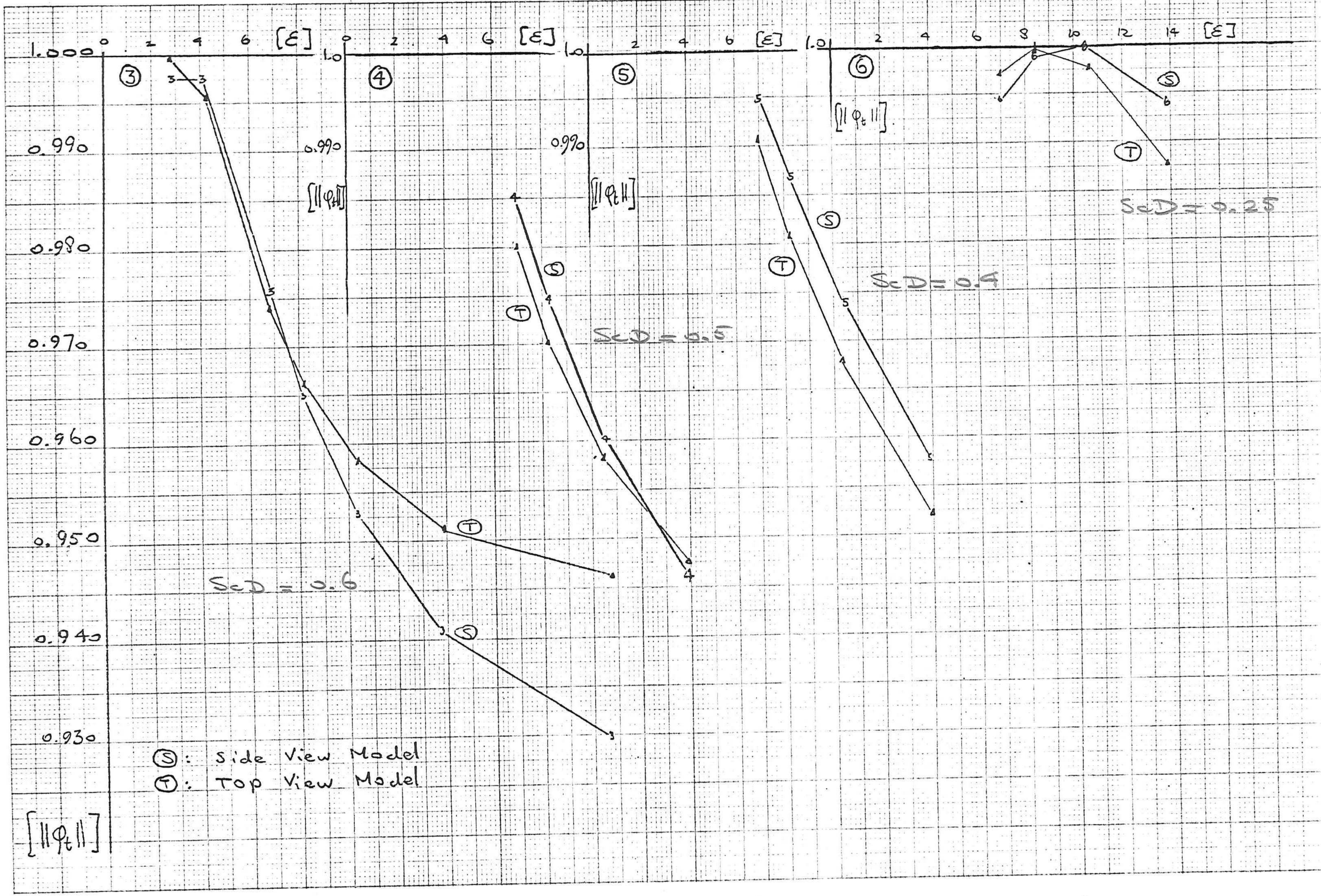
Notably the results of graph (9.02.3) suggests that local effects have to be accounted for in the comparison of the two models (and thus: in the search for the limits of applicability of the refraction - diffraction equation).

The result of CASE 7 (Ratio = 1.5) falls in the interval of nonlinear behaviour described in section 8.02, where the predictions for CASEs 1 and 2 show aberrational behaviour. Perhaps this is related to the fact that the result for this case differs greatly from the other results in graph (9.02.2).

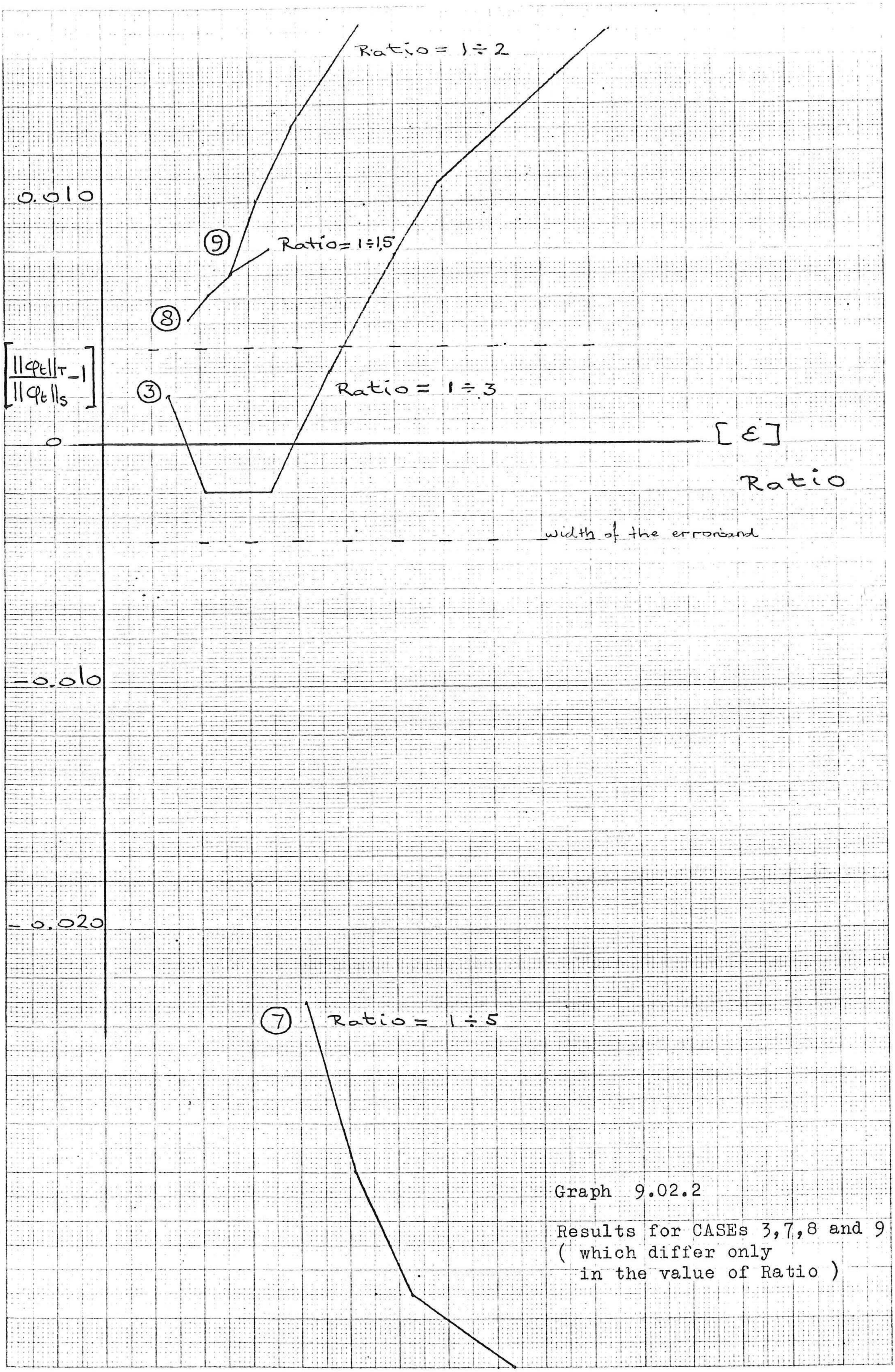


Graph 9.02.1

Results for CASEs 3,4,5 and 6
(which differ only
in the value of ScD)

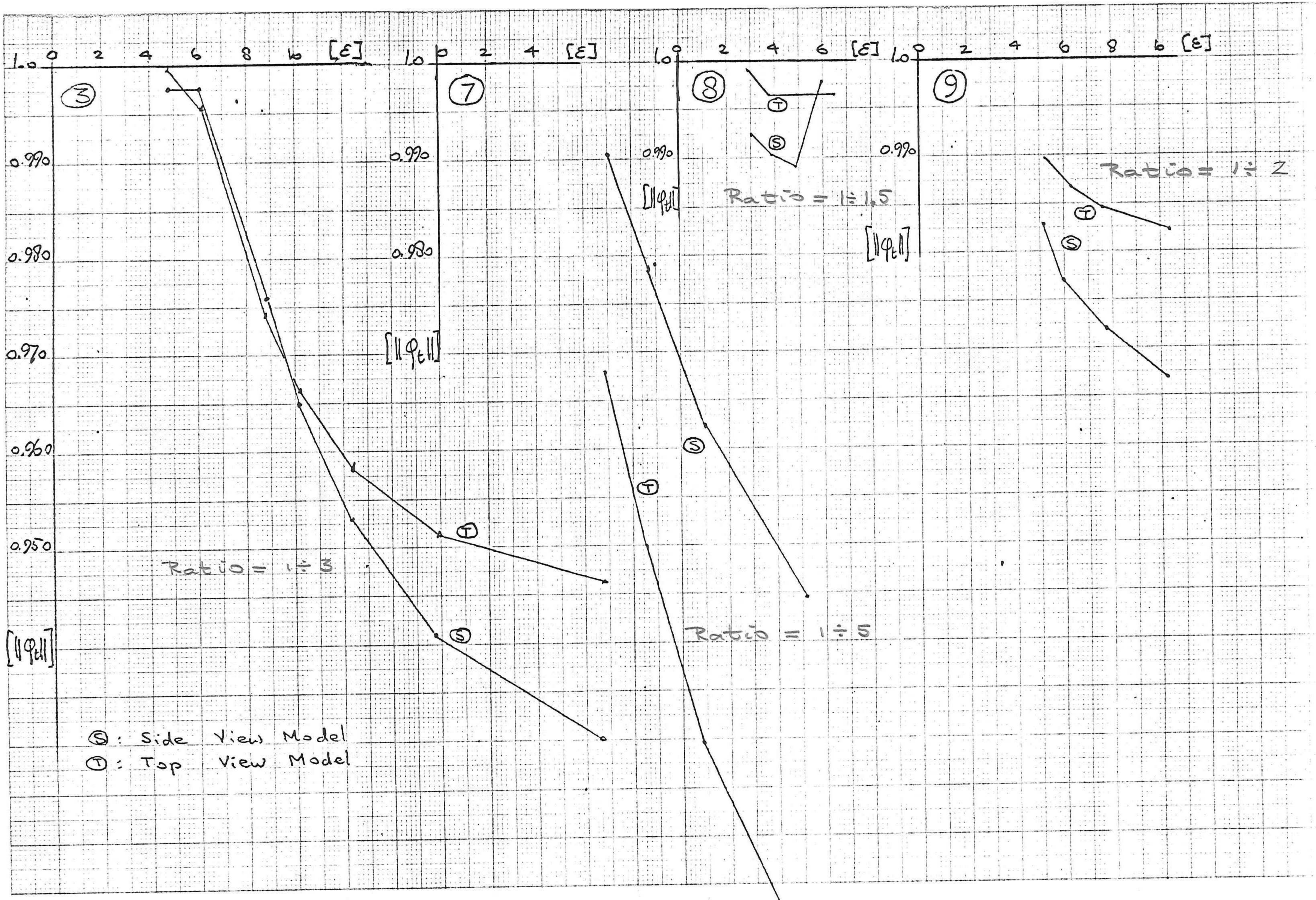


Graph 9.02.1A Compared values of $\|\varphi_t\|$, for variation of Sc D

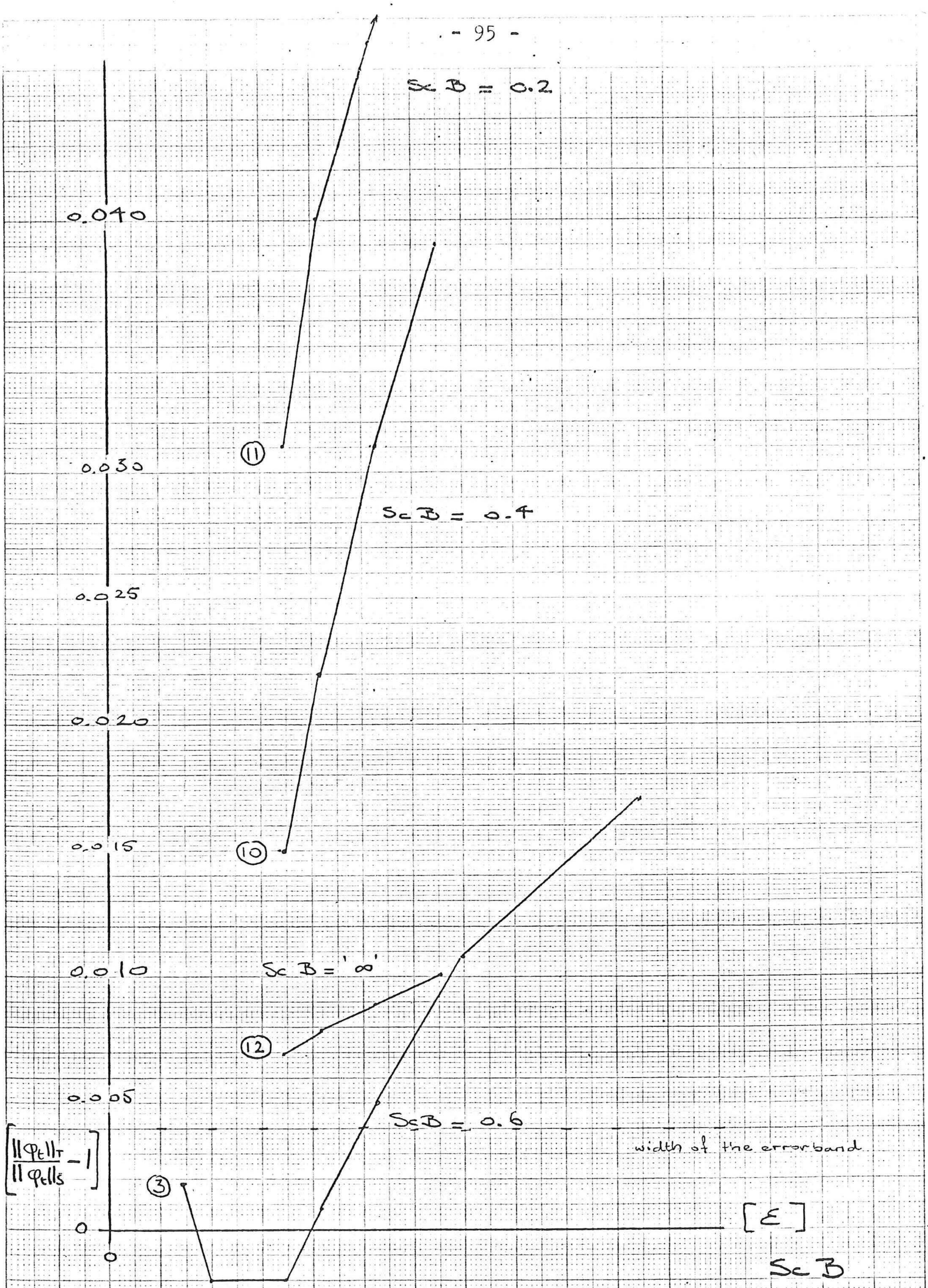


Graph 9.02.2

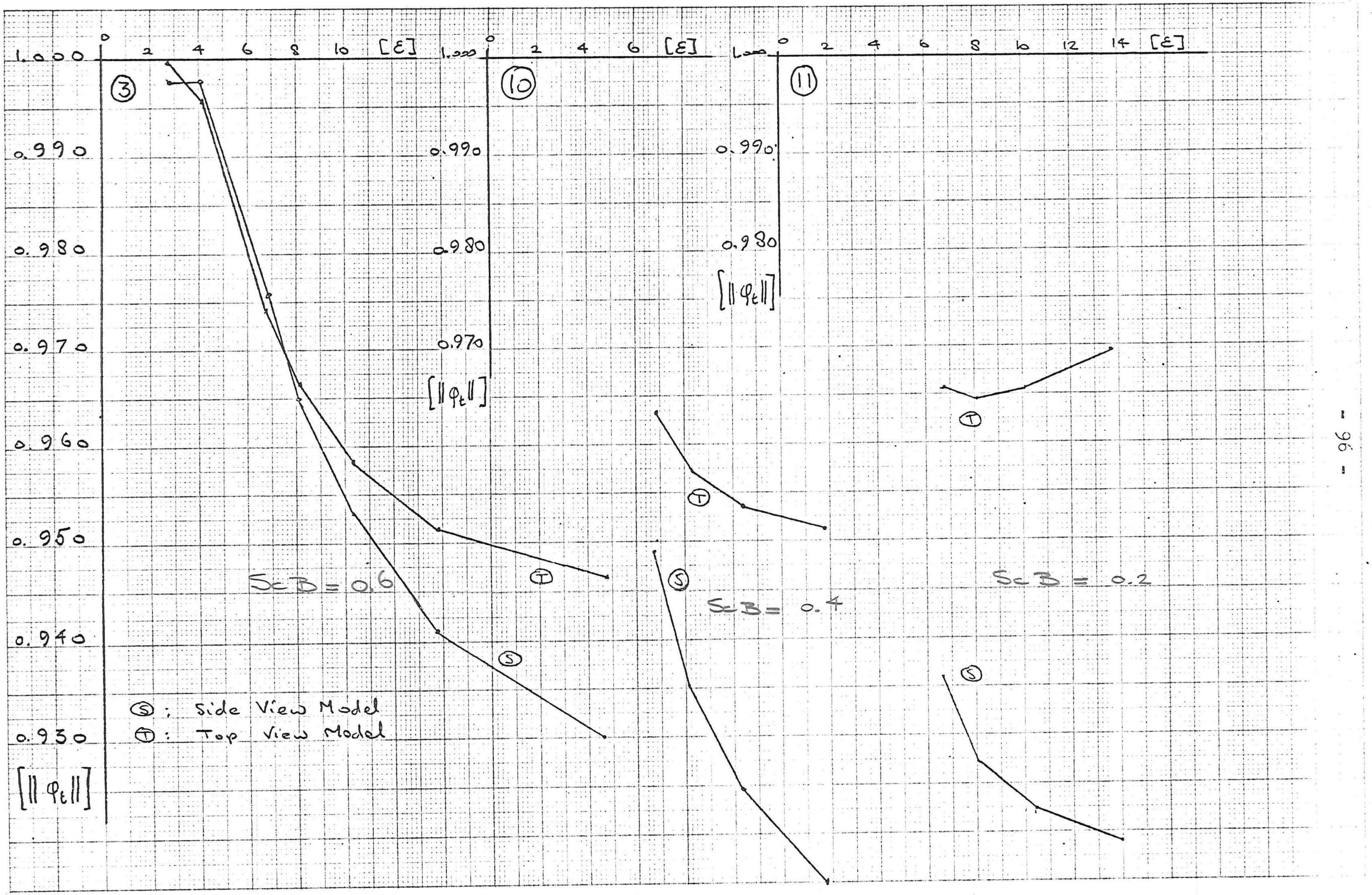
Results for CASEs 3,7,8 and 9
(which differ only
in the value of Ratio)



Graph 9.02.2A Compared values of $\|\varphi_t\|$, for variation of Ratio



Graph 9.02.3 Results for CASEs 3, 10, 11 and 12
(which differ only in the value of ScB)



Graph 9.02.3A Compared values of $\|\varphi_t\|$ for variation of ScB

10 Conclusions

This study has been undertaken in order to find out what is to be understood by a 'nearly flat bottom ' which forms the basic assumption in Berkhoff's derivation of the refraction - diffraction equation.
That is, it is an inquiry into the limits of applicability of the refraction - diffraction equation.

For the study a computational program has been made which calculates the wave potential at the water surface using the refraction - diffraction equation (Top View Model) and also using the set of equations from which the refraction - diffraction equation has been derived, based on the laplace equation (Side View Model).

The magnitude of the transmitted wave potential is used as the characteristic parameter: $\| \varphi_t \|$.

The results of the computation are compared in the form:

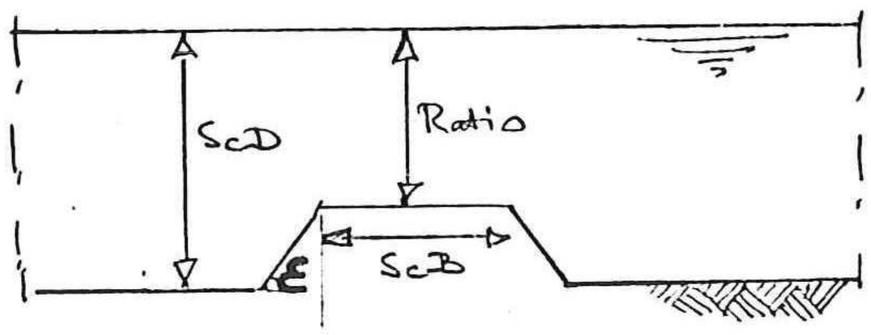
$$\frac{\| \varphi_t \|_T}{\| \varphi_t \|_S} - 1$$

A measure of significance of the difference of the predictions is determined.

The expectation was that a correlation would exist between the characteristic parameter of the bottomslope, ε, and the discrepancy in the predictions by the two models.

This correlation does indeed exist for one specific case (CASE 3 of the examined cases), but the behaviour is not at all characteristic.

It follows from the study that local parameters, characteristic for the bottom configuration, are of importance:



ScD : waterdepth
Ratio : relative waterdepth over dam
ScB : crestwidth
ℓ : bottomslope

Due to the influence of these parameters it has not been possible to reach the goal of this study and to determine limits of applicability of the refraction - diffraction equation.

Further study will be required to find a more suitable formulation of the characteristic parameter of the bottom slope, encorporating local characteristics of the bottom profile.

A major spin off of this study has been the development of a standard element of AFEP which contains a two dimensional description of the refraction - diffraction equation with general first order boundary conditions (complex number plane).
This model can handle a very broad set of bottom configurations.

11 Recommendations for further study

This study has produced a very versatile and generally useful computer program for the calculation of wave potential using either the Laplace equation or the refraction - diffraction equation with general first order boundary conditions in the complex plane.

It is recommended that the program be tested for its numerical behaviour (stability, optimal stepsize, etc.) for the various general cases for which it may be used so that it can be used as a standard program, without requiring each user to go through the tedious and costly actions of numerical testing.

It is recommended that the characteristic parameter of the bottom profile, ε, be reformulated in such a way that local behaviour of the bottom configuration is accounted for.

Once this has been done it will be possible to try to determine the limits of applicability of the refraction - diffraction equation.

It is recommended that the TOP VIEW MODEL (ELM301) be checked against experimental results, two dimensionally.

It is recommended that the Top View Model be used to determine the characteristics of wave transmission over a submerged barrier for oblique wave incidence. The existing information is scarce, and describes mainly experimental results on scale models for simple cases (of orthogonal wave incidence).

Comparison of the two dimensional Top View Model against a three dimensional 'Laplace model' will provide more insight into the practicality of the two models, especially when compared to experimental results or field data.

It is recommended that the models used in this study be " polished ", making them fit for general usage.

- economisation of storage and time.
- mesh generation considering the required stepsize as follows from numerical requirements and the anticipated wavelength.
- adapting the boundary conditions for oblique waves.

It is also recommended that the ~~a~~ initial phase of the students' encounters with the computer be closely guided. This greatly increases the effectivity of operating with the computer and saves much time and money.

12

Acknowledgements

I am truly grateful for the freedom which I have been given in undertaking this study and I thank professor Battjes and professor Vreugdenhil for it.

Nico Booy has been a very valuable advisor and I have greatly appreciated his helpful criticism and comments.

Thank you very much, Guus Segal, for the cheerful help and support you gave when I fled the gloom of computer-mania.

Appendix 1

Literature

1) M. Abramowitz — Handbook of mathematical functions
Dover press
1970

2) J.C.W. Berkhoff — Mathematical models for simple harmonic linear waterwaves, wave diffraction and refraction.
Delft Hydraulic Laboratory,
Publication nr. 163, april 1976
1976

3) R. Booy — Turbulentie (in dutch)
Collegedictaat TH Delft, C.T.
1978

4) P.M. Chirlian — Introduction to Fortran IV
Academic press
1973

5) P.M. Chirlian — Intoduction to structured fortran
Matrix Publ.
1979

6) R. Courant & D. Hilbert — Methods of mathematical physics
Wiley interscience
1953

7) T.M. Dick — On solid and permeable submerged breakwaters
Queens university, ontario,
1968

8) . Duncan — The principles of Galerkins method
Aero Res.Comm.Tech.Rep. nr 2, 589-612
1938

9) B. Finlayson — The method of weighted residuals and variational principles
Academic press
1972

10) O. Hinze — Turbulence
McGraw Hill
1975

11) Jv Kan & A. Segal
Eindige elementen methoden; differentie methoden; toepassingen in de techniek. (in dutch)
Collegedictaat, TH Delft
1979

12) H. Lamb
Hydrodynamics
Cambridge university press
1975.

13) B. LeMehauté
Submerged breakwater for silt deposotion reduction.
Queens university, ontario,
1968

14) J. Lepetit
Etude de la réfraction de la houle monochromatique par le calcul numérique.
Dépt. national de l'hydraulique, no. 9
1964

15) R. Lindsay
Mechanical radiation
McGraw Hill
1960

16) Lyngby
Formulae and tables; sinusoidal and cnoidal waves
Lyngby
1974

17) I. Mobarek & A. Fahim
Use of submerged breakwater in trapping sediment.
Cairo university, bulletin fac.of eng nr L5
1974

18) A. Mynett
Toepassing van de eindige elementen methode voor het berekenen van de vloeistof beweging rond een oscillerend schip op ondiep water. (in dutch)
Thesis, TH Delft, C.T.
1977

19) Rekencentrum
Q.E.D. gids
Rekencentrum, TH Delft
1978

20) Rekencentrum — Stuurtaal basiscursus
Rekencentrum TH Delft
1978

21) Rekencentrum — TSO/RJE handleiding
Rekencentrum TH Delft
1975

22) Rekencentrum — Fortran IV taalbeschrijving
Rekencentrum TH Delft
1976

23) A. Segal — AFEP User manual
TH Delft
1975

24) A. Segal & . Briemer — AFEP report NA-13
TH Delft
1975

25) C. Shannon — Theory of communications
Urbana
1949

26) J. Steketee — College Aero 1
TH Delft
1973

27) G. Strang & J. Fix — An analysis of the finite element method
Prentice Hall
1973

28) R. Wiegel — Transmission of waves past a vertical thin barrier
Journ. of waterways & harbours div.
1960

29) S. Yuan — Foundations of fluid mechanics
Prentice Hall
1964

30) O. Zienkiewicz — The finite element method in engineering science
McGraw Hill
1971

31) O. Zienkiewicz — Finite elements in fluids, Vol.1
Wiley interscience
1975

Appendix 2

Derivation of the free surface boundary condition

The surface of the fluid varies in place and time. The general description of a time and place dependent variation, for any function Φ has the form:

$$\frac{d\Phi}{dt} = \frac{\partial\Phi}{\partial t} + \frac{\partial\Phi}{\partial x}\cdot\frac{dx}{dt} + \frac{\partial\Phi}{\partial y}\cdot\frac{dy}{dt} + \frac{\partial\Phi}{\partial z}\cdot\frac{dz}{dt} =$$

$$= \frac{\partial\Phi}{\partial t} + \nabla\Phi\cdot\vec{v} \qquad (2.1)$$

In a field where the variation of momentum ($\Phi =: \rho v_i$) is the characteristic parameter and where conservation of momentum may be assumed ($\frac{d\rho v_i}{dt} = 0$) the equation expands to the Navier Stokes equation (Hinze, 1975; R. Booy, 1978):

$$\frac{\partial\rho v_i}{\partial t} + (\nabla\rho\vec{v}_i)\vec{v}_j + \rho(\vec{v}_i\nabla)\vec{v}_i + \nabla p_i - \mu\nabla^2 v_i - \rho F = 0 \qquad (2.2)$$

where: p : pressure
F : force field

using 1) $(\vec{v}.\nabla)\vec{v} = \frac{1}{2}\nabla(\vec{v}.\vec{v}) - \vec{v}*(\nabla*\vec{v})$

2) $\nabla * v \triangleq 0$ (no rotation: (2.01.5))

3) $F \triangleq -\nabla\Omega$ force potential

this becomes:

$$\frac{\partial\rho v_i}{\partial t} + (\nabla\rho\vec{v}_i)\vec{v}_j + \rho\left(\frac{1}{2}\nabla(\vec{v}_i.\vec{v}_i)\right) + \nabla p_i + \rho\nabla\Omega = 0 \qquad (2.3)$$

using 4) $\frac{d\rho}{dt} = \frac{\partial \rho}{\partial t} + \nabla\rho.\vec{v} \overset{As}{\equiv} 0$ uniform and homogenous density distribution

5) $\nabla\rho \overset{As}{\equiv} 0$

6) $\vec{v} \overset{As}{\equiv} \nabla\phi$ (follows from 2))

7) $\frac{\partial \nabla\phi}{\partial t} = \nabla \frac{\partial \phi}{\partial t}$

this becomes:

$$\nabla \frac{\partial \phi}{\partial t} + \nabla\left(\tfrac{1}{2}\nabla\phi\nabla\phi\right) - \frac{\nabla}{\rho}p + \nabla\Omega = 0 \qquad (2.4)$$

using 8) $\int dr_3$ "zooming in on the surface"

9) $p \overset{As}{=:} 0$

this becomes:

$$\frac{\partial \phi}{\partial t} + \tfrac{1}{2}\nabla\phi\nabla\phi + \nabla\Omega = f\langle t\rangle \qquad (2.5)$$

using 10) $f\langle t\rangle \overset{As}{=:} 0$

11) $\Omega = gz$ only gravity is considered for the potential force

12) $z =: \eta$:at the surface

this becomes:

$$\frac{\partial \phi}{\partial t} + \tfrac{1}{2}\nabla\phi\nabla\phi + g\eta = 0 \qquad (2.6)$$

:at the surface

" the Bernoulli equation "

Linearising the equation by disregarding the higher order variations of :

$$\frac{\partial \Phi}{\partial t} + g\eta = 0 \qquad (2.7)$$

at the average surface level.

η describes the position of the surface ($\eta\{x,y,t\}$) but is not known explicitly.

A description may be determined by assuming that a particle at the surface moves with the surface:

$$\frac{dz}{dt} = \frac{\partial \eta}{\partial t} + \frac{\partial \eta}{\partial x} \cdot \frac{dx}{dt} + \frac{\partial \eta}{\partial y} \cdot \frac{dy}{dt} \qquad (2.8)$$

Disregarding the higher order (non-linear) variations:

$\frac{\partial \eta}{\partial x} \cdot \frac{dx}{dt}$ and $\frac{d\eta}{\partial y} \cdot \frac{dy}{dt}$, and

using $\frac{dz}{dt} = \frac{\partial \Phi}{\partial z}$

this becomes:

$$\frac{\partial \eta}{\partial t} - \frac{\partial \Phi}{\partial z} = 0 \qquad (2.9)$$

Combining the equation of the position of the surface (2.9) with the linearised Bernoulli equation (2.7):

(2.7): $\frac{\partial \Phi}{\partial t} + g\eta = 0 \quad \Rightarrow \quad \frac{\partial^2 \Phi}{\partial t^2} + g\frac{\partial \eta}{\partial t} = 0$

(2.9): $\frac{\partial \eta}{\partial t} - \frac{\partial \Phi}{\partial z} = 0$

this produces:

$$\frac{\partial \Phi}{\partial z} + \frac{1}{g} \cdot \frac{\partial^2 \Phi}{\partial t^2} = 0 \qquad \text{at the surface} \qquad (2.10)$$

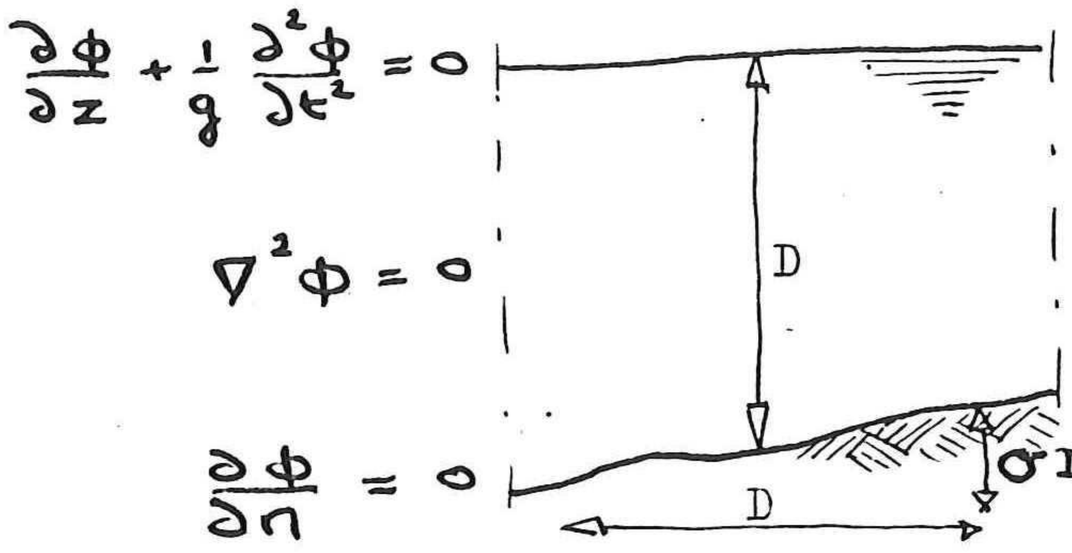
This is the free surface boundary condition:

$$\frac{\partial \Phi}{\partial z} + \frac{1}{g} \frac{\partial^2 \Phi}{\partial t^2} = 0$$

Appendix 3

Derivation of the refraction - diffraction equation

The equations for the fluid field are (2.04.1):



D: mean water depth
σ: bottom slope
λ_o: deepwater wave-length

(3.1)

A solution exists for the case of a flat horizontal bottom (2.05.1):

$$\varphi = \frac{ga}{\omega} \cdot \frac{\cosh(k(h+z))}{\cosh(kh)} \cdot e^{i(\omega t - kx)} \qquad (3.2)$$

This solution can be considered to be the solution at the surface ($\varphi = \frac{ga}{\omega} \cdot e^{i(\omega t - kx)}$) and a function describing the distribution of this solution over the vertical: $\frac{\cosh(k(h+z))}{\cosh(kh)}$.

Berkhoff (1976) assumed that for a 'relatively flat' bottom this solution would still be approximately valid:

He assumed a solution of the form:

$$\varphi\langle x,y,z,t\rangle = a \cdot \frac{\cosh(k(h+z))}{\cosh(kh)} \cdot \varphi\langle x,y,z\rangle \cdot e^{i(\omega t - kx)} \qquad (3.3)$$

The potential $\varphi\langle x,y,z\rangle$ will be a very weak function of z, so it is assumed, most of the variation in the direction of the vertical already being gescribed by the term $\frac{\cosh(k(h+z))}{\cosh(kh)}$.

The function $\varphi\langle x,y,z\rangle$ may be expanded in a power series:

$$\varphi\langle x,y,z\rangle = \varphi_o\langle x,y\rangle + \sigma^2 z^2 \varphi_1\langle x,y\rangle + \sigma^4 z^4 \varphi_2\langle x,y\rangle + \ldots \qquad (3.4)$$

At the surface: z=0 and the power series will provide the original potential: $\varphi\langle x,y,z=0\rangle \equiv \varphi_o\langle x,y\rangle$

At the bottom z has its maximal value: z=D and neglection of the higher order terms will introduce an error of order of magnitude: $\sigma^2 D^2$.

Therefore, if $\sigma^2 D^2 \ll 1$, then: $\varphi\langle x,y,z\rangle \approx \varphi_o\langle x,y\rangle$

Rescaling the equation, using " D " as a unit of distance along the bottom ($\sigma^* := \frac{\sigma}{D}$) the criterion becomes:

$$\sigma^{*2} \ll 1 \qquad (3.5)$$

Using the free surface characteristic length to make the parameter dimensionless:

$$\varepsilon := \frac{\sigma}{D} . \lambda_o \qquad (3.6)$$

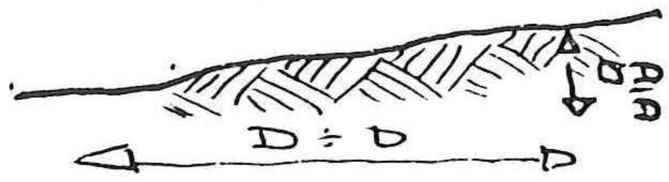
Using this the equations of the rescaled set become:

(see next page)

$$\frac{\partial \phi}{\partial z} + \frac{1}{g}\frac{\partial^2 \phi}{\partial t^2} = 0$$

$$\nabla^2 \phi = 0$$

$$\varepsilon\left(\frac{\partial h}{\partial x}\cdot\frac{\partial \Phi}{\partial x} + \frac{\partial h}{\partial y}\cdot\frac{\partial \Phi}{\partial y}\right) + \frac{\partial \Phi}{\partial z} = 0 \qquad (3.7)$$



using 1) $\varphi = a \cdot \frac{\cosh(k(h+z))}{\cosh(kh)} \cdot \varphi_0\langle x,y\rangle \cdot e^{i(\omega t - kx)}$

2) averaging for the mean energy flux by integration over the vertical from bottom to surface

Berkhoff obtains:

$$\nabla \cdot cc_g \nabla \varphi_0 + k^2 cc_g \varphi_0 = 0 \qquad (3.8)$$

the " Refraction - Diffraction equation "

The basic assumption (thus: the major possible restriction of the validity of the equation) is:

$$\sigma^2 D^2 << 1$$

which in scaled and dimensionless form becomes (3.6):

$$\varepsilon^2 << 1$$

where:

$$\varepsilon = \frac{\sigma \lambda_0}{D}$$

Appendix 4

Listing of the computer program

This appendix contains the complete listing of the computer program used for this study.

The program, written in Fortran, is based on the Finite Element Package AFEP.

Next to the program listing a brief description of the purpose and workings of its parts is presented.

For a more complete understanding of the program also consult chapters 3 and 4, the AFEP User Manual and texts on Fortran IV if required.

```
//AWJOED JOB 511,B,TIME=(10,39),REGION=2048K,MSGLEVEL=(2,0)
/*JOBPARM LINES=9
// EXEC FORTXCG,PARM.GO='LET,EP=MAIN,SIZE=MAX-12K'
//FORT.SYSIN DD *
C
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION COOR(3000),KELEM(15000),KBOUND(3000),KBNDPT(500),
     V          INTMAT(6),AMAT(90000),USOL(2000),RHSD(2000),
     V          IWORK(2000),WORK(500),DEPTH(200),USER(40),IUSER(40),
     V          IINPUT(200),RINPUT(99),BORDER(300),CONTLN(16)
      COMMON /CACTL/  IELEM,ITYPE,ISOORT,INPELM,ICOUNT,IFIRST,NOI,ISTART
     V      /CPLOTC/ SCALE,NCONT,NCEL,MCEL,ITURN,IMARK,NUMD
C
      COOR(1) = 3000
      KELEM(1)=15000
      KBOUND(1)=3000
      KBNDPT(1)= 500
      INTMAT(1)=    6
      AMAT(1)  =90000
      USOL(1)  =2000
      RHSD(1)  =2000
      IWORK(1) =2000
      WORK(1)  = 500
      DEPTH(1) = 200
      USER(1)  =  40
      IUSER(1) =  40
      IINPUT(1)= 200
      RINPUT(1)=  99
      BORDER(1)= 300
      CONTLN(1)=  16
C
C     SCO = 40-1
C
      OMEGA = 100
C
      RATIO = 300
C
      ITEST = 1
C
      NRUNS = 10
C
      CALL INTAKE(IINPUT,RINPUT,SCO,ITEST,RATIO)
      DO 1 I=1,NRUNS
      CALL VARY(I,RINPUT,SCO,ITEST)
      CALL MESDOD(I,KELEM,COOR,KBOUND,KBNDPT,IWORK,WORK,
     V            IINPUT,RINPUT,3000,0D0)
      CALL DEEP(KBNDPT,COOR,DEPTH,BORDER,N1,OMEGA,RATIO)
      IF (I.EQ.1)
     VCALL COMBA(2,KELEM,KBOUND,INTMAT)
      CALL AUSER(IUSER,USER,DEPTH,KBNDPT,COOR,OMEGA,N1)
      CALL MATRIX(-1,AMAT,COOR,KELEM,KBOUND,WORK,IUSER,USER,INTMAT,IWORK
     V           )
      CALL BUSER(IUSER,USER)
      CALL RHSIDE(-1,RHSD,COOR,KELEM,KBOUND,WORK,IUSER,USER,AMAT,USOL)
      CALL SOLVE(0,AMAT,USOL,RHSD,INTMAT,KBOUND)
      IF (I.EQ.1) WRITE(6,*) (USER(9+J-1),J=1,10)
      CALL GRAPH(12,7,USOL,COOR,KBNDPT)
      CALL INPLOT(USOL,KBNDPT,CONTLN,1,1,6,2,2)
      NCONT =.2*NRUNS
      CALL PLOTCN(1,KOUTP,COOR,KBNDPT(5)-6,KBNDPT,USOL,30D0,26D0,
     V            CONTLN,11)
      CALL INPLOT(USOL,KBNDPT,CONTLN,2,1,6,2,2)
      CALL PLOTCN(2,KOUTP,COOR,KBNDPT(5)-6,KBNDPT,USOL,30D0,26D0,
     V            CONTLN,11)
C
    1 CONTINUE
C
C     "END OF THE MAIN PROGRAM"
C
      STOP
      END
```

JOB INITIATION

€ "SIDE VIEW MODEL"

reservations for array space as required by FORTRAN.

common blocks required by AFEP for the element matrix and subroutine PLOTCN

storing the size of each array in the first position of the array. as required by AFEP.

The variable parameters of the test.

Only "NRUNS" has to be changed in line 3650 as well if change would be made.

```
C     " NEXT: THE REQUIRED SUBROUTINES "
C
C
      SUBROUTINE INTAKE(IINPUT,RINPUT,SCD,ITEST,RATIO)
C
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION IINPUT(1),RINPUT(1)
      READ(55,*) (IINPUT(J+5),J=1,91)
      READ(55,*) (RINPUT(J+5),J=1,30)
C
      GO TO (1,2,3),ITEST
C
    1 RINPUT(9) = SCD * RINPUT(9)
      RINPUT(11)= SCD * RINPUT(11)
      RINPUT(13)= SCD * RINPUT(13)
      RINPUT(16)= SCD * RINPUT(16)
      RINPUT(18)= SCD * RINPUT(18)
      RINPUT(21)= SCD * RINPUT(21)
      RETURN
    2 RINPUT(9) = SCD * RINPUT(9)
      RINPUT(11)= SCD * RINPUT(11)
      RINPUT(13)= SCD * RINPUT(13) / RATIO
      RINPUT(16)= SCD * RINPUT(16) / RATIO
      RINPUT(18)= SCD * RINPUT(18)
      RINPUT(21)= SCD * RINPUT(21)
      RETURN
    3 RINPUT(9) = SCD * RINPUT(9)
      RINPUT(11)= SCD * RINPUT(11)
      RINPUT(13)= SCD * RINPUT(13) / RATIO
      RINPUT(16)= SCD * RINPUT(16) / RATIO
      RINPUT(18)= SCD * RINPUT(18) / RATIO
      RINPUT(21)= SCD * RINPUT(21) / RATIO
      RETURN
      END
C
C
      SUBROUTINE VARY(I,RINPUT,SCD,ITEST)
C
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION RINPUT(1),TOE(4)
C
           TOE(1) = 8D-1
           TOE(2) = 4D-1
           TOE(3) = 2D-1
           TOE(4) = 1D-1
      TOEDAM = TOE(I)
C
      GO TO (1,2,3),ITEST
C
    1 IF (I.GE.11) STOP
           TOPDAM = SCD*(-1D0*(I-1)*(1D-1))
                  RINPUT(13) = TOPDAM
                  RINPUT(16) = TOPDAM
      RETURN
    2 IF (I.GE.5) STOP
         RINPUT(10) = 2.2 - TOEDAM
         RINPUT(31) = RINPUT(10)
         RINPUT(17) = 2.8 + TOEDAM
         RINPUT(24) = RINPUT(17)
      RETURN
    3 IF (I.GE.5) STOP
         RINPUT(10) = 2.2 - TOEDAM
         RINPUT(31) = RINPUT(10)
      RETURN
      END
```

Subroutine "INTAKE" and "VARY" depend on the choice of ITEST.

These subroutines change the data required for the making of the mesh by mesh generator MESH2D.
These data are read from dataset "STORG (INDATA)" by means of FT55F001 (see line 3210).

Observe that in this case "INTAKE" is called only once, and that all variations are made at each run by "VARY".
(compare versus remark at line 4170).

The form of the field, the number of nodal points per boundary, is described in "STORE (INDATA)". see line 3210

① raises the height of the trapezoidal dam

② displaces the toe of the symmetrical dam

③ displaces the toe of "halfdam".

```
      SUBROUTINE DEEP(KBNDPT,COOR,DEPTH,BORDER,N1,OMEGA,RATIO)
C
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION KBNDPT(1),COOR(1),DEPTH(1),BORDER(1)
      NBN=KBNDPT(5)
C
       NBN1=KBNDPT(1+NBN) - KBNDPT(0+NBN) + 1
       NBN2 =KBNDPT(6+NBN) - KBNDPT(1+NBN) + 1
       N1=NBN2+5
         DEPTH(2)   = NBN2
         DEPTH(3)   = N1
C
      DO 10 I=6,N1
         BORDER(I) = COOR(2*KBNDPT(I+NBN1-1)+4)
         DEPTH(I) = -COOR(2*KBNDPT(I+NBN1-1)+5)
   10 CONTINUE
C
C      IF REQUIRED, THE ARRAYS (DEPTH),(BORDER) MAY BE PRINTED BY
C     WRITE(6,*) DEPTH,BORDER
C     WRITE(6,*) KBNDPT
C     WRITE(10,*) N1,OMEGA,RATIO
C     WRITE(10,*) (DEPTH(I),I=1,N1)
C     WRITE(10,*) (BORDER(I),I=1,N1)
C
      RETURN
      END
C
C
        SUBROUTINE AUSER(IUSER,USER,DEPTH,KBNDPT,COOR,OMEGA,N1)
C
C
       IMPLICIT REAL *8(A-H,O-Z)
       DIMENSION IUSER(1),USER(1),DEPTH(1),KBNDPT(1),F(15),COOR(1)
       COMMON /CACTL/ ID(7),ISTART
      DO 1 I=6,40
           IUSER(I) = 0
            USER(I) = 0
    1    CONTINUE
      GRAV   = 100
      SURFAS = - (OMEGA**2)/GRAV
C
C      NEWTON-RAPHSON IS USED TO DETERMINE LOCAL WAVE-LENGTH:
C
      M = N1-5
      DO 2 I = 6,N1,M
      ITER   = 0
      AK     = 1.5
   10 ITER = ITER+1
C
      IF (ITER.LE.20) GO TO 20
      WRITE(6,11) ITER
   11 FORMAT('0THE ITERATION FAILS; ITER .GE.',I3,//)
      GO TO 22
C
   20 DELTA = (OMEGA**2) - GRAV*AK*DTANH(AK*DEPTH(I))
      AK    = AK+DELTA/(GRAV*(DTANH(AK*DEPTH(I))+AK*DEPTH(I)/
     V                  (DCOSH(AK*DEPTH(I))*DCOSH(AK*DEPTH(I)))))
      IF (DABS(DELTA).GE.10-6) GO TO 10
C
   22 IF (I.EQ.6) A2B1 = AK
                  A2B3 = AK
```

"DEEP" :

For each point along the border the depth is computed using the coordinates of surface and bottom boundaries.

The information is made available for the Top view Model using background array "STORE(LINK)" see lines 3200, 4080, 6420.

"AUSER"

1) computation of the value of "k" at the boundary (see page -15-) ; lines 1740 → 1950

2) print the values of the parameters characteristic for the problem lines 1960 → 2020

3) computation of the distribution of the wave potential over the height as follows from the analytical flat bottom solution (see page - -), which is used in the Berkhoff equation (page - -) lines 2060 → 2100

4) Filling IUSER and USER as described in appendix 5.
First the arrays are cleared by filling them with zeroes (lines 1680-1710
Then the pointers and values are inscribed lines 2220 - 2390

```
      22  IF (I.EQ.6) A2B1 = AK
                       A2B3 = AK
       2  CONTINUE
          WRITE(6,888) OMEGA,DEPTH(6),A2B1,A2B3
     888  FORMAT('OOMEGA =  ',F12.2,/
     V            '     SCO =  ',F12.2,/
     V            '    A2B1 =  ',F12.2,/
     V            '    A2B3 =  ',F12.2,/
     V            ' THE VALUE OF THE DEPTH IS DETERMINED ',/
     V            ' AT THE INFLUX BOUNDARY ',//)
C
C                 THE INFLUX BOUNDARY CONDITION IS PRESCRIBED.
C
          NBN=KBNDPT(5)
          NBOUN = KBNDPT(1+NBN) - KBNDPT(0+NBN) + 1
          DEEP  = DEPTH(6)
          DO 3 I=1,NBOUN
               J= I + KBNDPT(0+NBN) - 1
               Y = COOR(2*KBNDPT(J)+5)
             F(I)= DCOSH((A2B1)*(Y+DEEP))/DCOSH((A2B1)*DEEP)
          USER(8+I)= F(I)*2*A2B1
C
       3  CONTINUE
          WRITE(6,999) (F(I),I=1,NBOUN)
     999  FORMAT('0THE DISTRIBUTION FUNCTION OVER THE HEIGHT ',/
     V            ' AS ASSUMED BY BERKHOFF : ',/1DF8.4,//)
C
C                   ARRAY IUSER IS FILLED
C
          ISTART = 10
          IUSER(6) = 1
          IUSER(7) = 1
          IUSER(8) = 0
          IUSER(9) = 1
          IUSER(10)= 1
          IUSER(11) = 16
          IUSER(12) = 20
          IUSER(13) = 24
          IUSER(14) = 28
          IUSER(15) = 32
          IUSER(21) =  9
          IUSER(23) = -6
          IUSER(31) = -7
          IUSER(34) = -8
          USER(6) = +A2B1
          USER(7) = +A2B3
          USER(8) = SURFAS
C         WRITE(6,*) (IUSER(J),J=1,40)
C         WRITE(6,*) (USER(J),J=1,40)
          RETURN
          END
C
          SUBROUTINE BUSER(IUSER,USER)
C
          IMPLICIT REAL *8(A-H,O-Z)
          DIMENSION IUSER(1),USER(1)
          COMMON /CACTL/ ID(7),ISTART
          IUSER(6) = 0
          IUSER(7) = 1
          IUSER(8) = 0
          IUSER(9) = 0
          IUSER(10)= 0
          RETURN
          END
```

Lines 2230 → 2270 form the "switches"; counterpart of the set described in "BUSER".

Array KBNDPT is the AFEP array, made by the mesh generator, in which the positions of the boundary points are stored as well as the number of points per boundary.

"BUSER" contains the "switches" described in appendix 5.

```
      SUBROUTINE INPLOT(USOL,KBNDPT,CONTLN,ICHOIS,NUM1,NUM2,M1,M2)
C     FOR THIS CASE: NUM1=:1, NUM2=:6, M1=:M2=:2. SEE SUBROUTINE CALL
C
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION USOL(1),KBNDPT(1),CONTLN(1)
      COMMON/CPLOTC/ SCALE,NCONT,NCEL,MCEL,ITURN,IMARK,NUMD
C
      NBN=KBNDPT(5)
       N =KBNDPT(NUM1+NBN) - KBNDPT(0+NBN) + 1
      NCEL = N*M1
       M =KBNDPT(NUM2+NBN) - KBNDPT(NUM1+NBN) + 1
      MCEL = M*M2
      SCALE = 100
C
        AMAX = 300
        AMIN =-AMAX
         NUS = USOL(3)
               IINIT = ICHOIS+5
      DO 1 J=IINIT,NUS,2
                   A = USOL(J)
         IF (A.GT.AMAX) AMAX=A
         IF (A.LT.AMIN) AMIN=A
    1 CONTINUE
         RANGE = AMAX - AMIN
         STEP  = RANGE/101
      DO 2 J=1,11
                   CONTLN(J+5) = AMIN + (J-1)*STEP
    2 CONTINUE
      WRITE(6,*) (CONTLN(J+5),J=1,11)
      RETURN
      END
        SUBROUTINE GRAPH(NUM1,NUM2,USOL,COOR,KBNDPT)
C
      IMPLICIT REAL *8 (A-H,O-Z)
      DIMENSION USOL(1),COOR(1),KBNDPT(1)
      REAL *4  FI1(100),FI2(100),ABSFIE(100)
      INTEGER *4 IARR(100)
      NBN = KBNDPT(5)
      NBOUN = KBNDPT(12+NBN) - KBNDPT(7+NBN) + 1
C
      DO 1 I=1,NBOUN
                   LOCUS = 2*KBNDPT(NBN-I)
      AID1 = USOL(4+LOCUS)
      AID2 = USOL(5+LOCUS)
C
      FI1(I) = AID1
      FI2(I) = AID2
      ABSFIE(I) = DSQRT(AID1**2 + AID2**2)
    1 CONTINUE
C
      CALL PRFIE(FI1,IARR,NBOUN,1,NBOUN)
      CALL PRFIE(FI2,IARR,NBOUN,1,NBOUN)
      CALL PRFIE(ABSFIE,IARR,NBOUN,1,NBOUN)
C
      RETURN
      END
//GO.SYSLIB DD DISP=SHR,DSN=WITA.BRON
// DD DSN=SYS2.FORTLIB,DISP=SHR
// DD DSN=SYS2.NUMLIBF,DISP=SHR
// DD DSN=SYS2.PLOTLIBF,DISP=SHR
// DD DSN=WITA.PRFT2,DISP=SHR
// DD DSN=WWWBLVT.PROGR,DISP=SHR
//GO.FT01F001 DD DSN=WITA.PRFT3,DISP=SHR
//GO.FT10F001 DD DSN=WWWBDEQ.STORE(LINK1),DISP=SHR
//GO.FT05F001 DD DSN=WWWBDEQ.STORE(INDATA),DISP=SHR
//GO.FT06F001 DD DSN=WWWBDEQ.USOL1,DISP=SHR
//GO.PLTDS DD DSN=WWWBDEQ.LP,DISP=SHR
//GO.FT04F001 DD DSN=WWWBDEQ.LC,DISP=SHR
//GO.SYSIN DD *
IN
   3  5  2  0
011110
   3  2  2  2  2
   147
   148
   148
   148
   148
```

"INPLOT":

Uses the number of points along the boundaries to determine the number of cells for subroutine PLOTCN (See AFET Manual)

The threshold levels of the potential are chosen at intervals of 10% between the maximum and minimum values. The calculated values of these threshold levels are stored in array "CONTLN", as required by "PLOTCN".

GRAPH:

The subroutine "PRFIE", stored in library WWWBLVT.PROGR (see line 3180), is used to present the values of the potential as determined by "SOLVE".
The numbers have to be changed from real*8 to real*4.

Printed are the values:
φ_1 ("real" part of the potential)
φ_2 ("imaginary" ")
$\| \varphi \| = \sqrt{(\varphi_1)^2 + (\varphi_2)^2}$

The Libraries required for the job as required for AFET and PRFIE.

Input and output datasets:

STORE(LINK1) : see lines 1570, 4080, 6420
STORE(INDATA) : 790, 800

LP / required by MES82D
LC / PLOTCN

NUMERICAL DATA : See AIEP MANUAL, subroutine INIT

The numbers 147, 148 indicate the elements ELM 147, ELM 148, ELR 148 See page — 44 —

```
//
// EXEC FORTXCG,PARM.GO='LET,EP=MAIN,SIZE=MAX-12K'
//FORT.SYSIN DD *
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION COOR(1000),KELEM(5000),KBOUND(2000),KBNDPT(300),
     V          INTMAT(6),AMAT(25000),USOL(1000),RHSD(1000),
     V          IWORK(1000),WORK(8000),DEPTH(200),BORDER(200),CONTLN(16),
     V          IINPUT(100),RINPUT(300),IUSER(100),USER(1000)
      COMMON /CACTL/  IELEM,ITYPE,ISOORT,INPELM,ICOUNT,IFIRST,NOI,ISTART
     V       /CPLOTC/ SCALE,NCONT,MCEL,MCEL,ITURN,IMARK,NUMD
C
C
      COOR(1)   = 1000
      KELEM(1)  = 5000
      AMAT(1)   =25000
      WORK(1)   =  8000
      KBOUND(1)= 2000
      IWORK(1)  = 1000
      INTMAT(1)=    6
      KBNDPT(1)=  300
      USOL(1)   = 1000
      RHSD(1)   = 1000
      DEPTH(1)  =  200
      BORDER(1)=  200
      USER(1)   = 1000
      IUSER(1)  =  100
      IINPUT(1)=  100
      RINPUT(1)=  300
C
C
      NRUNS = 10
      DO 2 I=1,NRUNS
C
      CALL INCOPY(BORDER,DEPTH,N,OMEGA,RATIO)
      CALL INTAKE(IINPUT,RINPUT,BORDER,I,N,RATIO)
      CALL MESSDO(I,KELEM,COOR,KBOUND,KBNDPT,IWORK,WORK,
     V            IINPUT,RINPUT,3000,ODC)
      IF (I.EQ.1)
     VCALL COMMAT(2,KELEM,KBOUND,INTMAT)
      CALL LOCAL(DEPTH,USER,COOR,N,OMEGA)
      CALL AUSER(IUSER,USER,COOR,OMEGA)
      CALL MATRIX(-1,AMAT,COOR,KELEM,KBOUND,WORK,IUSER,USER,INTMAT,IWORK
     V           )
      CALL BUSER(IUSER,USER)
      CALL RHSIDE(-1,RHSD,COOR,KELEM,KBOUND,WORK,IUSER,USER,AMAT,USOL)
      CALL SOLVE(3,AMAT,USOL,RHSD,INTMAT,KBOUND)
      CALL GRAPH(NUM1,AUX2,USOL,COOR,KBNDPT)
      CALL INPLOT(USOL,KBNDPT,CONTLN,1,1,2,2,2)
      NCONT = 2*NRUNS
      CALL PLOTC(1,KOUTP,COOR,KBNDPT(5)-6,KBNDPT,USOL,30D0,26D0,
     V           CONTLN,11)
      CALL INPLOT(USOL,KBNDPT,CONTLN,2,1,2,2,2)
      CALL PLOTC(2,KOUTP,COOR,KBNDPT(5)-6,KBNDPT,USOL,30D0,26D0,
     V           CONTLN,11)
C
2     CONTINUE
C
C      "END OF THE MAIN PROGRAM"
C
      STOP
      END
C
C
C     " THEN : THE REQUIRED SUBROUTINES "
C
C
C        SUBROUTINE INCOPY(BORDER,DEPTH,N,OMEGA,RATIO)
C
C         THESE DATA ARE TAKEN FROM THE " SIDE MODEL "
C         USING THE BACKGROUND MEMORY.
C
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION BORDER(1),DEPTH(1)
C
      READ(10,*) N1,OMEGA,RATIO
      READ(10,*) (DEPTH(I),I=1,N1)
      READ(10,*) (BORDER(I),I=1,N1)
C
      N=N1-5
C
      RETURN
      END
C
```

JOB INITIATION & "TOP VIEW. MODEL"

as for Side View Model.

INCOPY : see lines 1560 → 1580

" N ": the number of nodes along the border
(5 after-positions are subtracted from the array length.)

```
      SUBROUTINE INTAKE(IINPUT,RINPUT,BORDER,I,N,RATIO)

         THE DATA FOR "IINPUT" & "RINPUT"
         ARE FETCHED FROM THE BACKGROUND MEMORY.

      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION IINPUT(1),RINPUT(1),BORDER(1)

      IF (I.EQ.1)
     VREAD(55,*) (IINPUT(J+5),J=1,51)
      IF (I.EQ.1)
     VREAD(55,*) (RINPUT(J+5),J=1,2)
            THE SIDE-BOUNDARY ("BORDER") IS READ:

            DO 11 J=1,N
                  RINPUT(2*J+6)=BORDER(J+5)
                  RINPUT(2*J+7)= 0D0
   11       CONTINUE
            FOLLOWED BY READING THE TOP-BOUNDARY:

            DO 12 J=1,N
                  K= N-J+6
                  RINPUT(2*J+2*N+6)=BORDER(K)
                  RINPUT(2*J+2*N+7)= 1D0
   12       CONTINUE

      IF REQUIRED THE ARRAY (RINPUT) MAY BE PRINTED BY:
      WRITE(6,*) (RINPUT(J),J=1,233)

      RETURN
      END

        SUBROUTINE LOCAL(DEPTH,USER,COOR,N,OMEGA)

      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION DEPTH(1),USER(1),COOR(1),AKK(300),CCG(300)
      AKK(1) = 3.0
      NPARMS = COOR(4)

      CALL PRINAL('LOCAL','USER',5+2*NPARMS,-1,USER,IDUMMY)
      CALL PRINAL('LOCAL','AKK',5+NPARMS,-1,AKK,IDUMMY)
      THIS CALL CHECKS THE SIZE OF ARRAYS "USER", "AKK".
      WRITE(6,*) N,(DEPTH(J),J=1,N)

      NBOUN = N
      GRAV = 1D0

      DO 111 I=1,NBOUN
            I5= I+5
      IF (DABS(DEPTH(I5)-DEPTH(I5-1)).LE.1D-6) GO TO 15
                        NO ITERATION IS REQUIRED.
       ELSE
       NEWTON-RAPHSON IS USED TO DETERMINE LOCAL WAVE-LENGTH:

            H = DEPTH(I5)
      (first) AK   = 1D0          ; AK = AKK((I-1)*5)+1)
            ITER = 0
    1       ITER = ITER+1

            IF (ITER.LE.20) GO TO 12
                  WRITE(6,11) ITER
                  FORMAT('0THE ITERATION FAILS: ITER .GE.',I3,//)
   11             GO TO 13

   12       DELTA = (OMEGA**2) - GRAV*AK*DTANH(AK*H)
            AK    = AK+DELTA/(GRAV*(DTANH(AK*H)+(AK*H)/
           V                  (DCOSH(AK*H)**2)))
            IF (DABS(DELTA).GE.1D-6) GO TO 1


       THE LOCAL WAVE-NUMBER HAS BEEN DETERMINED: "AK".
       LOCALLY THE FACTOR "CCG" IS:

      CCGL = GRAV*(5D-1)*(DSINH(AK*H)*DCOSH(AK*H)+(AK*H))/
           V                  (AK*(DCOSH(AK*H)**2))

            AN ALTERNATIVE:
         CL = DSQRT(GRAV*DTANH(AK*H)/AK)
        CGL = CL*(5D-1)*(1D0+2D0*AK*H/DSINH(2D0*AK*H))

       IF (I.LE.10) WRITE(6,*) H,AK,CCGL,OMEGA,N
```

SUBROUTINE "INTAKE" prepares the data required by MESH2D, using the coordinates, along the border of the side view Model and the Top view Model, which have been generated in the Side View Model.
For each variation of the Side View Model the corresponding border is used.

"border": the coordinates of the nodes along the border.

LOCAL: compare versus AUSER in the Side View Model.

① computation of the local wave-number along the border
lines 4650 → 4880

② calculation of cc_g
(see page -15-)
lines 4890 → 4990

③ using the property $\frac{\partial}{\partial y} = 0$, the values of k, cc_g are attributed to the nodal points having the same x-coordinate, thus obtaining the local values of the parameters "ccg" and "k" throughout the field, as required by element ELM301
(see page — —)
lines 5000 → 5040

④ If the depth has not changed, no computation is required: the values in the prior point are used
lines 5060 → 5110

⑤ The number of parameters, and the values of the parameters "AKK", "CCG" may be printed
line 5130

```
C
      IF (I.LE.10) WRITE(6,*) H,AK,CCGL,OMEGA,N
C
C     THE CONDITIONS ALONG THE LENGTH OF THE FIELD ARE ATTRIBUTED,
C     PER STEP, TO THE WIDTH OF THE FIELD.
   13    DO 14 J=1,5
                 AKK(5*I+J)    = AK
   14            CCG(5*I+J)    = CCGL
         GO TO 111
C     FOR CONSTANT DEPTH: LOCAL VALUES: SAME AS IN PRIOR POINT.
   15    DO 16 J=1,5
                 AKK(5*I+J)    = AKK(5*(I-1)+J)
   16            CCG(5*I+J)    = CCG(5*(I-1)+J)
C
  111 CONTINUE
C
C                    WRITE(6,*) NPARMS,AKK,CCG
C
C
      DO 222 I=1,NPARMS
         I5=I+5
         USER(I5)           = CCG(I5)
         USER(I5+NPARMS)    = CCG(I5)*(AKK(I5)**2)
  222    CONTINUE
         USER(5+2*NPARMS+1) = AKK(6)*2
         USER(5+2*NPARMS+2) = AKK(6)
         USER(5+2*NPARMS+3) = AKK(NBOUN*5+5)
C
      PI=4*DATAN(1D0)
      TPER   = (2D0*PI)/OMEGA
      WAVEP= GRAV*(TPER**2)/(2D0*PI)
      WAVE = 2*PI/AKK(6)
C
      WRITE (6,999) TPER,WAVEP,WAVE,AKK(6)
  999 FORMAT('0WAVE PERIOD                 = ',F12.6,/
     V       ' DEEP WATER WAVELENGTH       = ',F12.6,/
     V       ' EXISTING WAVELENGTH         = ',F12.6,/
     V       ' EXISTING WAVENUMBER K       = ',F12.6,//)
      RETURN
      END
        SUBROUTINE AUSER(IUSER,USER,COOR,OMEGA)
C
       IMPLICIT REAL *8(A-H,O-Z)
       DIMENSION IUSER(1),USER(1),COOR(1)
       COMMON /CACTL/ ID(7),ISTART
      DO 1 I=6,40
         IUSER(I) = 0
    1   CONTINUE
      NPOINT = COOR(4)
      NPARMS= NPOINT
       ISTART = 10
       IUSER(6) = 1
       IUSER(7) = 1
       IUSER(8) = 0
       IUSER(9) = 1
       IUSER(10)= 0
       IUSER(11) = 16
       IUSER(12) = 20
       IUSER(13) = 24
       IUSER(14) = 28
       IUSER(15) = 32
      LAST = 5 + 2*NPARMS
       IUSER(21)=-(LAST+1)
       IUSER(23)=-(LAST+2)
       IUSER(31)=-(LAST+3)
C     WRITE(6,*) USER
       RETURN
       END
C
        SUBROUTINE BUSER(IUSER,USER)
C
       IMPLICIT REAL *8(A-H,O-Z)
       DIMENSION IUSER(1),USER(1)
       COMMON /CACTL/ ID(7),ISTART
       IUSER(6) = 0
       IUSER(7) = 1
       IUSER(8) = 0
       IUSER(9) = 0
       IUSER(10)= 0
       RETURN
      END
```

⑤ The values of "cg" and of "$k^2 cc_g$" are stored in array "USER", as well as the parameters for the boundary conditions.
2k, k, k as required for ELM 301 resp ELM 302 and ELR 302
(See appendix 5)
lines 5160 → 5230

⑦ Some characteristic information for the problem is computed and printed
lines 5250 → 5340

"AUSER"
See lines 2200 → 2410
and appendix 5

"BUSER"
see lines 2450 → 2560
and appendix 5

```
C
      SUBROUTINE INPLOT(USOL,KBNDPT,CONTLN,ICHOIS,NUM1,NUM2,M1,M2)
C
      IMPLICIT REAL *8(A-H,O-Z)
      DIMENSION USOL(1),KBNDPT(1),CONTLN(1)
      COMMON/CPLOTC/ SCALE,NCONT,NCEL,MCEL,ITURN,IMARK,NUMD
C
      SCALE = 1D0
C
      NBN=KBNDPT(5)
       N =KBNDPT(1+NBN) - KBNDPT(0+NBN) + 1
      NCEL = N*2
       M =KBNDPT(2+NBN) - KBNDPT(1+NBN) + 1
      MCEL = M*2
C
        AMAX = 0D0
        AMIN =-AMAX
         NUS = USOL(3)
               IINIT = ICHOIS+5
      DO 1 J=IINIT,NUS,2
                   A = USOL(J)
         IF (A.GT.AMAX) AMAX=A
         IF (A.LT.AMIN) AMIN=A
    1 CONTINUE
         RANGE = AMAX - AMIN
         STEP  = RANGE/10
      DO 2 J=1,11
                   CONTLN(J+5) = AMIN + (J-1)*STEP
    2 CONTINUE
      WRITE(6,*) (CONTLN(J+5),J=1,11)
      RETURN
      END
        SUBROUTINE GRAPH(NUM1,NUM2,USOL,COOR,KBNDPT)
C
      IMPLICIT REAL *8 (A-H,O-Z)
      DIMENSION USOL(1),COOR(1),KBNDPT(1)
      REAL *4  FI1(100),FI2(100),ABSFIE(100)
      INTEGER *4 IARR(100)
      NBN = KBNDPT(5)
      NBOUN = KBNDPT(4+NBN) - KBNDPT(3+NBN) + 1
C
      DO 1 I=1,NBOUN
                   LOCUS = 2*KBNDPT(NBN-I)
      AID1 = USOL(4+LOCUS)
      AID2 = USOL(5+LOCUS)
C
      FI1(I) = AID1
      FI2(I) = AID2
      ABSFIE(I) = DSQRT(AID1**2 + AID2**2)
    1 CONTINUE
C
      CALL PREFIE(FI1,IARR,NBOUN,1,NBOUN)
      CALL PREFIE(FI2,IARR,NBOUN,1,NBOUN)
      CALL PREFIE(ABSFIE,IARR,NBOUN,1,NBOUN)
C
      RETURN
      END
//GO.SYSLIB DD DISP=SHR,DSN=ITA.PRON
//  DD DSN=SYS2.FORTLIB,DISP=SHR
//  DD DSN=SYS2.MOMSUBF,DISP=SHR
//  DD DSN=SYS2.PGMLIB2,DISP=SHR
//  DD DSN=ITA.PIET2,DISP=SHR
//  DD DSN=.WLBLVI.PROGR,DISP=SHR
//GO.FT01F001 DD DSN=ITA.PIET3,DISP=SHR
//GO.FT10F001 DD DSN=&&WBDEO.STORE(LINK1),DISP=SHR
//GO.FT55F001 DD DSN=&&WBDEO.STORE(N2DATA),DISP=SHR
//GO.FT56F001 DD DSN=&&WBDEO.USOL1,DISP=SHR
//GO.PLTDS DD DSN=&&WBDEO.BP,DISP=SHR
//GO.FT04F001 DD DSN=&&WBDEO.BC,DISP=SHR
//GO.SYSIN DD *
IN
   4   5   2   0
011110
   4   2   2   2   2
   300
   302
   302
   302
   302
IN
//
```

"INPLOT"
See lines 2570 → 2870

"GRAPH"
See lines 2880 → 3120

"DCL"
See lines 3130 → 3250

NUMERICAL DATA:
See lines 3200 → 3350

Appendix 5

Input for subroutines IUSER and USER

The input for the boundary conditions (the values of the parameters $\dot{a}$, $\dot{g}$ for $\frac{\partial \varphi}{\partial n} = -\dot{a}\varphi + \dot{g}$) is provided using arrays " USER " and " IUSER " (AFEP convention).

IUSER: array containing " integer " numbers.

USER: ,, " real " ,,

In order to save memory space these arrays operate in the following way:

- the position in array IUSER corresponds with a value of a boundary condition of a specific boundary.
- a shift in the positions in the arrays may be accounted for by using shift parameter ISTART (which is also useful when other values are also stored in IUSER).
- The number (" # ") in a position of array IUSER indicates the position(s) in array USER where the required value(s) is stored belonging to the specific parameter of the considered boundary. (That is: # acts as a "pointer")

The following convention is used:

< 0 : The value of the parameter is constant along the boundary. The value is stored in position " # " of array USER, and is to be used in all nodes belonging to the boundary.

= 0 : The value of the parameter is zero. (No values are stored in array USER.)

#>0 : The value of the parameter varies along the boundary. The value for each consecutive node is stored in consecutive positions of array USER, beginning at position "#" for the first node.

Apart from this information concerning the values of the parameters for the boundary conditions, array IUSER contains some administrative information as well concerning the structure of array IUSER itself.
Apart from that, it also contains a set of "switches" which indicate if a boundary has only zero-values for its parameters ("0") or not ("1").

If all the values of the parameters are zero, the switch is "off" ("0"), the contributions of the boundary element matrices and vectors do not have to be computed for that boundary (due to the natural boundary conditions of the finite element method).

The administrative section of IUSER describes the "lay-out": the number of boundaries involved; the number of parameters per boundary; the position of thefirst "pointer" per boundary.

It will be shown how this is done using the boundary conditions of the Side View Model as an example.

The parameter (a1, a2, g1, g2) of boundary "# " is denoted by: a1b#, a2b#, ...

ISTART = 10

A shift of 10 positions is indicated.

Apart from the standard 5 positions reserved for AFEP,

10 positions are used for the "administrative section":

- 5 positions containing the "switches", one for the field and four for the boundaries, indicating whether the element matrix is to be computed or not ("0").
- 5 positions with pointers for the first positions occupied by the parameters.

IUSER :

1	2	3	4	5
6	7	8	9	10
1	1	0	1	1
11	12	13	14	15
16	20	24	28	32
16	17	18	19	
0	0	0	0	
20	21	22	23	
0	9	0	-6	
24	25	26	27	
0	0	0	0	
28	29	30	31	
0	0	0	-7	
32	33	34	35	
0	0	-8	0	

field:

boundary 1

boundary 2

boundary 3

boundary 4

g1 g2 a1 a2

5 AFEP positions

" switches "

pointers per boundary

pointers to the positions in array USER, per parameter, per boundary.

Compare versus (2.10.1), (2.10.2), (3.07.1) and (3.07.2).

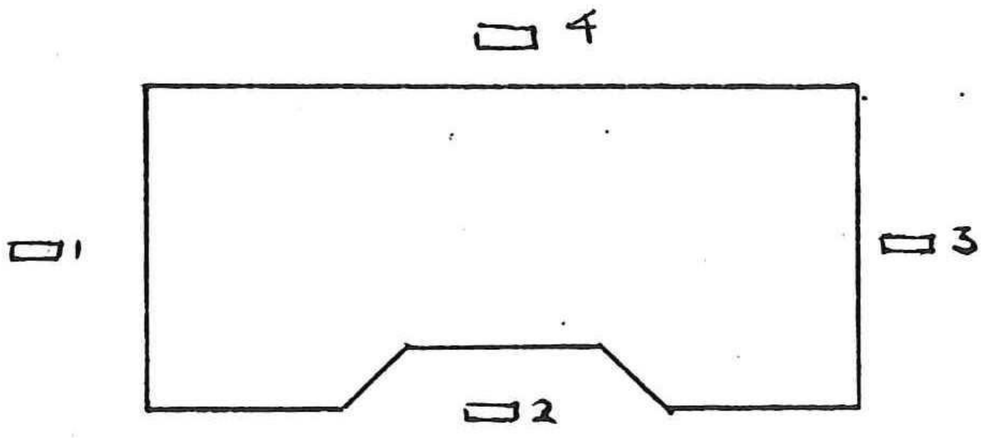
(AFEP requires a description of the parameter for the field condition. See the AFEP User Manual.)

USER :

position:

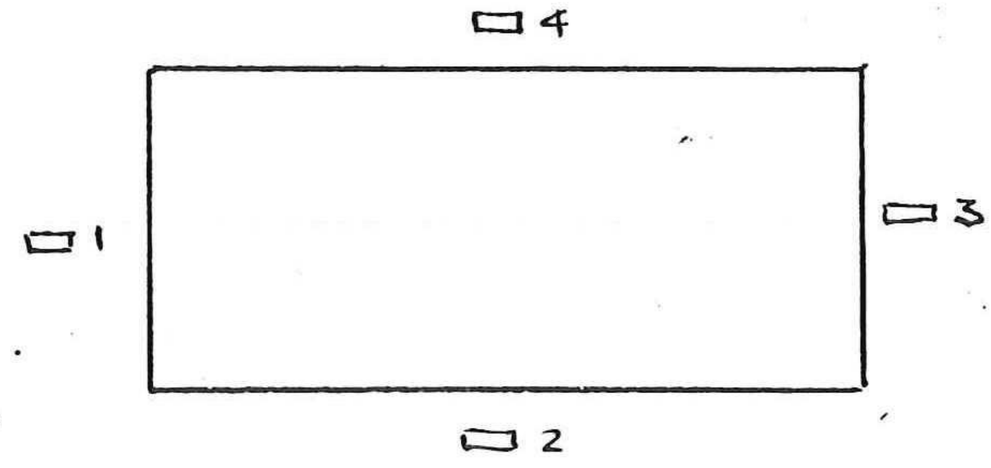
1	} AFEP reserved
2	
3	
4	
5	
6	a2b1 = k
7	a2b3 = k
8	a1b4 = ω^2 + grav
9	g2b1 for node 1
10	2
:	3
.	.
	.
	.

Presented in another form, for both the Side View Model and the Top View Model:



g1	g2	a1	a2	
0	0	0	0	field
$\omega \cdot f(i)$	0	0	k	1
0	0	0	0	2
0	0	0	k	3
0	0	$\frac{\partial^2}{\partial t^2}$	0	4

(See (2.10.1))



g1	g2	a1	a2	
0	0	0	0	field
ω	0	0	k	1
0	0	0	0	2
0	0	0	k	3
0	0	0	0	4

(See (2.10.2))

Appendix 6

Numerical aspects

In this appendix four facets will be dealt with:

1) verification of the location of the influx boundary,
2) verification of the steplength of the numerical computation,
3) (corollary of (2)): verification of the predictions of the Top View Model for some cases,
4) determination of a criterion for the significance of the difference in the predictions of the two models.

1) Position of the influx boundary

Only for the Side View Model is it required to check the location of the influx boundary, which has to be sufficiently far removed from the barrier. (This follows from the properties of the model, allowing a redistribution of the potential along the vertical).

CASE A has been computed to check the effect of the variation of the length of the wavefield upwave of the dam. The only difference between CASE A and CASE 3 is the value of ScL (See figure 6.03.3)). The length of the wavefield is one unit longer, upwave of the dam, in CASE A. This amounts to an increase of approximately 50% compared to CASE 3

The results of the two cases are:

Table A6.1.1 effects of varied steplength	
	$\| \varphi_t \|$
CASE A	(0.939 $\pm$ 0.002)
CASE 3	(0.941 $\pm$ 0.002)

The difference between the two predictions is not significant.

The influx boundary is indeed sufficiently far removed from the dam.

2) Steplength

The computational accuracy depends upon the size of the steplength; the computational costs also: accuracy improves with decrease of steplength; costs go down with increase of steplength.

The steplength varies throughout the field of computation, the mesh being refined where the variations of the potential will be greater: over the dam. See figure (7.01.2).

The size of the steplength is verified by comparing CASE B versus CASE 3 for the Side View Model, and by comparing CASE 3 versus computations done using a one dimensional model bases on the refraction - diffraction equation, made by N. Booy. (These computations also provide a possibility to verify the results of the Top View Model. This will be

discussed in paragraph (3)).

Because of requirements from the program concerning memory space in the program, the comparison of CASE 3 versus CASE B has been done using a double steplength for CASE B. The numerical error may be determined as follows.

For a linear model the relation between the errors in the results and the steplength may be described by:

$$E_1 - E_2 = C.(\Delta x_1^2 - \Delta x_2^2) \qquad (A6.2.1)$$

gives:

$$\alpha = C.\Delta x^2 \qquad (A6.2.2)$$

where: E_i : result of computation with steplength x_i

α : error in the result.

C : coefficient.

(See Kan & Segal, 1979)

In this case: $E_1 = .941$ (CASE 3, 6° item, table (7.04.1).)

$E_2 = .934$ (CASE 9)

$\Delta x_1 = .1$

$\Delta x_2 = .21$

Substitution in (7.03.1) gives:

$$C = .2025$$

Substitution in (7.03.2) gives:

$$\alpha = (0.2025).(0.1)^2 = 0.002$$

The calculation shows that the steplength of the Side View Model has a suitable size.

The stepsize in the Top View Model is verified by computation done using the one dimensional model made by N. Booy, based on the refraction - diffraction equation.

These computations have been done using the same number of nodes as well as with the double number of nodes (thus with half the steplength) as has been used in CASE 3.

The results are identical for these three cases:

Table A6.2.3

Presentation of the results of the comparison of the results of the Top View Model versus the model by N. Booy with variation of steplength.

Top View Model CASE 3	Booy model same steplength	Booy model half steplength
$\|\varphi_t\|$: .941	.941	.941

The comparison shows that the steplength of the Top View Model has a suitable size.

3 Verification of the Top View Model

It has been possible to check the values of $\|\varphi_t\|$, as predicted by the Top View Model, by using the model made by N. Booy to compute $\|\varphi_t\|$ for identical configurations. The resultes were identical.

The model by Booy has also been used to provide some values of $\|\varphi_t\|$ for those combinations of parameters which lead to doubtful results (See section 8.02).

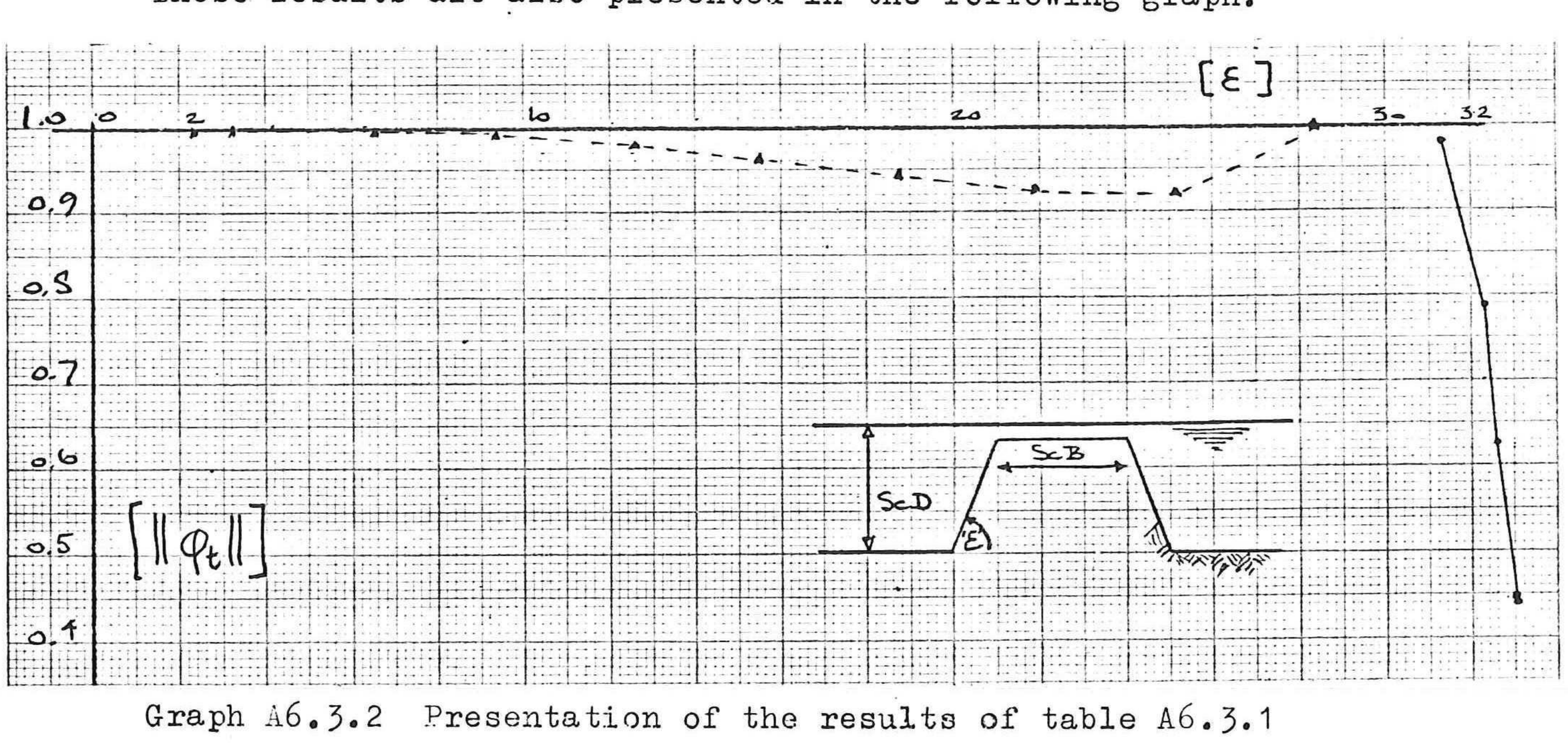
The results are presented in the following table.

Table A6.3.1

Values of $\|\varphi_t\|$ computed using the model made by N. Booy. Compare versus diagram 6.07.1 and table 7.04.1, CASE 2.

#	ScD	Ratio	ScB	ΔD	Δx	sigma	epsilon	$\|\varphi_t\|$
1	0.6	1÷12	0.6	.55	0.2	2.75	31.27	.980
2		1÷20		.570		2.85	32.30	.788
3		1÷24		.575		2.875	32.59	.624
4		1÷40		.585		2.925	33.25	.447
5		1÷60		.590		2.95	33.54	.440
6		1÷3		.40	2.0	0.2	2.27	.999

These results are also presented in the following graph.



Graph A6.3.2 Presentation of the results of table A6.3.1

The results of the computaions with the model made by N. Booy confirm the predictions made using the Top View Model, even in the behaviour which does not conform with the expectations, as described in section 8.02, for low values of Ratio. Further investigation will be required to explain these results.

4) Accuracy of the values of $\|\varphi\|$

As has been described in section (6.04), $\|\varphi_t\|$ has been selected as characteristic parameter of the comparative calculation.

In this paragraph the error in determining $\|\varphi_t\|$ will be determined. (For comparison this will also be done for $\|\varphi_c\|$ and $\|\varphi_r\|$; it will be seen that the uncertainty in these parameters is greater.)

The results of the computations for this study are presented in the form depicted in figure (7.02.3).

The roundoff error in the results is, for all cases, one unit of the fifth significant digit.

The uncertainty in $\|\varphi_t\|$

The uncertainty in $\|\varphi_t\|$ can be determined directly from the graphs of $\|\varphi\|$ which form the output of the computer program.

There is a slight fluctuation, a "fuzziness", in the values of $\|\varphi_t\|$ (See e.g. figure (7.02.3)).

This fuzziness is a measure of the uncertainty of the values of $\|\varphi_t\|$.

For all cases the fuzziness has been determined from the graphs. The results are displayed in Table (A6.4.1).

Table A6.4.1

Order of magnitude of the fuzziness of per computed case. (multiplied times 10^3).

CASE	1	2	3	4	5	6.	7	8	9	10	11	12
Side View M.	.4	.5	.6	.2	.2	.4	.5	.4	.7	.5	.5	1.
Top View M.	.3	.5	.5	.4	.6	.4	.4	.4	1.	.4	.5	2.

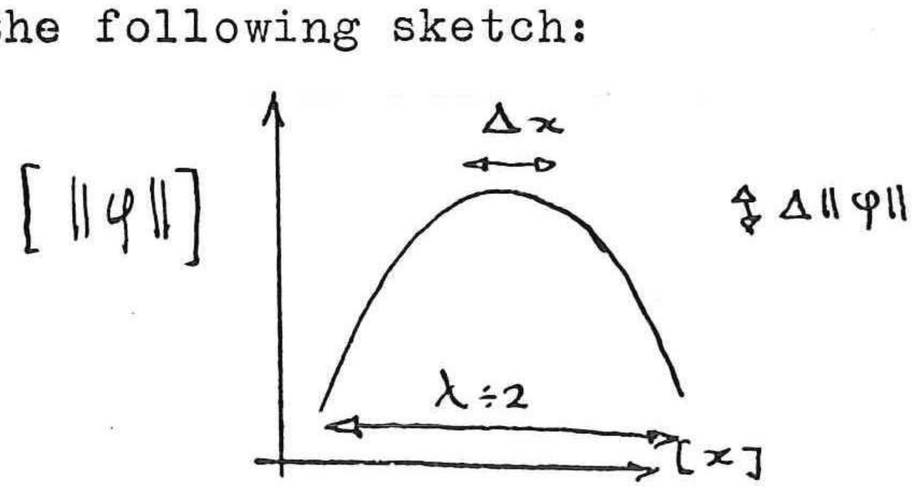
The uncertainty in $\|\varphi_i\|$ and $\|\varphi_r\|$

The values of $\|\varphi_i\|$ and $\|\varphi_r\|$ have to be determined from the maximum and minimum values of $\|\varphi\|$, as described in section (6.05).

Because the locations of the maxima and minima will not neccessarily coincide with the location of the nodes of the computation, an error will be introduced in the value of the maxima and minima.

An estimate of the size of this error may be made using the following sketch:

(A6.4.2)



$$\frac{\Delta\|\varphi\|}{\Delta x} \cong \frac{d \sin x}{dx} = \cos x$$

With this information the uncertainty in the value of at the location of the maxima and minima may be determined.

The steplength Δx can be expressed in the wavelength:

$$\Delta x \cong 0.11 \; ; \; \lambda \cong 3.71 \Rightarrow \Delta x \cong 0.030\,\lambda$$

The uncertainty is:

$$\Delta \| \varphi \| = \cos 0 - \cos \frac{\Delta x}{\lambda} = 2.10^{-3}$$

Error calculus shows that the error in the values of $\| \varphi_i \|$ and $\| \varphi_r \|$ will be described by:

$$\Delta \| \varphi_{i,r} \| = \frac{\Delta \| \hat{\varphi} \|}{\| \hat{\varphi} \|} + \frac{\Delta \| \check{\varphi} \|}{\| \check{\varphi} \|} \cong \frac{2.10^{-3}}{1.5} + \frac{2.10^{-3}}{0.5} \cong 5.10^{-3}$$

(Using the order of magnitude of $\| \hat{\varphi} \|$ and $\| \check{\varphi} \|$.)

It is clear that the uncertainty in $\| \varphi_t \|$ is smaller than in $\| \varphi_i \|$ and $\| \varphi_r \|$.

The uncertainty in $\| \varphi_t \| = 2.10^{-3}$ (Table A6.4.1)

error $\| \varphi_t \| = 2.10^{-3}$ (Paragraph 2)

This leads to the criterion to be used for the uncertainty of $\| \varphi_t \|$: $\Delta \| \varphi_t \| = 2.10^{-3}$.

As a consequence the error in the parameter of comparison will be:

$$\Delta \left(\frac{\| \varphi_t \|_T}{\| \varphi_t \|_S} - 1 \right) = \frac{\Delta \| \varphi_t \|_r}{\| \varphi_t \|_r} + \frac{\Delta \| \varphi_t \|_S}{\| \varphi_t \|_S} \cong 4.10^{-3}$$

⟶

Appendix 7

The listings of the elements of the element vectors and element matrices for both the Laplace equation and the refraction - diffraction equation are presented on the next pages.

A description of the derivation of the elements is presented in chapter 3.

Jun 13 04:04 1981 elm147 Page 1

```
      subroutine elm147(coor,work,iuser,user
      implicit real *8 (a-h,o-z)
      dimension coor(1),work(1),user(1),iuse
      common /cwork/ index1(100),index2(100)
     v       /cacti/ ielem,itype,isoort
  100 i1=index1(1)
      i2=index1(2)
      i3=index1(3)
      x1=coor(4+2*i1)
      x2=coor(4+2*i2)
      x3=coor(4+2*i3)
      y1=coor(5+2*i1)
      y2=coor(5+2*i2)
      y3=coor(5+2*i3)
      e1=y2-y3
      e2=y3-y1
      e3=y1-y2
      d1=x3-x2
      d2=x1-x3
      d3=x2-x1
      h=1d0/(2d0*dabs(e2*d3-d2*e3))
      a11=(e1*e1+d1*d1)*h
      a12=(e1*e2+d1*d2)*h
      a13=(e1*e3+d1*d3)*h
      a22=(e2*e2+d2*d2)*h
      a23=(e2*e3+d2*d3)*h
      a33=(e3*e3+d3*d3)*h
      work(6)=a11
      work(8)=a12
      work(10)=a13
      work(19)=a12
      work(20)=a22
      work(22)=a23
      work(30)=a13
      work(32)=a23
      work(34)=a33
      do 102 i=6,30,12
      do 102 j=1,5,2
      work(i+j+6)=work(i+j-1)
      work(i+j)=0d0
      work(i+j+5)=0d0
  102 continue
      return
      end
```

Jun 13 04:04 1981 elm143 Page 1

```
1               subroutine elm143(coor,work,iuser,user)
2               implicit real *8 (a-h,o-z)
3               real * 8 naam
4               dimension coor(1),work(1),iuser(1),user(1)
5               dimension x(2,2)
6               common /cwork/ index1(100),index2(100)
7              v       /cacti/ ielem,itype,isoort,inpelm,icount,ifirst,not,istart
8               data  naam /4helem/
9         c     *element matrices for the laplaceequation (boundaryconditions)
10        c      itype=105   linear conforming element (mixedtype)
11        c     *local parameters
12        c      x          array containing the coordinates
13        c      ih         help variable
14        c      h1,h2      help variables to compute h
15        c      h          factor h (see description)
16        c      iusrt      position in array user  (iuser(nstart))
17        c      a1,a2      factors a1,a2 (see description)
18        c     *computation of x and h
19              do 10 i=1,2
20                 ih=2*index1(i)
21                 x(i,1)=coor(ih+4)
22          10     x(i,2)=coor(ih+5)
23              h1=x(1,1)-x(2,1)
24              h2=x(1,2)-x(2,2)
25              h=dsqrt(h1*h1+h2*h2)
26              if(h.ne.0) goto 30
27        c       *error 1: volume is 0
28                 call ertrap(naam,1,15,itype,ielem,0)
29           30   if(ifirst.eq.1) goto 60
30        c       *ifirst=0, test inpelm and icount
31                 if(inpelm.eq.2) goto 40
32        c          *error 2: inpelm#2
33                    call ertrap(naam,2,20,itype,ielem,inpelm)
34           40    if(icount.eq.4) goto 50
35        c          *error 3: icount#4
36                    call ertrap(naam,3,20,itype,ielem,icount)
37           50    nstart=iuser(isoort+istart)
38        c          test nstart
39                 if(nstart.gt.5) goto 55
40        c          *error 4: |iuser(nstart)|<6
41                    call ertrap(naam,4,15,itype,ielem,0)
42             55  iusrt1 = iuser(nstart+2)
43                 iusrt2 = iuser(nstart+3)
44                 if (iusrt1) 1,2,3
45              1  isrt1 = iabs(iusrt1)
46                   a11 = user(isrt1)
47                   a12 = a11
48                                      n1 = isrt1
49                   go to 4
50              2    a11 = 0d0
51                   a12 = 0d0
52                                      n1 = 0
53                   go to 4
54              3   isrt1 = iusrt1
55                                      n1 = isrt1 + work(4)
56              4  if (iusrt2) 5,6,7
```

```
57          5  isrt2 = iabs(iusrt2)
58               a21 = user(isrt2)
59               a22 = a21
60                                   n2 = isrt2
61               go to 8
62          6    a21 = 0d0
63               a22 = 0d0
64                                   n2 = 0
65               go to 8
66          7  isrt2 = iusrt2
67                                   n2 = isrt2 + work(4)
68          8  call prinal(naam,5hiuser,nstart+3,6,dummy,iuser)
69               nu = n1
70               if(n2.gt.n1) nu = n2
71             call prinal(naam,4huser,nu,-7,user,dummy)
72       c                   the values of a1 and a2 have been attributed
73             if (iusrt1.ne.0.or.iusrt2.ne.0) go to 60
74             not=1
75             return
76       c
77        60   if (iusrt1.le.0) go to 11
78             a11 = user(isrt1)
79                                   isrt1 = isrt1 + 1
80             a12 = user(isrt1)
81        11   if (iusrt2.le.0) go to 22
82             a21 = user(isrt2)
83                                   isrt2 = isrt2 + 1
84             a22 = user(isrt2)
85       c
86        22   fac= h/24
87                    work(6)  =  fac*(6*a11+2*a12)
88                    work(7)  = -fac*(6*a21+2*a22)
89                    work(8)  =  fac*(2*a11+2*a12)
90                    work(9)  = -fac*(2*a21+2*a22)
91                    work(10) =  fac*(6*a21+2*a22)
92                    work(11) =  fac*(6*a11+2*a12)
93                    work(12) =  fac*(2*a21+2*a22)
94                    work(13) =  fac*(2*a11+2*a12)
95                    work(14) =  fac*(2*a11+2*a12)
96                    work(15) = -fac*(2*a21+2*a22)
97                    work(16) =  fac*(2*a11+6*a12)
98                    work(17) = -fac*(2*a21+6*a22)
99                    work(18) =  fac*(2*a21+2*a22)
100                   work(19) =  fac*(2*a11+2*a12)
101                   work(20) =  fac*(2*a21+6*a22)
102                   work(21) =  fac*(2*a11+6*a12)
103      c
104      c
105             return
106      c     *error messages
107      c       1:   volume of element = 0
108      c       2:   inpelm # 2
109      c       3:   icount # 2
110      c       4:   |iuser(iscort+istart)|<6
111      c       5:   sigma <= 0
112      c       6:   length of array iuser too small
```



```
113      c       7:   length of array user too small
114             end
```

Jun 13 04:04 1981 elr143 Page 1

```
      subroutine elr143(coor,work,iuser,user)
      implicit real *8 (a-h,o-z)
      real * 8 naam
      dimension coor(1),work(1),iuser(1),user(1)
      dimension x(2,2)
      common /cwork/ index1(10),index2(10)
     v       /cactl/ ielem,itype,isoort,inpelm,icount,ifirst,not,istart
      data naam /'elrns'/
c     *element vectors for the laplace equation (bounderycontions)
c      itype=104   linear conforming element  (neumann type)
c      itype=105   linear conforming element  (mixed type)
c     *local parameters
c      x          array containing the coordinates
c      in         help variable
c      h1,h2      help variables to compute h
c      h          factor h (see description)
c      iusrt      position in array user  (iuser(isoort+istart))
c      g          array containing the functionvalues (derivetives)
c     *computation of x and h
      do 10 i=1,2
         in=2*index1(i)
         x(i,1)=coor(in+4)
   10    x(i,2)=coor(in+5)
      h1=x(1,1)-x(2,1)
      h2=x(1,2)-x(2,2)
      h=dsqrt(h1*h1+h2*h2)
      if(h.ne.0) goto 30
c       *error 1: volume is 0
        call ertrap(naam,1,15,itype,ielem,0)
   30 if(ifirst.eq.1) goto 60
c       *ifirst=0, test inpelm and icount
      call prinal(naam,'iuser',isoort+istart,6,dummy,iuser)
         if(inpelm.eq.2) goto 40
c          *error 2: inpelm#2
           call ertrap(naam,2,20,itype,ielem,inpelm)
   40    if(icount.eq.4) goto 50
c          *error 3: icount#4
           call ertrap(naam,3,20,itype,ielem,icount)
   50    nstart=iuser(isoort+istart)
c          test nstart
         if(nstart.gt.5) goto 55
c          *error 4: |iuser(nstart)|<6
           call ertrap(naam,4,15,itype,ielem,0)
   55    iusrt1 = iuser(nstart)
         iusrt2 = iuser(nstart+1)
         if (iusrt1) 1,2,3
    1    isrt1 = iabs(iusrt1)
           g11 = user(isrt1)
           g12 = g11
                                  n1 = isrt1
         go to 4
    2      g11 = 0d0
           g12 = 0d0
                                  n1 = 0
         go to 4
    3    isrt1 = iusrt1
```

```
57                                      n1 = isrt1 + work(4)
58          4   if (iusrt2) 5,6,7
59          5   isrt2 = iabs(iusrt2)
60                g21 = user(isrt2)
61                g22 = g21
62                                      n2 = isrt2
63                go to 8
64          6     g21 = 0d0
65                g22 = 0d0
66                                      n2 = 0
67                go to 8
68          7   isrt2 = iusrt2
69                                     n2 = isrt2 + work(4)
70          8   call prinal(naam,5hiuser,nstart+3,6,dummy,iuser)
71                nu = n1
72                if(n2.gt.n1) nu = n2
73              call prinal(naam,4huser,nu,-7,user,dummy)
74        c                       the values of g1 and g2 have been attributed
75        c                        for the case g1,g2 constant
76              if (iusrt1.ne.0.or.iusrt2.ne.0) go to 60
77              not=1
78              return
79        c
80          60  if (iusrt1.le.0) go to 11
81                g11 = user(isrt1)
82                                        isrt1 = isrt1 + 1
83                g12 = user(isrt1)
84          11  if (iusrt2.le.0) go to 22
85                g21 = user(isrt2)
86                                        isrt2 = isrt2 + 1
87                g22 = user(isrt2)
88        c
89          22    h=h/6
90                      work(6)=h*(2*g11+g12)
91                      work(7)=h*(2*g21+g22)
92                      work(8)=h*(g11+2*g12)
93                      work(9)=h*(g21+2*g22)
94                 return
95        c       *error messages
96        c         1:    volume of element = 0
97        c         2:    inpelm # 2
98        c         3:    icount # 2
99        c         4:    |iuser(iscort+istart)|<6
100       c         5:    sigma <= 0
101       c         6:    length of array iuser too small
102       c         7:    length of array user too small
103               end
```

Jun 13 04:03 1981 elm300 Page 1

```
1               subroutine elm300(coor,work,iuser,user)
2       c
3               implicit real *8(a-h,o-z)
4               dimension coor(1),work(1),iuser(1),user(1),
5              v          t(3,3),sh(6,6),ind2(6)
6               common /cwork/ index1(100),index2(100)
7              v       /nwork/ inde(3)
8              v       /cactl/ ielem,itype,isoort,inpelm,icount,ifirst
9       c
10              npoint = coor(4)
11              nparms= npoint
12      c *** computation of e and delta in r2
13              do 600 i=1,8
14              do 600 j=1,8
15      600        t(i,j)=0
16              inde(1)=index1(1)
17              inde(2)=index1(3)
18              inde(3)=index1(4)
19              call elm301(coor,work,iuser,user)
20              do 610 i=1,6
21              do 610 j=1,6
22      610        sh(i,j)=work(j+6*i-1)
23              ind2(1)=1
24              ind2(2)=2
25              ind2(3)=5
26              ind2(4)=6
27              ind2(5)=7
28              ind2(6)=8
29              do 620 i=1,6
30              do 620 j=1,6
31                 i1=ind2(i)
32                 j1=ind2(j)
33      620        t(i1,j1)=t(i1,j1)+sh(i,j)
34              inde(1)=index1(1)
35              inde(2)=index1(2)
36              inde(3)=index1(3)
37              call elm301(coor,work,iuser,user)
38              do 710 i=1,6
39              do 710 j=1,6
40      710        sh(i,j)=work(j+6*i-1)
41              ind2(1)=1
42              ind2(2)=2
43              ind2(3)=3
44              ind2(4)=4
45              ind2(5)=5
46              ind2(6)=6
47              do 720 i=1,6
48              do 720 j=1,6
49                 i1=ind2(i)
50                 j1=ind2(j)
51      720        t(i1,j1)=t(i1,j1)+sh(i,j)
52              inde(1)=index1(1)
53              inde(2)=index1(2)
54              inde(3)=index1(4)
55              call elm301(coor,work,iuser,user)
56              do 810 i=1,6
```

```
57              do 810 j=1,6
58        810      sh(1,j)=work(j+6*1-1)
59              ind2(1)=1
60              ind2(2)=2
61              ind2(3)=3
62              ind2(4)=4
63              ind2(5)=7
64              ind2(6)=8
65              do 820 i=1,6
66              do 820 j=1,6
67                 i1=ind2(i)
68                 j1=ind2(j)
69        820      t(i1,j1)=t(i1,j1)+sh(i,j)
70              inde(1)=index1(2)
71              inde(2)=index1(3)
72              inde(3)=index1(4)
73              call elm301(coor,work,iuser,user)
74              do 910 i=1,6
75              do 910 j=1,6
76        910      sh(i,j)=work(j+6*i-1)
77              ind2(1)=3
78              ind2(2)=4
79              ind2(3)=5
80              ind2(4)=6
81              ind2(5)=7
82              ind2(6)=8
83              do 920 i=1,6
84              do 920 j=1,6
85                 i1=ind2(i)
86                 j1=ind2(j)
87        920      t(i1,j1)=t(i1,j1)+sh(i,j)
88              do 940 i=1,6
89              do 940 j=1,8
90                work(j+8*i-8)=t(i,j)/2
91         940  continue
92              return
93        c *** error messages
94        c *** 1: volume of element equals zero
95              end
```

Jun 13 04:03 1981 elr300 Page 1

```
1             subroutine elr300(coor,work,iuser,user)
2       c
3             implicit real *8(a-h,o-z)
4             dimension coor(1),work(1),iuser(1),user(1),
5            v          x(3,2),b(2),f1(4)
6             common /cwork/  index1(100),index2(100)
7            v       /cactl/  ielem,itype,isoort
8         100 do 110 i=1,2
9                 in=2*index1(i)
10                x(i,1)=coor(in+4)
11                x(i,2)=coor(in+5)
12        110     b(i)= user(9+index1(i))
13            p=dsqrt((x(2,1)-x(1,1))*(x(2,1)-x(1,1))+
14           v        (x(2,2)-x(1,2))*(x(2,2)-x(1,2)))
15            fac= p/6
16            f1 = user(3)
17            f2 = user(9)
18      c                user(3)& user(9) contain  "f1" resp "f2"
19            work(6) = fac* (2*b(1)+b(2))*f1
20            work(7) = fac* (2*b(1)+b(2))*f2
21            work(3) = fac* (b(1)+2*b(2))*f1
22            work(9) = fac* (b(1)+2*b(2))*f2
23            return
24            end
```

```
      subroutine elm301(coor,work,iuser,user)
c
      implicit real*8(a-h,o-z)
      dimension coor(1),work(1),iuser(1),user(1),
     v          e(3,2),x(3,2),ind(5),s(3,3),g(3),h(3)
      common /cacti/ ielem,itype,isoort,inpelm,icount,ifirst,not,istart
     v      /nwork/index1(3)
      data ind/1,2,3,1,2/, naam/4helem/
      nparms = coor(4)
      icount=6
      do 10 i=1,3
         in=2*index1(i)
         x(i,1)=coor(in+4)
         x(i,2)=coor(in+5)
         g(i)= user(index1(i)+5)
c              this section of "user" contains: c*cg.
   10    h(i)= user(index1(i)+5+nparms)
c              this section of "user" contains: c*cg*k*k.
      do 20 i=1,3
         e(i,1)=x(ind(i+1),2)-x(ind(i+2),2)
   20    e(i,2)=x(ind(i+2),1)-x(ind(i+1),1)
      delta=e(1,2)*e(3,1)-e(1,1)*e(3,2)
      if (delta) 22,21,25
c *** error 1: volume is 0
   21 call ertrap(naam,1,15,itype,ielem,0)
      return
   22 delta=-delta
   25 nm1=icount+1
      nm2=3*(icount-1)
      ik =icount*icount+5
      do 35 i=6,ik
   35    work(i)=0
      fac1= (g(1)+g(2)+g(3))/(6*delta)
      fac2=-delta/120
      do 110 i=1,3
      do 110 j=1,i
         if (i.eq.j) goto 100
         s(i,j)=(e(i,1)*e(j,1)+e(i,2)*e(j,2))*fac1
         s(j,i)=s(i,j)
         goto 110
  100    s(i,j)=(e(i,1)*e(j,1)+e(i,2)*e(j,2))*fac1
  110    continue
      s(1,1)=s(1,1)+(6*h(1)+2*h(2)+2*h(3))*fac2
      s(2,2)=s(2,2)+(2*h(1)+6*h(2)+2*h(3))*fac2
      s(3,3)=s(3,3)+(2*h(1)+2*h(2)+6*h(3))*fac2
      s(2,1)=s(2,1)+(2*h(1)+2*h(2)+1*h(3))*fac2
      s(3,1)=s(3,1)+(2*h(1)+1*h(2)+2*h(3))*fac2
      s(3,2)=s(3,2)+(1*h(1)+2*h(2)+2*h(3))*fac2
      s(1,2)=s(2,1)
      s(1,3)=s(3,1)
      s(2,3)=s(3,2)
      do 200 k=1,2
         k1=k*nm1-nm2
         do 200 i=1,3
         do 200 j=1,3
            ih=2*(i*6+j)+k1
```



```
  200       work(ih)=s(i,j)
            icount=3
         return
          end
```

Jun 13 04:04 1991 elm302 Page 1

```
1               subroutine elm302(coor,work,iuser,user)
2               implicit real *8 (a-h,o-z)
3               real * 8 naam
4               dimension coor(1),work(1),iuser(1),user(1),
5              v          x(2,2),b(2)
6               common /cwork/ index1(100),index2(100)
7              v       /cactl/ ielem,itype,isoort,inpelm,icount,ifirst,not,istart
8               data  naam /4helem/
9               nparms = coor(4)
10        c     *element matrices for the berkhoff equation (boundaryconditions)
11        c      itype=302   linear conforming element (mixedtype)
12        c     *local parameters
13        c      x          array containing the coordinates
14        c      d          array containing the local values of c*cg*q
15        c      ih         help variable
16        c      h1,h2      help variables to compute h
17        c      h          factor h (see description)
18        c      iusrt      position in array user  (iuser(nstart))
19        c      a1,a2      factors a1,a2 (see description)
20        c     *computation of x and h
21              do 10 i=1,2
22                 ih=2*index1(i)
23                 x(i,1)=coor(ih+4)
24                 x(i,2)=coor(ih+5)
25           10    b(i) = user(5+index1(i))
26        c        in these positions of "user" the values of "c*cg" are stored.
27        c        see: subroutine "local".
28                h1=x(1,1)-x(2,1)
29                h2=x(1,2)-x(2,2)
30                h=dsqrt(h1*h1+h2*h2)
31                if(h.ne.0) goto 30
32        c         *error 1: volume is 0
33                  call ertrap(naam,1,15,itype,ielem,0)
34           30   if(ifirst.eq.1) goto 60
35        c       *ifirst=0, test inpelm and icount
36                  if(inpelm.eq.2) goto 40
37        c           *error 2: inpelm#2
38                    call ertrap(naam,2,20,itype,ielem,inpelm)
39           40     if(icount.eq.4) goto 50
40        c           *error 3: icount#4
41                    call ertrap(naam,3,20,itype,ielem,icount)
42           50     nstart=iuser(isoort+istart)
43        c           test nstart
44                  if(nstart.gt.5) goto 55
45        c             *error 4: |iuser(nstart)|<6
46                      call ertrap(naam,4,15,itype,ielem,0)
47              55  iusrt1 = iuser(nstart+2)
48                  iusrt2 = iuser(nstart+3)
49                  if (iusrt1) 1,2,3
50               1  isrt1 = iabs(iusrt1)
51                    a11 = user(isrt1)
52                    a12 = a11
53                                  n1 = isrt1
54                    go to 4
55               2    a11 = 0.0
56                    a12 = 0.0
```

```
57                                      n1 = 0
58                go to 4
59           3    isrt1 = iusrt1
60                                      n1 = isrt1 + work(4)
61           4    if (iusrt2) 5,6,7
62           5    isrt2 = iabs(iusrt2)
63                a21 = user(isrt2)
64                a22 = a21
65                                      n2 = isrt2
66                go to 8
67           6    a21 = 0d0
68                a22 = 0d0
69                                      n2 = 0
70                go to 8
71           7    isrt2 = iusrt2
72                                 n2 = isrt2 + work(4)
73           8    call prinal(naam,5hiuser,nstart+3,6,dummy,iuser)
74                nu = n1
75                if(n2.gt.n1) nu = n2
76                call prinal(naam,4huser,nu,-7,user,dummy)
77           c                    the values of a1 and a2 have been attributed
78           c                                 for the case: a1,a2 constant
79                if (iusrt1.ne.0.or.iusrt2.ne.0) go to 60
80                not=1
81                return
82           c
83             60 if (iusrt1.le.0) go to 11
84                a11 = user(isrt1)
85                                      isrt1 = isrt1 + 1
86                a12 = user(isrt1)
87             11 if (iusrt2.le.0) go to 22
88                a21 = user(isrt2)
89                                      isrt2 = isrt2 + 1
90                a22 = user(isrt2)
91           c
92             22 fac= h/24
93                         work(5)  =  fac*(6*a11*b(1)+2*a12*b(2))
94                         work(7)  = -fac*(6*a21*b(1)+2*a22*b(2))
95                         work(8)  =  fac*(2*a11*b(1)+2*a12*b(2))
96                         work(9)  = -fac*(2*a21*b(1)+2*a22*b(2))
97                         work(10) =  fac*(6*a21*b(1)+2*a22*b(2))
98                         work(11) =  fac*(6*a11*b(1)+2*a12*b(2))
99                         work(12) =  fac*(2*a21*b(1)+2*a22*b(2))
100                        work(13) =  fac*(2*a11*b(1)+2*a12*b(2))
101                        work(14) =  fac*(2*a11*b(1)+2*a12*b(2))
102                        work(15) = -fac*(2*a21*b(1)+2*a22*b(2))
103                        work(16) =  fac*(2*a11*b(1)+6*a12*b(2))
104                        work(17) = -fac*(2*a21*b(1)+6*a22*b(2))
105                        work(18) =  fac*(2*a21*b(1)+2*a22*b(2))
106                        work(19) =  fac*(2*a11*b(1)+2*a12*b(2))
107                        work(20) =  fac*(2*a21*b(1)+6*a22*b(2))
108                        work(21) =  fac*(2*a11*b(1)+6*a12*b(2))
109          c
110          c
111                return
112          c     *error messages
```



```
113          c       1:   volume of element = 0
114          c       2:   inpelm # 2
115          c       3:   icount # 2
116          c       4:   |iuser(isoort+istart)|<6
117          c       5:   sigma <= 0
118          c       6:   length of array iuser too small
119          c       7:   length of array user too small
120                end
```

Jun 13 04:04 1981 elr302 Page 1

```
1               subroutine elr302(coor,work,iuser,user)
2               implicit real *8 (a-h,o-z)
3               real * 8 naam
4               dimension coor(1),work(1),iuser(1),user(1)
5               dimension x(2,2),b(2)
6               common /cwork/ index1(100),index2(100)
7              v       /cacti/ ielem,itype,isoort,inpelm,icount,ifirst,not
8               data naam /5nelrns/                                    istart
9               nparms = coor(4)
10        c     *element vectors for the berknoff equation (boundaryconditions
11        c      itype=302   linear conforming element  (mixed type)
12        c     *local parameters
13        c      x          array containing the coordinates
14        c      ih         help variable
15        c      h1,h2      help variables to compute h
16        c      h          factor h (see description)
17        c      iusrt      position in array user  (iuser(isoort+istart))
18        c      g          array containing the functionvalues (derivatives)
19        c     *computation of x and h
20              do 10 i=1,2
21                 ih=2*index1(i)
22                 x(i,1)=coor(ih+4)
23                 x(i,2)=coor(ih+5)
24          10     b(i) = user(5+index1(i))
25        c                 this section of "user" contains: c*cg.
26                 h1=x(1,1)-x(2,1)
27                 h2=x(1,2)-x(2,2)
28                 h=dsqrt(h1*h1+h2*h2)
29                 if(h.ne.0) goto 30
30        c          *error 1: volume is 0
31                    call ertrap(naam,1,15,itype,ielem,0)
32           30    if(ifirst.eq.1) goto 60
33        c          *ifirst=0, test inpelm and icount
34                 call prinal(naam,5hiuser,isoort+istart,6,dummy,iuser)
35                    if(inpelm.eq.2) goto 40
36        c             *error 2: inpelm#2
37                       call ertrap(naam,2,20,itype,ielem,inpelm)
38           40       if(icount.eq.4) goto 50
39        c             *error 3: icount#4
40                       call ertrap(naam,3,20,itype,ielem,icount)
41           50       nstart=iuser(isoort+istart)
42        c             test nstart
43                    if(nstart.gt.5) goto 55
44        c             *error 4: |iuser(nstart)|<6
45        c              call ertrap(naam,4,15,itype,ielem,0)
46              55    iusrt1 = iuser(nstart)
47                    iusrt2 = iuser(nstart+1)
48                    if (iusrt1) 1,2,3
49               1    isrt1 = iabs(iusrt1)
50                       g11 = user(isrt1)
51                       g12 = g11
52                                          n1 = isrt1
53                       go to 4
54               2       g11 = 0d0
55                       g12 = 0d0
56                                          n1 = 0
```

```
57                go to 4
58          3    isrt1 = iusrt1
59                                   n1 = isrt1 + work(4)
60          4   if (iusrt2) 5,6,7
61          5   isrt2 = iabs(iusrt2)
62                g21 = user(isrt2)
63                g22 = g21
64                                   n2 = isrt2
65                go to 8
66          6     g21 = 0d0
67                g22 = 0d0
68                                   n2 = 0
69                go to 8
70          7   isrt2 = iusrt2
71                                   n2 = isrt2 + work(4)
72          8   call prinal(naam,5hiuser,nstart+3,0,dummy,iuser)
73                nu = n1
74                if(n2.gt.n1) nu = n2
75              call prinal(naam,4nuser,nu,-7,user,dummy)
76        c                        the values of g1 and g2 have been attributed
77        c                         for the case g1,g2 constant
78              if (iusrt1.ne.0.or.iusrt2.ne.0) go to 60
79              not=1
80              return
81        c
82           60   if (iusrt1.le.0) go to 11
83                 g11 = user(isrt1)
84                                   isrt1 = isrt1 + 1
85                 g12 = user(isrt1)
86           11   if (iusrt2.le.0) go to 22
87                 g21 = user(isrt2)
88                                   isrt2 = isrt2 + 1
89                 g22 = user(isrt2)
90        c
91           22    fac=h/6
92                       work(6)= fac*(2*g11*b(1)+g12*b(2))
93                       work(7)= fac*(2*g21*b(1)+g22*b(2))
94                       work(8)= fac*(g11*b(1)+2*g12*b(2))
95                       work(9)= fac*(g21*b(1)+2*g22*b(2))
96                 return
97        c       *error messages
98        c         1:    volume of element = 0
99        c         2:    inpelm # 2
100       c         3:    icount # 2
101       c         4:    |iuser(isoort+istart)|<6
102       c         5:    sigma <= 0
103       c         6:    length of array iuser too small
104       c         7:    length of array user too small
105                end
```