The Use of Tidal Current data in Marine Spill Simulations

R.J. Ogilvie
THE USE OF TIDAL CURRENT DATA IN MARINE SPILL SIMULATIONS.

by

R.J. Ogilvie

Engineers thesis
Coastal and Offshore Engineering Group
Department of Civil Engineering
Delft University of Technology

March 1986
1. INTRODUCTION

The Marine Spill Simulation Software Set (MS4) is being designed to simulate the behavior of an oil spill at sea. The oil spill will move, spread, and age. To calculate the movements of the oil spill, wind data, water current data, and wave data are needed. Water currents consist of tidal currents, wind caused currents, and oceanographic currents (the Gulf stream for instance). Tidal current data can be entered separately from the other current data types. Interpolation routines have to be developed to handle these tidal current data.

Tidal current data can come from different sources, for instance from navigational charts. Such data can be incomplete and a method is needed that makes these data useful for the simulation program. Tidal current data from navigational charts are often given only on spring and on neap tide, the simulation program needs this data every hour, so some kind of interpolation is needed.

When the spill simulation program uses spatially distributed input data, a two dimensional interpolation method is needed. These data are entered per spatial point and the spill simulation program will use the data in a point near the oil slick center or will interpolate the input data between points nearest to the location of the oil slick center.
The following chapters describe the following items:

Chapter 2 gives information about MS4, its structure, its possibilities, and its limitations. More information can be found in the MS4 User's Handbook (Massie, 1985).

Chapter 3 describes the time interpolation method used for tidal current data. By means of a least square method ellipses are fit in the source data.

Chapter 4 describes the use of spatially dependent data.

Chapter 5 describes the nearest point method, a method that reduces the simulation run time when the simulation uses the data on the point nearest to the oil slick center.

Chapter 6 describes the two dimensional linear interpolation method that can be used to interpolate input data between three data points.

Chapter 7 discusses the nearest point method and the interpolation methods with a Poincaré wave and real tidal current data.

Chapter 8 gives some ideas about a special interpolation method that can eventually be used when the distance between the data points is too big for proper interpolation.
2. MARINE SPILL SIMULATION SOFTWARE SET.

MS4 is a software package for simulating the behavior of marine spills using the Digital Professional 300 series computers. The use of a specified type of computers is due to the employment of special machine-specific software. The MS4 model simulates the combined influences of three processes:
- Spreading: The extent of the spill.
- Transport: The movement of the spill, both on the sea surface and (if desired) while submerged.
- Aging: The changes in spill volume and properties.

MS4 can be used to evaluate the consequences of potential oil spills (Environmental Impact studies) and to plan combat strategies and guide combat operations during actual spills. This last use poses special requirements for operational speed and convenience in use.

MS4 offers several special features which make its use especially attractive during spill disasters:
- Convenient user interaction with the software is provided by means of menus and on-screen forms; the user doesn’t need to be a computer expert.
- The simulation can be tailored to suit the user’s individual needs for output, precision, and speed.
- The interrelations between spreading and aging are included.
- All environmental input data can be location dependent as well as time dependent as chosen by the user.
- Submergence of a spill can be predicted as well as its movement in a submerged state.
- Data for successive simulation runs can be edited from existing data for previous runs.
- Multiprogramming features of the Professional Computer are utilized to achieve maximum overall operational speed.
A wide variety of user selected input and output data units and conventions are provided.

- Data for some 200 types of crude oil and oil products is provided in a data file, accessed simply by specifying a name.
- Reversed transport - to 'backtrack' and localize the origin of a spill - is provided for.
- The simulation of soluble or insoluble chemical spills is foreseen and provided for.
- For convenience in documenting multiple runs, all data for each run is stored separately. A document file for each run stores its specific description.

A typical simulation run involves the successive use of several independent but related portions of the software set. These are listed and described briefly below.

Simulation SETUP
- The user defines conventions for the coordinate systems, as well as forms and units for data input and output.
- Since this information is usually run-independent, this program module needs only be used once at the start, and later only to modify the chosen settings.

Simulation DESIGN
- The individual run is defined and its document file is created.
- The user selects the process, associated data input options, and simulation control parameters commensurate with his needs for the particular run. The design from an earlier run may be copied or edited if desired.
- Data input options for wind, waves, tides, currents, and water properties include whether they are constant or time-dependent, as well whether they are location dependent or valid for the entire spill area.
Data INPUT
- The user enters data for each of the five environmental conditions listed above.
- If data is time dependent, the user may enter the data predictions for whatever times they may be available; the software performs the necessary interpolations.
- Vertical profiles of tides, currents, and water properties are needed if submerged transport has been selected.
- Data from previous runs may be copied or edited (using the standard editor) in order to speed up the overall simulation process.
- User-provided data is stored in the same form as it is provided. It is concurrently converted to the necessary form for the simulation computations.
- Spill material properties are taken from a file provided; only the name is needed.

Simulation RUN
The simulation computations themselves are carried out separately.
- The user can choose to do this under direct online control so that a limited list of intermediate results are available directly. This mode of operation will also be appropriate when interactively evaluating various spill combat strategies.
- An alternative is to carry out the computations in 'the background' without user intervention. This is convenient if he wishes to work on other runs (preparing additional input or examining previously generated output) for example.
- The duration of these simulation computations will vary from a few minutes to a few tens of minutes per run, depending upon the specific simulation design parameters chosen, and the specific computer configuration.
OUTPUT of Results
- The user can examine the output in tabular, or graphical form. Tables of output items at chosen intervals can be generated. This same data can be plotted (on the screen or on paper) as functions of time.
- Data on transport and spreading can be visualized as a plan showing the track and size of the spill at a scale convenient for transfer to a map.

Auxilliary programs make it possible to:
- examine the SETUP and DESIGN settings chosen for a given run,
- edit or add to the oil properties file.

MS4 does not currently include spill combat. Expansion to include this feature on an interactive basis is expected. The simulation of chemical spills and the computation of 'backtrack' transport has been provided for in the overall program architecture, but is not yet further implemented.

LIMITATIONS
MS4 does not attempt to match the computational power and sophistication of many other larger-scale computer simulation models. Each mathematical description of a process affecting an oil slick must fit into the MS4 program structure. The simulation of the behavior of an oil spill at sea progresses stepwise, using a time-stepping integration of various processes. Each subroutine that performs specific process calculations concerning (for example) tidal current interpolations in a given time interval, receives the needed data from the simulation program, but is limited in the use of memory and computer run time.
This means that, for instance, the use of a numeric model to describe the tidal currents (to calculate the tidal currents which are needed to calculate the translation of the oil slick) is out of the question; the tidal current data has to be entered. It is possible to interpolate this tidal current data between data points.
3. THE TIDAL CURRENT DATA ANALYSIS.

A surface oil slick will move due to influences of wind, water currents and waves (Kuipers 1980, v. Huijstee 1985, Lee 1980). Wind causes a movement of the oil slick relative to the sea-water which is also moving due to currents; wind is often, but not always, the most important transport factor. Water currents can be divided in tidal currents and residual currents (wind caused currents and oceanographic currents). In the North Sea, for instance, tidal currents are also an important factor, especially in case of a short duration simulation run.

Tides are caused by an interaction between the sea, the earth, the sun, and the moon. Different tide components can be distinguished, each component belonging to a specific kind of influence. For instance, the semidiurnal tide caused by the moon and the semidiurnal tide caused by the sun are the strongest tides in the North Sea, a small difference between their frequencies causes spring and neap tide (Defant 1961, Sverdrup 1942).

Since the tidal current is a function of time and of location, these data are entered for locations at sea where the tidal current values are known. When these data points are near to each other, the nearest point method can be used to select the appropriate simulation run data (Fig. 1). Otherwise a two dimensional interpolation method must be used (see sections 4,5 and 6).

The user can enter tidal current data in five different forms:
Fig. 1. An example of the calculation of the location of the oil slick, using the nearest point method. Only tidal current influences are accounted for. The dotted line is the solution calculated with the 'raw' source data and the solid line is the solution calculated with the analysed tidal current component coefficients (the duration is thirteen hours on both spring and neap tide).
1. A continuous time dependent record of velocities such as might be produced by an external (to MS4) tidal simulation model.

2. A continuous time dependent record of velocities such as might be provided by measurements.

3. Tidal current component coefficients as are produced by a tidal current analysis.

4. A record of velocities covering an average tidal current period.

5. Two records of velocities, one covering an average tidal current period near spring tide, the other near neap tide.

3.1 A continuous time dependent record of velocities. The data of this record must cover the entire simulation. MS4 will use the needed tidal current data directly from this record.

3.2 A continuous time dependent record of velocities such as might be provided by measurements. This record must cover 28 days with a time step of one hour. These data will be analysed. The velocities in the X and in the Y direction can be described as the sum of sine functions. Each sine function is a function of the time, and has a specific phase angle, amplitude, and angular velocity. The angular velocities are known, the phase angle and the amplitude are calculated using the data record. The interaction between the different tide components (each with a unique angular velocity) is small due to the fact that the vertical movements of the watersurface are relatively small compared to the waterdepth; the various tidal currents can be superimposed.
MS4 analyses the data with four tide components: M2, S2, K1 and O1. No constant current component is analysed, because a constant current component depends on influences, such as the average wind direction. The tide components are used to calculate the tidal current in the future, so such a constant current component can be faulty.

The tidal currents are described with:

\[
\mathbf{\mathbf{v}} = \sum_{i=1}^{4} \left( p_i \cos(w_i t) + q_i \sin(w_i t) \right)
\]

in which:

- \( \mathbf{\mathbf{v}} \) is the tidal current velocity vector, with two components, one in the \( \mathbf{X} \) direction and one in the \( \mathbf{Y} \) direction. \([\text{m/s}]\)
- \( p_i, q_i \) are vectors found with the least squares calculation. \([\text{m/s}]\)
- \( w_i \) is the angular velocity of the tide component \( i \). \([\text{rad/s}]\)
- \( t \) is the time. \([\text{s}]\)

The envelope of vector of one component has the form of an ellipse.

To calculate the tide component coefficients (the vectors \( p_i \) and \( q_i \)) two sets of equations are solved, one for the tidal currents in the \( \mathbf{X} \) direction and one for the tidal currents in the \( \mathbf{Y} \) direction. The vector \( \mathbf{\mathbf{v}} \) is a function of time and is given, the vectors \( p_i \) and \( q_i \) are constant in time and have to be calculated. There are more tidal current data available then is necessary to solve the equations, so a least squares method is used.
Each set of equations can be described by:

\[ A \vec{x} = \vec{b} \]

where:
\( A \) : is the coefficient matrix; it contains the values of the sine and cosine terms on the times the tidal current velocities are known.
\( \vec{x} \) : is the vector of the unknown ellipse coefficients, in the x direction or in the y direction.
\( \vec{b} \) : is the vector of known tidal current velocities, in the x direction or in the y direction.

Both sides are multiplied with the transposed matrix of \( A \):

\[ A^T A \vec{x} = A^T \vec{b} \]

The remaining set of equations can be solved by means of Gauss elimination; in this case also partial pivoting is used to improve numerical accuracy.

During the Gauss elimination process, horizontal vectors are subtracted from each other in floating point. It is possible that due to this floating point operation a linear dependency arises. This is fatal, because the set of equations will become unsolvable. This can be prevented when the value of the multiplier is as small as possible. (A horizontal vector is multiplied with the multiplier before the other vectors are reduced with the horizontal vector.) In this case partial pivoting is used; the horizontal vectors are exchanged so that the multiplier is as small as possible.

With the tide component coefficients (\( \vec{p}_i \) and \( \vec{q}_i \)) it is possible to calculate the tidal current on every desired moment. MS4 stores the calculated tidal current component coefficients on a file and creates a record of velocities which covers the entire simulation run.
3.3 Tidal current component coefficients as produced by a tidal current analysis.
The tidal current component coefficients calculated by MS4 or obtained from other sources can be entered directly. The user has to enter the coefficients in the X and in the Y direction; in both directions the cosine and the sine coefficient (see also page 10). MS4 creates with the tidal current component coefficients a record of velocities which covers the entire simulation run.

3.4 A record of velocities covering an average tidal current period.
The user has to choose between a diurnal or a semi-diurnal tide component and respectively between the O1 and the K1 or the M2 and the S2 tide component. The user has to enter a reference time and in case of a diurnal tide 25 current data and in case of a semi-diurnal tide 13 current data, both with a time step of one hour.

Then the tide component coefficients are calculated by means of a least squares method and MS4 creates a record of velocities covering the entire simulation run.

3.5 Two records of velocities, one covering an average tidal current period near spring tide, the other near neap tide. Data in this form are often published on navigational charts and tide atlases. When the user wants to enter these data he has to choose between diurnal or semi-diurnal tides (in case of diurnal tides MS4 calculates the coefficients of the O1 and K1 coefficients, in case of semi-diurnal tides respectively the M2 and S2 coefficients). The user has to enter the date and time of spring tide and in case of diurnal tides 25 current data on spring and on neap tide and in case of semi-diurnal tides 13 current data on spring and neap tide, both with a time step of one hour.
When the tide component coefficients are calculated without further action, the moment of maximal velocity can be faulty (Fig. 2). On spring tide the vertical displacements of the water surface are maximal, so the water currents will be maximal somewhere near the moment of spring tide. The amplitude of the vector $\mathbf{V}$ is maximal when the two tide component vector amplitudes are maximal. The time that the amplitude of the vector $\mathbf{V}$ is maximal can be calculated from the data ($\pm $ $\frac{\pi}{4}$ hours). This becomes more clear when the vector $\mathbf{v}$ (page 10) is written in a different way (see also section 3.6):

$$\mathbf{v}_i = a_i \cos (w_i t + \phi) + b_i \sin (w_i t + \phi)$$

where:
- $\mathbf{v}_i$ is a partial tidal current velocity vector, belonging to the component with the angular velocity $w_i$.
- $a_i$ is a vector with a magnitude equal to half the major axis and a direction coinciding with the major axis.
- $b_i$ is a vector with a magnitude equal to half minor axis and a direction perpendicular to $a_i$ (the difference is $\pi/2$ or $3\pi/8$ radians, depending on the turning direction).
- $\phi$ is the phase angle.

$a_i$ and $b_i$ are perpendicular to each other, so:

$$b_{ix} = n_i a_{iy}$$
$$b_{iy} = -n_i a_{ix}$$

where:
- $n_i$ is a real number.
Fig. 2. It is possible that the tidal component coefficients calculated with a least squares method represent faulty amplitudes and phase angles. This can happen because the data entered represent the current only near spring and neap tide and the faulty coefficients give a better fit to these incomplete data. This is illustrated with the envelopes of the tidal current (consisting of two components) in one direction.
The vector $\mathbf{v}$ has its maximum amplitude somewhere near spring tide. That maximum amplitude is equal to the amplitude of $\mathbf{a}$, so:

$$w_i \cdot ts + \phi = 0$$

where:

$ts$ : is the time that the amplitude of $\mathbf{v}$ has its maximum.

The six unknown values $a_{ix}$, $a_{iy}$ and $n_i$ are calculated with a least squares method. The set of equations is non-linear:

$$a_{lx} \cdot f_1 + n_1 \cdot a_{iy} \cdot f_2 + a_{2x} \cdot f_3 + n_2 \cdot a_{2y} \cdot f_4 = v_x$$

$$a_{ly} \cdot f_1 - n_1 \cdot a_{lx} \cdot f_2 + a_{2y} \cdot f_3 - n_2 \cdot a_{2x} \cdot f_4 = v_y$$

where:

$fi$ : are the cosine and sine terms, the time and phase angles are known.

$n_i$ : are the ratios between the amplitudes of $\mathbf{a}$ and $\mathbf{b}$ and lie in the range of $-1$ to $+1$.

The six unknown terms can be calculated using an iteration process. The sum of squares of the differences between the actual velocities and the calculated velocities must be minimized.
The vectors $\vec{p_i}$ and $\vec{q_i}$ can be calculated with $a_i$, $b_i$ and $\phi$:

$$\vec{p_i} = a_i \cos(\phi) + b_i \sin(\phi)$$

$$\vec{q_i} = a_i \cos(\phi + \pi) + b_i \sin(\phi + \pi)$$

Then MS4 creates a record of velocities covering the entire simulation run.

3.3 The conversion of the ellipse coefficients.

The envelope of the vector of one component has the form of an ellipse. It is possible to write the vector $\vec{v_i}$ in a way that the elliptic character is more clear. An ellipse is distinguished by a major axis, and a minor axis (perpendicular to each other). The major axis has a certain angle with the $x$ axis and a given length. The minor axis also has a given length (Fig. 3). The magnitudes of the major and the minor axis, the angle of the major axis with the $x$ axis, the phase angle and the turning direction give a better picture of the ellipse than the two vectors $\vec{p_i}$ and $\vec{q_i}$. Both ways of describing an ellipse are tested in section 7.
Fig. 3. An ellipse can be described with two vectors $p$ and $q$, or with $a$ and $b$. These two vectors are multiplied with a cosine and a sine function, which angles are a function of time, and a phase angle.
The vector $\vec{v}$ can be described in an alternative way:

$$\vec{v}_i = a_i \cos(w_i t + \phi) + b_i \sin(w_i t + \phi)$$

where:

- $\vec{v}_i$ is a partial tidal current velocity vector, belonging to the component with the angular velocity $w_i$.
- $a_i$ is a vector with a magnitude equal to half the major axis and a direction coinciding with the major axis.
- $b_i$ is a vector with a magnitude equal to half minor axis and a direction perpendicular to $a_i$ (the difference is $\pi/2$ or $3\pi/2$ radians, depending on the turning direction).
- $\phi$ is the phase angle.

To calculate the vectors $a_i$ and $b_i$, using $p_i$ and $q_i$, the magnitude of $\vec{v}_i$ is determined and its maximum and minimum are calculated:

$$\vec{v}_i = p_i \cos(f) + q_i \sin(f)$$

$$|\vec{v}_i|^2 = v_x^2 + v_y^2 = (p_x \cos(f) + q_x \sin(f))^2 + (p_y \cos(f) + q_y \sin(f))^2$$

$$\frac{d|\vec{v}_i|^2}{df} = (-p_x^2 - p_y^2 + q_x^2 + q_y^2)\sin(2f) + (p_x q_x + p_y q_y)2\cos(2f)$$

$|a_i|$ and $|b_i|$ are found when $\frac{d|\vec{v}_i|^2}{df} = 0$:

$$f = \frac{\omega \arctan[2(p_x q_x + p_y q_y)] + \omega m \pi}{p_x^2 + p_y^2 - q_x^2 - q_y^2}$$

where:

- $m = 0, 1, 2, \ldots$
- $f$ is the angle where $|\vec{v}_i|^2$ is maximal or minimal.
When $|\mathbf{a}|$ and $|\mathbf{b}|$ are calculated, the angle $\alpha$ of the main axis with the x axis is known.

The phase angle is calculated with the following equations:

$$\mathbf{c}_i = \mathbf{p}_i \cos(\phi) - \mathbf{q}_i \sin(\phi)$$

where:

- $\phi$ is the phase angle
- $c_{ix} = \mathbf{p}_x \mathbf{a}_y - \mathbf{p}_y \mathbf{a}_x$
- $c_{iy} = \mathbf{q}_y \mathbf{a}_x - \mathbf{q}_x \mathbf{a}_y$

These equations are found when $\mathbf{v}_i$ is calculated at $t = \phi/w_i$ with both ways of describing the ellipse.

The turning direction is assumed to be clockwise and when this is not the case, the magnitude of $\mathbf{b}_i$ is multiplied with $-1$.

The important ellipse coefficients used are:

$$A_i = |\mathbf{a}_i| : \text{is half the major axis of the ellipse.}$$
$$B_i = \pm |\mathbf{b}_i| : \text{is half the minor axis of the ellipse; a positive value indicates a clockwise turning direction.}$$
$$\alpha : \text{the angle of the main axis with the x axis.}$$
$$\phi : \text{the phase angle; when it is zero and the time is zero, } \mathbf{v}_i \text{ is equal to } \mathbf{a}_i.$$

The vectors $\mathbf{a}_i$ and $\mathbf{b}_i$:

$$\mathbf{a}_x = A \cos(\alpha)$$
$$\mathbf{a}_y = A \sin(\alpha)$$
$$\mathbf{b}_x = B \cos(\alpha - \pi/2)$$
$$\mathbf{b}_y = B \sin(\alpha - \pi/2)$$
3.4 The results.
To check the MS4 methods, computations were executed, using data provided by the KNMI (by dhr. Riepma of the Royal Dutch Meteorological Institute). The KNMI uses the analysis of current data to condense large sets of current data. The KNMI uses a more extensive analysis than MS4 uses (Table 1, source: KNMI - v.d. Veen).

The results of the analysis of 28 days of current data (time step one hour) agree with the results of the KNMI computations (source: KNMI - RIEH tide analysis of currents, Mr. Riepma). The small differences can be explained with the fact that the KNMI analyses eight tidal components (O1, K1, N2, M2, S2, K2, M4, and M6) and a constant current instead of the four components (O1, K1, M2, and S2). The results are listed in Table 2, Figure 4 gives an impression of the residual currents.

The method using data near the moment of spring and neap tide is tested with data prepared with the calculated tide component coefficients (Fig. 5). When data were used taken a few days past spring and neap tide, the error mentioned in Fig. 2 did not occur, the amplitudes are faulty of course but the phase angles are correct.
<table>
<thead>
<tr>
<th>Tidal component</th>
<th>Period</th>
<th>1970 Freq.</th>
<th>C.H. Depth</th>
<th>10 m 20 m</th>
<th>10 m 20 m</th>
<th>10 m 20 m</th>
<th>10 m 20 m</th>
<th>10 m 20 m</th>
<th>10 m 20 m</th>
<th>10 m 20 m</th>
</tr>
</thead>
<tbody>
<tr>
<td>M2</td>
<td>+28.984</td>
<td>N 35.7 107.6 20.6 97.4 32.3 95.8 28.4 95.5 31.0 103.7 29.3 106.3</td>
<td></td>
<td></td>
<td></td>
<td>36.2 127.8 29.4 117.0</td>
<td></td>
<td>36.1 125.8 29.5 126.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 31.6 169.9 25.2 157.0 28.1 162.1 23.2 166.6 26.2 170.2 23.9 171.8</td>
<td></td>
<td></td>
<td></td>
<td>31.9 193.8 28.6 182.8</td>
<td></td>
<td>31.8 191.8 24.7 191.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S2</td>
<td>+30.0000</td>
<td>N 5.1 175.9 3.7 170.1 12.0 175.5 9.7 184.4 13.7 171.3 13.2 176.7</td>
<td></td>
<td></td>
<td></td>
<td>6.9 170.3 5.0 164.3</td>
<td></td>
<td>6.8 168.0 5.0 171.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 4.6 261.2 3.2 240.7 12.3 251.8 10.8 261.5 12.0 244.5 12.6 245.4</td>
<td></td>
<td></td>
<td></td>
<td>6.3 248.5 5.5 236.6</td>
<td></td>
<td>6.1 246.6 5.5 244.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N4</td>
<td>+57.9682</td>
<td>N 5.9 268.1 5.1 225.5 4.3 210.3 3.0 223.2 2.9 215.3 2.6 226.5</td>
<td></td>
<td></td>
<td></td>
<td>4.5 280.5 3.9 257.8</td>
<td></td>
<td>4.8 281.4 4.1 278.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 1.7 44.6 5 103.3 1.2 238.7 3.2 206.8 3.2 253.6 3.3 17.5</td>
<td></td>
<td></td>
<td></td>
<td>6.6 296.7 7 311.8</td>
<td></td>
<td>6.9 292.8 8 389.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M2</td>
<td>+28.4397</td>
<td>N 4.7 98.6 3.3 97.5</td>
<td></td>
<td></td>
<td></td>
<td>4.2 171.2 3.6 176.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 3.8 7 2 5.7</td>
<td></td>
<td></td>
<td></td>
<td>2.1 210.4 1.7 206.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q1</td>
<td>+13.396</td>
<td>N 2.6 268.4 2.2 257.1 2.6 263.2 1.9 249.9 2.9 250.8 2.5 253.2</td>
<td></td>
<td></td>
<td></td>
<td>3.4 295.0 2.6 289.5</td>
<td></td>
<td>3.1 291.4 2.4 290.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 1.6 138.5 2 72.2 1.2 141.8 0.8 164.2 0.6 283.4 0.5 185.3</td>
<td></td>
<td></td>
<td></td>
<td>0.1 33.1 1.7 151.4</td>
<td></td>
<td>0.1 337.2 0.6 159.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>O1</td>
<td>+13.9430</td>
<td>N 2.9 107.2 2.3 104.7 3.0 142.8 2.9 154.2 1.2 127.2 1.0 133.3</td>
<td></td>
<td></td>
<td></td>
<td>3.3 99.7 2.8 99.5</td>
<td></td>
<td>3.2 98.0 2.8 103.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 1.0 327.3 5 308.3 0.8 222.3 0.3 266.6 1.1 270.9 0.7 318.2</td>
<td></td>
<td></td>
<td></td>
<td>0.2 260.4 0.4 244.4</td>
<td></td>
<td>0.2 257.1 0.4 243.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K1</td>
<td>+15.0410</td>
<td>N 2.9 107.2 2.3 104.7 3.0 142.8 2.9 154.2 1.2 127.2 1.0 133.3</td>
<td></td>
<td></td>
<td></td>
<td>3.3 99.7 2.8 99.5</td>
<td></td>
<td>3.2 98.0 2.8 103.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 1.0 327.3 5 308.3 0.8 222.3 0.3 266.6 1.1 270.9 0.7 318.2</td>
<td></td>
<td></td>
<td></td>
<td>0.2 260.4 0.4 244.4</td>
<td></td>
<td>0.2 257.1 0.4 243.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J1</td>
<td>+15.584</td>
<td>N 2.9 107.2 2.3 104.7 3.0 142.8 2.9 154.2 1.2 127.2 1.0 133.3</td>
<td></td>
<td></td>
<td></td>
<td>3.3 99.7 2.8 99.5</td>
<td></td>
<td>3.2 98.0 2.8 103.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 1.0 327.3 5 308.3 0.8 222.3 0.3 266.6 1.1 270.9 0.7 318.2</td>
<td></td>
<td></td>
<td></td>
<td>0.2 260.4 0.4 244.4</td>
<td></td>
<td>0.2 257.1 0.4 243.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>u2</td>
<td>+27.9682</td>
<td>N 1.6 242.0 2.1 204.6 2.6 206.2 2.1 229.6 8.2 230.4 7.2 231.8</td>
<td></td>
<td></td>
<td></td>
<td>3.1 273.2 2.7 269.6</td>
<td></td>
<td>3.1 284.1 2.7 282.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 1.8 244.2 2.5 240.3 3.0 220.8 1.8 190.3 5.9 280.8 5.8 279.0</td>
<td></td>
<td></td>
<td></td>
<td>3.2 291.8 1.5 297.9</td>
<td></td>
<td>2.9 296.1 1.5 310.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2</td>
<td>+29.5377</td>
<td>N 2.7 249.5 1.4 242.0</td>
<td></td>
<td></td>
<td></td>
<td>2.7 249.5 1.4 242.0</td>
<td></td>
<td>2.9 249.5 1.4 242.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 2.5 297.1 1.7 250.4</td>
<td></td>
<td></td>
<td></td>
<td>2.5 297.1 1.7 250.4</td>
<td></td>
<td>2.5 297.1 1.7 250.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26M2</td>
<td>+31.0159</td>
<td>N 1.1 75.9 6.5 55.5 3.0 101.6 8.1 115.6 5.5 114.0 6.6 23.9</td>
<td></td>
<td></td>
<td></td>
<td>1.1 78.9 1.2 118.6</td>
<td></td>
<td>1.1 79.8 1.0 118.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 1.0 134.9 7 92.8 2.3 154.7 1.0 123.2 1.2 84.8 0.4 63.7</td>
<td></td>
<td></td>
<td></td>
<td>0.4 194.2 0.6 167.3</td>
<td></td>
<td>0.4 193.8 0.5 141.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NS4</td>
<td>+58.984</td>
<td>N 1.9 318.5 1.6 291.0</td>
<td></td>
<td></td>
<td></td>
<td>1.9 318.5 1.6 291.0</td>
<td></td>
<td>1.9 318.5 1.6 291.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>E 2.2 238.4 0.4 309.3</td>
<td></td>
<td></td>
<td></td>
<td>2.2 238.4 0.4 309.3</td>
<td></td>
<td>2.2 238.4 0.4 309.3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. The results of a least square analysis of current data. The current enveloping ellipses are fitted into hourly means of measurements (6 per hour). In the table are the amplitude (cm/s) and the phase angle (degrees) given for the north and the east component of the current.
The tidal current component coefficients calculated with ELLIPS:

<table>
<thead>
<tr>
<th>Tide component</th>
<th>East component</th>
<th>North components</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>px:</td>
<td>py:</td>
</tr>
<tr>
<td>O1</td>
<td>1.98730</td>
<td>0.10180</td>
</tr>
<tr>
<td>K1</td>
<td>1.11427</td>
<td>1.08233</td>
</tr>
<tr>
<td>M2</td>
<td>0.76628</td>
<td>-2.95711</td>
</tr>
<tr>
<td>S2</td>
<td>1.12077</td>
<td>1.43402</td>
</tr>
</tbody>
</table>

The tidal current component coefficients calculated with RIEH:

<table>
<thead>
<tr>
<th>Tide component</th>
<th>East component</th>
<th>North components</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>px:</td>
<td>py:</td>
</tr>
<tr>
<td>O1</td>
<td>1.98912</td>
<td>0.10177</td>
</tr>
<tr>
<td>K1</td>
<td>1.11659</td>
<td>1.08214</td>
</tr>
<tr>
<td>M2</td>
<td>0.76679</td>
<td>-2.95721</td>
</tr>
<tr>
<td>S2</td>
<td>1.12077</td>
<td>1.43399</td>
</tr>
</tbody>
</table>

Table 2. The four tide component coefficients are calculated with 672 current data (28 days, time step one hour).
The currents in northward direction:

![Northward Currents](image)

The currents in eastward direction:

![Eastward Currents](image)

Fig. 4. The calculated current (---) compared to the actual current (-----). Also the difference between the currents (-----) is displayed (the residual current). The currents are calculated with eight tide components (O1, K1, N2, M2, S2, K2, M4 and M6) and a constant component.
The currents (in one direction) calculated with the analysed tidal current components.

Fig. 5. When the two data records are sampled a few days past spring and neap tide, the least squares method will give the actual tidal current components. If the data records are sampled near spring and neap tide and contain deviations from a linear two dimensional wave, amplitude and phase errors will occur (fig. 2). To prevent these errors, the phase angles are calculated beforehand, using the moment when the velocity vector has its maximum amplitude. To test this method, the $M_2$ and $S_2$ component coefficients are calculated using data records which are sampled a few days past spring and neap tide and the actual time of spring tide.
4. THE USE OF SPATIALLY DISTRIBUTED DATA.

Many environmental input data are dependent upon location \((X, Y)\) as well as time. Tidal current data are an example of this. An accurate analytical description of the spatial dependency of input data is often impossible and the use of a small computer limits the application of numeric models; for instance a numerical tide model of the North Sea would be inappropriate for MS4.

The time dependent and/or water depth dependent input data have to be entered one by one. The simulation program can search for the data given at the data point nearest to the location of the oil slick center or the simulation program can interpolate the input data between three data points nearest to the location of the oil slick.

4.1 The nearest point method.

One method to find the nearest point is to compute all the distances between the data points and the location of the oil slick and to search for the smallest distance; this calculation is needed during every simulation step. MS4 allows a maximum of twenty five data point locations for a given type of input. This can result in the nearest point method costing up to a hundred and fifty operations per time step (per data point two subtractions, two multiplications, one addition and one test). When four type of spatially distributed data are used and a simulation run time of four days using a time step of one minute is desired, the total number of operations, needed by this nearest point method, can run up to three and a half million.
This number of operations can be reduced using an alternative nearest point method. The method developed will need a relatively large amount of computer run time and memory before the simulation program is started. MS4 executes this preparatory program during the execution of the input program; the user will hardly notice this because the input program takes relatively much user-interactive run time, but less real computer run time. Subsequently the results can be used for different simulation runs.

Using this nearest point method, the simulation program will have to execute three to seven nearest point tests (one nearest point test needs two multiplications, three additions and one test). This will give an average of thirty operations per time step and an reduction of eighty percent relative to the other nearest point method. (An additional method of saving operations is to do this testing less frequently - every ten time steps, for example.)

4.2 Two dimensional linear interpolation.
The spatial interpolation, the second alternative, is a linear interpolation between three data points. With one dimensional linear interpolation a straight line is used, analogously two dimensional linear interpolation uses a flat plane. During the execution of the input program a preparatory program fits triangles between all data points. Three test values, used to confirm that the oil slick location is within the triangle, and interpolation coefficients are also calculated. This program needs the results from the preparatory nearest point program.
5. THE NEAREST POINT METHOD.

This method is designed to reduce the number of operations in the simulation run program. This is done by reducing the number of tests, by preparing these tests ahead of time and by using a special mathematical technique. Tests, additions, multiplications, divisions, roots etc. take successively more computer run time. Homogeneous coordinates are used to prevent the execution of division operations (literature: Newman et al., 1981). All the necessary calculations can be accomplished with these homogeneous coordinates; the only operations that are needed are additions and multiplications (see also chapter 5.1).

Each data point is the nearest point in a certain area. This area is bounded by perpendicular bisectors between the chosen data point and other data points (the perpendicular bisector between two points p and q is the line of which every point is equidistant to p and q). Not all the possible perpendicular bisectors are needed as borders; a few in the neighborhood of the data point will suffice (Fig. 6). The preparatory program has to minimize the number of borders of the nearest point area, to prepare the tests and to add a pointer to each test.

To minimize the number of nearest point area borders the preparatory program takes the following actions: Every data point gets a number and at first all possible borders are assumed. Then the number of borders of the nearest point area are minimized by choosing two perpendicular bisectors between the point which nearest point area borders are to be minimized and two other points. With these two perpendicular bisectors other boundaries are tested to determine if they are superfluous (Fig. 7). When at least two borders are parallel a different test is used. Of two parallel borders can be said if one is superfluous or not (Fig. 8).
Fig. 6. The nearest point areas of eleven tidal current data points in the German Bight.
Fig. 7. On the left the perpendicular bisector nr. 3 is a superfluous nearest point area border (of point $P$), on the right nr. 3 is not.

Fig. 8. On the left the perpendicular bisector nr. 2 is a superfluous nearest point area border (of point $P$), on the right both are not superfluous.
When all other borders are tested, a new pair of borders are taken. This is done with every point until the number of all nearest point area borders are minimized. Once a border is found to be superfluous it is not tested again (see also Fig. 10, chapter 4.2 and appendix 1).

When a perpendicular bisector intersects two borders of the nearest point area in the same point it is a superfluous border. This is possible when the locations are, for instance, square or hexagonally divided (Fig. 9). The method to eliminate such borders is simply to add a very small random number to all the coordinates before the eliminating process is started. This seems dangerous but since there is a maximum of six digits in a coordinate and the random number is a number between one and two, multiplied with the shortest distance between two points and divided by 10 million, it is impossible that a once square data grid point stays exactly square. This is important because the preparatory interpolation program uses the result of the preparatory nearest point program.

After the number of boundaries of the nearest point area has been minimized, a matrix remains which specifies the needed perpendicular bisectors (needed for a nearest point test). Per data point the tests are then created and stored. A test is a calculation of the scalar product of the coordinates of the oil slick and a perpendicular bisector, both in homogeneous coordinates. When this scalar product is positive the oil slick lies on the same side of the perpendicular bisector as the data point to which the test belongs. When this scalar product is negative this is not the case, and the pointer gives the number of the next data point to be used, as the new data point. In that case a new set of tests must be done.
Fig. 9. Perpendicular bisector nr. 3 is a superfluous nearest point area border of point P.
When the scalar product is zero, the oil slick lies exactly on the border; this is treated as a positive value. The distribution of the nearest point areas can be user checked by means of a graphical presentation on the screen (see also chapter 5.3 and appendix 1).

Due to the adding of small random values to the coordinates is it possible that the program handles double coordinates, but these double coordinates are faulty and before the preparatory program is run all the coordinates have to be tested (the random numbers are so small that the extra point will be invisible on the graphical presentation).

This method is unsuitable for less than three points, in that case has the test to be calculated in a different way.

5.1 Homogeneous coordinates.
A two dimensional point \((x, y)\) is represented by the homogeneous vector \([wx, wy, w]\). The 3-vector \([a, b, c]\) is converted back to ordinary coordinates \((a/c, b/c)\). Any nonzero scalar multiple of the homogeneous representation for a point represents the same two-dimensional point.

A line is represented by a column vector:

\[
L = \begin{bmatrix}
a \\
b \\
c
\end{bmatrix}
\]

The condition that a point \(v\) is on a line \(L\) is \(v \cdot L = 0\) (the inner product), when \(v \cdot L\) is positive, \(v\) lies on one side of \(L\), negative if it lies on the other side.

The line \(L\) between two points \(p\) and \(v\) is given by \(L = p \times v\) (the cross product).

The point \(v\) at the intersection of line \(L\) and \(M\) is given by \(v = L \times M\) (the cross product).
5.2 The calculation of the nearest point area borders.
The preparatory nearest point program starts to create a
two dimensional array of N x N numbers (there are N data
points). The number of borders (perpendicular bisectors)
around a data point is given by the numbers on the diagonal.
The array value (I,J) is equal to one when point number
I and point number J share a nearest point area border;
it is not equal to one if they don't. The program tests
all nearest point area borders or perpendicular bisectors
against each other to create this array.

The three coefficients of the perpendicular bisector
between the points number Q and R are calculated as
follows:

\[
\begin{align*}
AR &= X(R) - X(Q) \\
BR &= Y(R) - Y(Q) \\
CR &= \sqrt{(X(Q)^2 - X(R)^2 + Y(Q)^2 - Y(R)^2)}
\end{align*}
\]

where:
X(\_\_) and Y(\_\_) are the x and y coordinates.
A_, B_ and C_ are the coefficients of the perpendicular
bisector.

This is also done for the perpendicular bisectors
between points numbers Q and S and numbers Q and T.
Then the point of intersection of the perpendicular
bisectors of QR and QT is calculated:

\[
\begin{align*}
ART &= BR*CT - CR*BT \\
BRT &= CR*AT - AR*CT \\
CRT &= AR*BT - BR*AT
\end{align*}
\]

where:
A__, B__, C__ are the homogeneous coordinates of the point
of intersection.
This is also done for the perpendicular bisectors of QS and QT. Then the inner products are calculated of perpendicular bisector QR and point number Q and of perpendicular bisector QR and the point of intersection of perpendicular bisectors QS and QT (if the sign of C is negative then all the homogeneous coordinates of the point of intersection have to be multiplied by -1, otherwise the inner product sign would be wrong):

\[ DRQ = AR*X(Q) + BR*Y(Q) + CR \]
\[ DSRT = AS*ART + BS*BRT + CS*CRT \]

where:
DRQ and DSRT are the inner products.

This is also done with the perpendicular bisector QS and point number Q and the point of intersection between perpendicular bisectors QR and QT. If of each pair both inner products have opposite signs, then it is possible that the perpendicular bisector QT is a border of the nearest point area of point number Q (Fig. 5); this can be different in a later testing round.

When C is very small relative to A and/or B lie the coordinates of the point of intersection of two perpendicular bisectors in the infinitive (the coordinates are \((a/c, b/c)\)). In other words, the two perpendicular bisectors are parallel. Parallel nearest point area borders can be tested in a different way (Fig. 8):

First a line L is constructed through point number Q and point number R (since the two perpendicular bisectors are parallel, point S will also lie on this line):

\[ AL = Y(Q) - Y(R) \]
\[ BL = X(R) - X(Q) \]
\[ CL = X(Q)*Y(R) - Y(Q)*X(R) \]
Then the points of intersection of \( L \) and the two perpendicular bisectors are calculated (the cross product of the two lines). Next the inner products are calculated of perpendicular bisector QR and point Q and of QR and the point of intersection of line L and perpendicular bisector QS. This is also done for the other perpendicular bisector. If each pair of inner products has the same sign, then point number Q lies in between the two perpendicular bisectors and both are possible nearest point area borders (Fig. 8).

For each point Q all possible perpendicular bisectors between point Q and T have to be tested against every possible pair of perpendicular bisectors QR and QS (Fig. 10). If one nearest point area is found to be superfluous, it is not necessary to test it again, or to use it as a test for other nearest point area borders, of course; but still a large number of computations has to be done, and it is possible to achieve this in a reasonable time because the homogeneous coordinate technique is used.

5.3 The calculation of test parameters.
To prepare a nearest point area test all the coefficients of the nearest point area borders, or perpendicular bisectors, are needed. Then the inner product of a border and the point, whose nearest point area is tested, is calculated.

A nearest point area border:
\[ AZ = X(R) - X(Q) \]
\[ BZ = Y(R) - Y(Q) \]
\[ CZ = \frac{1}{2} \{ X(Q)^2 - X(R)^2 + Y(Q)^2 - Y(R)^2 \} \]

The inner product:
\[ F = X(Q) \times AZ + Y(Q) \times BZ + CZ \]
N: the number of data points
\( P(N,N) \): an integer array containing \( N \times N \) 1's
\( X(N) \): a real array, containing the X coordinates.
\( Y(N) \): a real array, containing the Y coordinates.

\[
\begin{align*}
\text{DO } Q &= 1 \text{ TO } N \\
\text{DO } R &= 1 \text{ TO } N-1 \\
\text{IF } R &= Q \text{ DO NEXT } R \\
\text{Calculate perpendicular bisector } Q-R \\
\text{DO } S &= R+1 \text{ TO } N \\
\text{IF } S &= Q \text{ DO NEXT } S \\
\text{Calculate perpendicular bisector } Q-S \\
\text{DO } T &= 1 \text{ TO } N \\
\text{IF } T &= Q \text{ OR } T = R \text{ OR } T = S \text{ DO NEXT } T \\
\text{Calculate perpendicular bisector } Q-T \\
\text{Calculate points of intersection:} \\
\text{RT: between } Q-R \text{ and } Q-T \\
\text{ST: between } Q-S \text{ and } Q-T \\
\text{when two line parallel, call subroutine and do next } T \\
\text{Calculate inproducts of } ST \text{ with } Q-R \text{ and } Q \text{ with } Q-R \\
\text{Calculate inproducts of } RT \text{ with } Q-S \text{ and } Q \text{ with } Q-S \\
\text{If both pair of inproducts have opposite signs, then} \\
P(Q,T) &= 0
\end{align*}
\]

Fig. 10. The structure diagram of the preparatory nearest point program. The matrix P gives the nearest point area structure.
If the above inner products sign is negative, the three border coefficients are all multiplied by -1. The three coefficients are the three test parameters and point number R (the perpendicular bisector was calculated between points number Q and R) is the pointer. The simulation program executes a number of tests for a nearest point area, when one of the tests is negative (a test is the inner product of the coordinates of the point whose area is being tested, and the three test parameters), the program can locate the next nearest point with the pointer.

5.4 Summary.
The preparatory nearest point program needs a set of coordinates of the data point locations. The points are numbered and the nearest point borders are minimized; the tests are calculated and the pointers are defined. All data, belonging to a point, are written to an indexed file; the index is the number of the point. An indexed file is a data storage method which allows random access to a file of data records on the basis of given key or index values.

The simulation program starts with reading the nearest point data from the file under index number 1. When one of the tests is negative, the pointer gives the new index and the program continues until all tests are greater or equal to zero. The program has found the nearest point area in which the center of the oil slick lies and can use the data given at the related data point. The nearest point method is applied when the user chooses this and when the interpolation method is applied and the oil slick center moves out of the area in which interpolation is possible. The use of an indexed file frees directly addressable memory for executable code and other data storage.
6. THE TWO DIMENSIONAL INTERPOLATION.

The most simple interpolation method is linear interpolation. In one dimension, a function is dependent on one variable and a straight line can be constructed between two function values. This straight line is an approximation of this function between the two variable values. When the two values are widely spaced, this approximation can be incorrect (Fig. 11).

In two dimensions, a flat plane can be constructed through three function values (the function depending on two variables). This plane has a simple mathematical description and the interpolated value can be written as a function of the oil slick coordinates, nine interpolation coefficients and the three function values. Of course, the oil slick must lie within a triangle with vertices at the coordinates of the three data points in which the function values are known (chapter 6.5, fig. 12, and literature: Chung, 1978). The nine interpolation coefficients are intermediate calculation results, these preparatory calculations make the interpolation during the simulation run faster (see also page 32).

Triangles can be fit into a collection of points in different ways. When the vertices of the triangles are selected with the criteria of the nearest point method, the area between all the points is as equally distributed as possible among the triangles (Fig. 13). If the perpendicular bisectors, which intersect with two boundaries of the nearest point area in the same point, are not deleted then triangles can overlap each other (Fig. 14).
Fig. 11. The interval between the two variables is too large for an accurate interpolation of \( f(x) \).
\[ f(x,y) = \sum_{i=1}^{3} F_i f(P_i) \]

\[ X_i = x_i - \frac{(x_1+x_2+x_3)}{3} \]
\[ Y_i = y_i - \frac{(y_1+y_2+y_3)}{3} \]

\[ a_1 = \frac{(X_2 Y_3 - X_3 Y_2)}{D} \]
\[ a_2 = \frac{(X_3 Y_1 - X_1 Y_3)}{D} \]
\[ a_3 = \frac{(X_1 Y_2 - X_2 Y_1)}{D} \]

\[ b_1 = \frac{(Y_2 - Y_3)}{D} \]
\[ b_2 = \frac{(Y_3 - Y_1)}{D} \]
\[ b_3 = \frac{(Y_1 - Y_2)}{D} \]

\[ c_1 = \frac{(X_3 - X_2)}{D} \]
\[ c_2 = \frac{(X_1 - X_3)}{D} \]
\[ c_3 = \frac{(X_2 - X_1)}{D} \]

\[ D = 2 \text{Area of the triangle} \]

\[ F_i = a_i + b_i x + c_i y \]

Fig. 12. Two dimensional linear interpolation.
Fig. 13. The two triangles on the right are fitted with the nearest point area matrix.

Fig. 14. Between the four points four triangles are fitted, because one unnecessary nearest point area border was not deleted.
The preparatory program fits triangles between the data points and calculates interpolation coefficients, tests and pointers. The triangles are fitted with the matrix filled by the preparatory nearest point program. This matrix contains zeros and non zeros. A zero at row M and column N indicates that between points M and N there are multiple nearest point area borders. When the matrix value is equal to one, only one nearest point border exist between the two points. The two points are vertex points of a triangle (see also Fig. 15, chapter 6.1 and appendix 1).

The preparatory program tests all the triangles at the edge of the total area. These triangles can be very long and thin because of an unlucky choice of data point locations or because of the addition of small random values to the data point coordinates. These triangles are deleted because they are not fit for interpolation (the vertex points lie too far apart) (see also Fig. 17, chapter 6.3 and appendix 1).

The edge of a triangle can adjoin another triangle or be on the perimeter of the total area. Each edge has one test and one pointer. If the scalar product of the location of the oil slick and the edge of the triangle (in homogeneous coordinates) is positive, then the oil slick and the center of area of the triangle lie on the same side of the edge of the triangle. If it is negative, the pointer indicates the next triangle or, when it is zero, to the use of the nearest point method (when the pointer is zero the oil slick is no longer in the area where triangles are specified). Since extrapolation can induce grave errors, the nearest point method must then be used (see also chapter 6.2, 6.4 and appendix 1).

The triangles can be user-checked by means of a graphical presentation on the screen.
6.1 The fitting of the triangles.
The preparatory program fits triangles between the data points and calculates interpolation coefficients, tests and pointers. The triangles are fitted with the matrix filled by the preparatory nearest point program. This matrix contains zeros and non zeros. A one at row M and column N indicates that a triangle edge runs between points M and N. To fit the triangles inbetween the data points the preparatory program starts with a point M and searches for a point N for which the matrix value \((M,N)\) is equal to one. Then it searches for a point P for which the matrix values \((M,P)\) and \((N,P)\) are both equal to one. Points M, N and P are the vertex points of a triangle number I. Next the program checks if this triangle number I is unique; because triangle number I can be created when the program starts with point M but also when it starts with point N or P. If the triangle is the only one with the vertex points number M, N and P then it is stored and the program searches for the next value of N. This is done for every point (Fig. 15).

6.2 The pointers.
The pointers are needed when the oil slick center moves from one triangle to another or out of the triangle area (the pointer equals to zero in that case). The simulation program uses these pointers to locate the adjacent triangle, in which three data values of the three data or vertex points are interpolated. To find the pointers the preparatory program chooses a triangle N then it searches for a triangle M which has an edge in common with triangle N. The number of triangle M is the pointer that belongs to the test for that specific edge of triangle N. If no triangle M is found, then stays the pointer value is set to zero; the edge in question lies on the perimeter of the entire data area.
N: number of data points
P(N,N): is an integer array, containing 1's and 0's, created by the preparatory nearest point program.
D(2N,3): is an integer array, containing 0's

TRI = 0

DO Q = 1 TO N
  DO R = 1 TO N
    IF R = Q DO NEXT R
    IF P(Q,R) = 0 DO NEXT R
  DO S = 1 TO N
    IF S = R OR S = Q DO NEXT S
    IF P(Q,S) = 0 OR P(R,S) = 0 DO NEXT S
  DO T = 1 TO TRI
    IF D(T,1) = Q AND D(T,2) = R AND D(T,3) = S DO NEXT S
    IF D(T,1) = Q AND D(T,2) = S AND D(T,3) = R DO NEXT S
    IF D(T,1) = R AND D(T,2) = S AND D(T,3) = Q DO NEXT S
    IF D(T,1) = R AND D(T,2) = Q AND D(T,3) = S DO NEXT S
    IF D(T,1) = S AND D(T,2) = R AND D(T,3) = Q DO NEXT S
    IF D(T,1) = S AND D(T,2) = Q AND D(T,3) = R DO NEXT S

TRI = TRI + 1
D(TRI,1) = Q
D(TRI,2) = R
D(TRI,3) = S

Fig. 15. The structure diagram of the triangle fitting program. The matrix D gives the data points of which triangle number TRI consists.
6.3 The triangles near the edge.
When a triangle has a pointer equal to zero, it lies near the edge of the triangle area. It is possible that this triangle has a flat shape (two vertex points lie too far apart); this makes the triangle unsuitable for data interpolation (the interpolation error is a function of the distance between the vertex points).
This kind of triangle is often created as a result of the addition of small random values to the coordinates of the data points. For instance, when the data originate from a numeric tidal current model with a square grid and a number of these data is used by MS4; the preparatory interpolation program creates these flat triangles near the edge of the triangle area (Fig. 16).

The program searches for a triangle with one pointer equal to zero and calculates the inside angles of this triangle. If one of these angles is smaller than a certain tolerance value then the triangle is unsuitable for interpolation, and the edge of the triangle (which pointer is zero) is deleted from the matrix created by the preparatory nearest point program. This is done for every triangle which has a zero pointer. Then the program starts again with the fitting of triangles in the modified matrix (of the preparatory nearest point program) (Fig. 17).

6.4 The tests.
To prepare three test of a triangle the preparatory interpolation program starts with a calculation of the lines through the vertex points (in homogeneous coordinates):

The line through the vertex points numbers Q and R:

\[
\begin{align*}
AL &= Y(R) - Y(Q) \\
BL &= X(Q) - X(R) \\
CL &= X(Q) \ast (Y(Q) - Y(R)) + Y(Q) \ast (X(R) - X(Q))
\end{align*}
\]
Fig. 16. Triangles fitted in a square grid. The use of random numbers introduces the flat triangles near the border of the triangle area; they can be seen clearly in the figure in the middle (a greater random number is added to the vertex point coordinates). The figure below shows the result when no random numbers are added.
P(N,N) is an integer array, created by the preparatory nearest point program.

D(TRI, 6) is an integer array, containing the numbers of the data point which are vertex points of triangle number TRI and the three pointers belonging to an edge.

TRI is the number of triangles

S = 0

DO T = 1 TO TRI

IF D(T,4) <> 0 AND D(T,5) <> 0 AND D(T,6) <> 0 DO NEXT T

Calculate the angles between the edges of triangle number T

If one edge is smaller than 0.087 radians do:

P(I,J) = 0 and S=1

I and J are the numbers of the vertex points (data points) on the perimeter of the total area.

If S <> 0, the program is sent back to the triangle fitting program and the pointer creating program, with the adapted matrix P

Fig. 17. The structure diagram of the program part with deletes triangles, near the edge of the triangle area, which have very small inner angles.

30b
Then it calculates the coordinates of the center of area of the triangle:

\[
CX = (X(Q) + X(R) + X(S))/3
\]
\[
CY = (Y(Q) + Y(R) + Y(S))/3
\]

The inner product of the coordinates of the center of area of the triangle and an edge of the triangle is:

\[
F = CX*AL + CY*BL + CL
\]

If the inner product sign is negative the three border coefficients are multiplied with -1. The three coefficients are the test that belong to a certain edge of a triangle. The simulation program calculates the inner product of the test and the coordinates of the slick center, when this is negative the slick center and the center of area of the triangle are on different sides of an edge of the triangle.

6.5 The interpolation coefficients.
The preparatory program calculates all the interpolation coefficients (Fig. 12).

The interpolation coefficients of a triangle are determined in terms of the local rectangular cartesian coordinates \((X, Y)\) with their origin at the center of area of the triangle. This triangle is identified with the global rectangular cartesian coordinates \((x, y)\); the following relationships exist:

\[
Xi = xi - (x1 + x2 + x3)/3
\]
\[
Yi = yi - (y1 + y2 + y3)/3
\]

where:
Xi, Yi are the local coordinates of a vertex point of the triangle.
xi, yi are the global coordinates of a vertex point of the triangle.
Now consider the expansion of a variable $U$ in the form:

$$U = k_0 + k_1X + k_2Y$$

This represents a linear variation of $U$ in both $X$ and $Y$ directions within the triangular element. The three constants $k_0$, $k_1$, and $k_2$, can be calculated with known values of $U$ at the three vertex points. The three constants can be calculated with matrix operations, the matrix operations are executed before the simulation run is started to save calculation time. These three equations and the expansion of $U$ give (literature: Chung, 1978):

$$U = F_1U_1 + F_2U_2 + F_3U_3$$

where:

$$F_i = a_i + b_iX + c_iY$$

and:

$$F_1 + F_2 + F_3 = 1$$

$$a_1 = (X_2Y_3 - X_3Y_2)/D$$

$$a_2 = (X_3Y_1 - X_1Y_3)/D$$

$$a_3 = (X_1Y_2 - X_2Y_1)/D$$

$$b_1 = (Y_2 - Y_3)/D$$

$$b_2 = (Y_3 - Y_1)/D$$

$$b_3 = (Y_1 - Y_2)/D$$

$$c_1 = (X_3 - X_2)/D$$

$$c_2 = (X_1 - X_3)/D$$

$$c_3 = (X_2 - X_1)/D$$

where:

$D = 2 \times \text{Area of the triangle.}$

To make it possible to use global instead of local rectangular cartesian coordinates, the coefficient $a_i$ is adapted:
ami = ai - bi*(x1 + x2 + x3)/3 - ci*(y1 + y2 + y3)/3

Then:

\[ U(x,y) = \sum_2^\infty Fi*Ui \]

where:

\[ Fi = ami + bi*x + ci*y \]

The preparatory interpolation program calculates all the coefficients ami, bi, and ci.

6.6 Summary.
The preparatory interpolation program needs a set of coordinates of the data points and an array (created by the preparatory nearest point program), which gives the locations of triangle edges between the data points. Then it fits triangles between these data points, calculates tests, pointers and interpolation coefficients. The triangles are numbered and all data, belonging to a triangle, are written to an indexed file; the index is the number of the triangle.

The simulation program starts with reading the triangle data from the file under index number 1. When one of the tests is negative, the pointer gives the new index and the program continues until all three tests are greater or equal to zero. The program has found the triangle in which the center of the oil slick lies and can interpolate the data given on the three vertex or data points.
7. TWO DIMENSIONAL INTERPOLATION OF TIDAL CURRENT DATA

As MS4 uses tidal current data it is necessary to have an idea about the consequences of using the nearest point method or the two dimensional interpolation method given a data point grid; also two different ways of interpolating tidal current data are possible; each method will have its own use of computer memory and run time as well as its own precision.

To obtain a general impression of interpolation errors, a Poincaré wave is used as an example and the errors are calculated. However, these results are of uncertain value compared to the real situation. To give an idea of the real situation, an example of the horizontal variability of the ellipse coefficients of the M2 tide is shown and some interpolation and nearest point method results are given.

7.1 Methods of tidal current interpolation.
It is possible to interpolate either the actual velocities at the data points or the coefficients of the enveloping ellipses. The precisions of both methods will be different; also the needed computer memory and run time will differ. The fastest and simplest way is to interpolate between the actual velocities on the data points.
The input program will store a long list of current data (possibly created with the ellipses) on a sequential file and the simulation run program will read from this file. This file is created during the execution of the input program by a "background" program. The computer executes this program when the "foreground" program (input program) waits for user actions (data input etc.). Every record will contain all the x and y components of the velocities in the area on a certain moment. The simulation run program will interpolate the x and the y components separately.

When the coefficients of the enveloping ellipses are interpolated, the simulation run program has to execute more interpolations (four per component), preliminary calculations of angles (two coefficients of each ellipse are angles), and the current calculation with the ellipses (six cosine/sine operations are necessary for each component) instead of two interpolations. This method will be interesting only if the results are much better.

The length of the major axis of the ellipse can simply be interpolated. The length of the minor axis of the ellipse is positive when the current turns clockwise and negative when the current turns counterclockwise. When values with alternating signs are interpolated, the turning direction will change somewhere in between the areas with opposite turning directions.
The interpolation of angles of the major axes of the ellipses with the x axis, needs preparatory calculations. When pi radians are added or substracted, the remaining angle is still the actual angle between the major axis of the ellipse and the x axis. Since angles need to be interpolated over the smallest angle possible, the simulation run program has to add or substract pi radians to the angles until the smallest possible interpolation angle is found. The angles are given by the preparatory ellipse fitting program between -pi/2 and +pi/2 radians.

When the angles are sorted it is easy to see that there are three possibilities of adding pi radians to one or more of the angles without guaranteeing the creation of a bigger interpolation angle (Fig. 18). The simulation run program calculates these six interpolation angles and takes the smallest. The phase angles of the ellipses whose main axes angles are increased with pi radians, are also increased with pi radians. When an angle value exceeds 2 pi radians, 2 pi radians are substracted (this is needed for the calculations for the interpolation of the phase angle).

Before the phase angles are interpolated a similar minimalization of the maximum interpolation angle is done; instead of pi radians, 2 pi radians are added.

Besides these two ways of interpolating tidal current data between data points, two other ways are possible:
Fig. 18. There are three possibilities of adding or subtracting pi radians to one of the angles, between a major axis and the x axis, without creating an identical or bigger smallest interpolation angle.
1. Instead of interpolating the x and the y components of the velocity, it is also possible to interpolate the magnitude and the angle of the velocity vector. The results of this interpolation are of varying quality and it costs more computer memory and run time (the angles have to be adapted), so this method is not applied.

2. Instead of interpolating the ellipse coefficients A, B, α, and ϕ, it is possible to interpolate the vectors \( \vec{p_i} \) and \( \vec{q_i} \). This method of interpolation is faster, because the preliminary calculations of the interpolation of angles is not necessary, but the results are of varying quality. An explanation this behavior can be found when a Poincaré wave is interpolated instead of real tidal currents. In case of a Poincaré wave, the coefficients A and B are solely dependent on y (A and B are sine/cosine functions of y), ϕ is linear dependent on x and α is constant (see section 7.3). The vectors \( \vec{p_i} \) and \( \vec{q_i} \) are dependent on both x and y in a non-linear way (so the interpolation error becomes larger).

7.2 The interpolation error.
The nearest point method is an interpolation method where the function is approximated with a constant. The maximum error that is introduced using the nearest point method is (literature: Chung, 1978):
\[ \sup |u(x,y) - \hat{u}(x,y)| \leq d \sup |D^1 u(x,y)| \]

where:
- \( \sup | | \) : is the absolute maximum.
- \( D^1 u(x,y) \) : is the first partial derivative of \( u(x,y) \) with respect to \( x \) and \( y \).
- \( d \) : is the distance between the data point and the location where the data is needed. When a square grid is used, and the spill stays inside the grid, the maximum value of \( d \) is \( \sqrt{2} \) times the grid distance.
- \( \hat{u}(x,y) \) : is the data value in the data point.
- \( u(x,y) \) : is the data value on the location \( x,y \).
- \( x \) and \( y \) must lie inside the nearest point area.

The maximum error that is introduced using the two dimensional linear interpolation method is (literature: Chung, 1978):

\[ \sup |u(x,y) - \hat{u}(x,y)| \leq \frac{t^2 \sup |D^2 u(x,y)|}{2} \]

where:
- \( D^2 u(x,y) \) : is the second partial derivative of \( u(x,y) \) with respect to \( x \) and \( y \).
- \( t \) : is the maximum distance between two data points. When a square grid is used, \( t \) is equal to \( \sqrt{2} \) times the grid distance.
- \( \hat{u}(x,y) \) : is the data value in the data point.
- \( u(x,y) \) : is the data value on the location \( x,y \).
- \( x \) and \( y \) must lie inside the area inside the triangle, formed by the 3 data point locations.

### 7.3 The Poincaré wave.
To obtain an idea about the precision of the nearest point method and the two interpolation methods, the interpolation errors are calculated assuming that the tidal wave is a Poincaré wave. This is a possibility to compare the three methods.
A Poincaré wave is a solution to a simplified set of equations, with special boundary conditions, which describe the water movements in the sea. This set of linear equations is (literature: Defant 1961, Pedlosky 1979):

**continuity:** \[ \frac{\partial H}{\partial t} + h \frac{\partial (u_x + u_y)}{\partial x} = 0 \]

**motion:** \[ \frac{\partial u_x}{\partial t} + g \frac{\partial H}{\partial x} - f u_y = 0 \]
\[ \frac{\partial u_y}{\partial t} + g \frac{\partial H}{\partial y} + f u_x = 0 \]

where:
- the x axis is parallel to a geographic parallel and eastwards is positive.
- the y axis is parallel to a meridian and northwards is positive.
- \( u_x \) : is the velocity in the x direction.
- \( u_y \) : is the velocity in the y direction.
- \( t \) : is the time.
- \( H \) : is the wave height.
- \( h \) : is the water depth.
- \( g \) : is the average acceleration of gravity.
- \( f = 2 \omega a \sin(\phi) \) : is the Coriolis parameter.
- \( \omega a \) : is the angular velocity of the earth rotation.
- \( \phi \) : is the latitude.

The following contributions are neglected:

- the friction forces.
- the convective acceleration terms.
- the horizontal and the vertical component of the tide generating force of the sun and the moon.

It is also assumed is that the variations of the water surface are small compared to the water depth.
It is possible to determine a harmonic solution of this set of equations for an infinitely long, canal with rectangular cross-section (parallel to the x axis with a width of L meters and a depth of ho meters, literature: Pedlosky 1979, Fig. 19 and 20):

\[
H = Ho*[(\cos(\gamma) - L*f*\sin(\gamma)]*\cos(\beta) \div n*pi*Cx
\]

\[
ux = Ho*[Co^2*\cos(\gamma) - L*f*\sin(\gamma)]*\cos(\beta) \div ho*Cx \div n*pi
\]

\[
uy = Ho*L*[f^2 + Co^2*n^2*pi]^2*\sin(\gamma)*\sin(\beta) \div ho*w*n*pi \div L^2
\]

The dispersion relation gives the relation between w and k:

\[
\frac{n^2*pi^2}{L^2} = \frac{w^2 - f^2 - k^2}{Co^2}
\]

where:

\( \gamma \) : is \( n*pi*y/L \)

\( \beta \) : is \( k*x - w*t + \varphi \)

\( \varphi \) : is an arbitrary wave phase.

Ho : is an arbitrary amplitude.

k : is the wave number, \( k = 2*pi/wave \text{ length} \).

n : is 1, 2, 3, ...

Co : is the phase speed in a non rotating channel.

Cx : is \( w/k \), the phase speed.

w : is the angular velocity of the tidal movement.

ho : is the still water depth.

L : is the canal width.

The first partial derivative of ux and uy with respect to x and y is necessary to calculate the error of the nearest point method (Fig. 21):
Fig. 19. The vertical displacements of a Poincaré wave, above for $n = 1$, below for $n = 2$. 
Fig. 20. The velocities of a Poincaré wave, for $n = 1$. Above the velocities in the $x$ direction, below the velocities in the $y$ direction.
Fig. 21. The first partial derivative, with respect to $x$ and $y$, of a Poincaré wave, for $n = 1$. Above the derivative of the velocity in the $x$ direction, below in the $y$ direction.
The second partial derivative of $u_x$ and $u_y$ is necessary to calculate the error of the interpolation of $u_x$ and $u_y$ (Fig. 22):

$$D_{1u}x = \frac{Ho\{k^2[Co^2\cos(\delta) - fL\sin(\delta)]\sin(\rho)\}}{ho \ Cx \ n\pi}$$

$$+ \frac{[-n^2\pi^2Co^2\sin(\delta) - f\cos(\delta)]\cos(\rho)}{L*Cx}$$

$$D_{1u}y = \frac{Ho*L* \ \left[ f^2 + Co^2 \pi^2 \right]}{ho\pi \ n\pi \ L^2}$$

$$\frac{n^2\pi^2\cos(\delta)\sin(\rho) + k\sin(\delta)\cos(\rho)}{L}$$

$$D_{2u}x = \frac{Ho\{k^2[Co^2\cos(\delta) - fL\sin(\delta)]\cos(\rho)\}}{ho \ Cx \ n\pi}$$

$$-2k\frac{[-n^2\pi^2Co^2\sin(\delta) - f\cos(\delta)]\sin(\rho)}{L*Cx}$$

$$+\frac{[-n^2\pi^2Co^2\cos(\delta) - f\pi\sin(\delta)]\cos(\rho)}{L^2*Cx}$$

$$D_{2u}y = \frac{Ho*L* \ \left[ f^2 + Co^2 \pi^2 \right]}{ho\pi \ n\pi \ L^2}$$

$$\frac{-n^2\pi^2\sin(\delta)\sin(\rho)}{L^2}$$

$$+\frac{2k\pi\sin(\delta)\cos(\rho)}{L}$$

$$-\frac{k^2\sin(\delta)\sin(\rho)}{L}$$
Fig. 22. The second partial derivative, with respect to x and y, of a Poincaré wave, for $n = 1$. Above the derivative of the velocity in the x direction, below in the y direction.
The second partial derivative of A and B with respect to x and y is necessary to calculate the error of the interpolation of the ellipse coefficients. The ellipse coefficients are used differently, here; the turning direction will depend on B and on A and no preparation of the angles is needed. The principle is the same, because the adding of pi radians to the angle of the main axis is the same as changing the sign of A. The interpolation of the two other ellipse coefficients is exact, because the angle of the major axis with the x axis is constant and the phase angle \( kx + \Phi \) is linear dependent on x (the second derivative is zero).

\[
\begin{align*}
A & = \frac{H_0\left(\cos(\gamma) - Lf\sin(\gamma)\right)}{h_0 \ C_x} \\
B & = \frac{H_0 L^2 \left[f^2 + \cos(2n\pi)\right] \sin(\gamma)}{h_0 \ w \ n \ \pi L^2} \\
D^2A & = \frac{H_0 \left[-n^2 \pi^2 \cos(\gamma) + n \pi f \sin(\gamma)\right]}{h_0 \ L^2 C_x} \\
D^2B & = -\frac{H_0 n \pi \left[f^2 + \cos(2n\pi)\right] \sin(\gamma)}{h_0 \ w \ L} 
\end{align*}
\]

The following Poincaré wave coefficients are chosen to calculate some numerical results:
\[ w = 0.141 \text{ E-03 [rad/s]} \quad (M2 \text{ tide}) \]
\[ f = 0.118 \text{ E-03 [rad/s]} \quad (54^\circ \text{ latitude}) \]
\[ L = 0.1 \text{ E+07 [m]} \]
\[ h_0 = 50 \text{ [m]} \]
\[ H_0 = 1 \text{ [m]} \]
\[ n = 1 \]
\[ C_0 = 22.147 \text{ [m/s]} \]
\[ C_x = 99.170 \text{ [m/s]} \]
\[ k = 0.142 \text{ E-05 [rad/m]} \]
\[ 2\pi/k = 4.43 \text{ E+07 [m]} \quad \text{(wave length in the x direction)} \]

The maxima:

\[ u_x = 0.751 \]
\[ D_{u_x} = 0.236 \text{ E-05} \]
\[ D_{u_{u_x}} = 0.895 \text{ E-11} \]
\[ D_{u_A} = 0.748 \text{ E-11} \]

\[ u_y = 0.850 \]
\[ D_{u_y} = 0.267 \text{ E-05} \]
\[ D_{u_{u_y}} = 0.101 \text{ E-10} \]
\[ D_{u_B} = 0.691 \text{ E-11} \]

The maximum grid distance \( D_{\text{max}} \) is calculated as follows:

The nearest point method:
\[ D_{\text{max}} = \frac{0.01u}{D_{u_{u}}\sqrt{2}} \]

The interpolation method:
\[ D_{\text{max}} = \frac{\sqrt{0.01u}}{D_{u_{u}}^2} \]
The Poincaré wave is a progressive wave in the x direction and a standing wave in the y direction. The wave length in the x direction is equal to $2\pi k$, the wave length in the y direction is equal to $2L/n$. When a maximum error of 1% of the maximum value which is interpolated, is allowed, the maximum grid distances (assuming a square grid) can be calculated and compared to the wave lengths:

<table>
<thead>
<tr>
<th>Dmax</th>
<th>ratio to $2\pi k$</th>
<th>ratio to $2L/n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>nearest point method:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$u_x$: $450\times 10^3$ [m]</td>
<td>.10 %</td>
<td>.23 %</td>
</tr>
<tr>
<td>$u_y$: $450\times 10^3$ [m]</td>
<td>.10 %</td>
<td>.23 %</td>
</tr>
<tr>
<td>interpolation method:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$u_x$: $290\times 10^4$ [m]</td>
<td>.65 %</td>
<td>1.45 %</td>
</tr>
<tr>
<td>$u_y$: $290\times 10^4$ [m]</td>
<td>.65 %</td>
<td>1.45 %</td>
</tr>
<tr>
<td>A: $315\times 10^4$ [m]</td>
<td>.71 %</td>
<td>1.58 %</td>
</tr>
<tr>
<td>B: $351\times 10^4$ [m]</td>
<td>.79 %</td>
<td>1.75 %</td>
</tr>
</tbody>
</table>

When the results are compared to the interpolation results of a one dimensional sine function:

$$u = A\sin(kx)$$

The maximal grid distances can be calculated as a ratio to $2\pi k$:

| nearest point method: | .32 % |
| interpolation method: | 2.25 % |
These results differ by a factor $\sqrt{2}$, with the results of the two dimensional methods compared to $2\times L/n$. This factor is introduced in the maximum distance between two data points, or in the maximum distance between a data point and the oil slick center when a square grid is used.

Conclusively can be said, that using the Poincaré wave as criterium, that the results of the two interpolation methods differ slightly, but that either of the interpolation methods are considerably better than the nearest point method. However, when instead of a 1% accuracy a 10% accuracy is allowed, the advantage of the interpolation methods will decline. The ratio between the maximum grid distances, of the interpolation methods and the nearest point method, will decline from 7 to 2.

7.4 The interpolation of tidal current data.
Figure 23 shows the M2 ellipses in a part of the North Sea (source: KNMI Mr. Riepma). These ellipses are fitted to current data by means of a least squares method. The current data are obtained from measurements between 10-20 meters water depth.

To obtain a global idea about the accuracy of the nearest point method and both interpolation methods, some calculations where made with different data points. (Fig. 24 - 29). The data points where taken from an area (Fig. 23), where the horizontal gradients of the ellipse coefficients are large; testing the interpolation methods in this area is a sort of worst case approach.

These pictures and other tests show that the nearest point method gives useful results with a grid distance up to approximately 10 km. and that both interpolation methods give useful results with a grid distance up to approximately 40 km.
The tide current enveloping ellipses, of the M2 component, in a part of the north sea. The scale of the map is approximately 1:850000, the scale of the velocities is approximately 1:25. The ellipse numbers correspond with the data point numbers used in the interpolation tests.
Fig. 24. Comparison of various interpolation methods to the actual ellipse. The results of the ellipse coefficient interpolation method (left), the velocity component interpolation method (middle), and the nearest point method (right). Only the M2 component is interpolated. The velocities at data point 9 were calculated with the data points 6, 11, and 12 (the data point numbers correspond with the data point numbers used in Fig. 23). The average grid distance is 23 km. and the nearest point distance is 10 km.
Fig. 25. Comparison of various interpolation methods to the actual ellipse. The results of the ellipse coefficient interpolation method (left), the velocity component interpolation method (middle), and the nearest point method (right). Only the M2 component is interpolated. The velocities at data point 7 were calculated with the data points 6, 8, and 12 (the data point numbers correspond with the data point numbers used in Fig. 23). The average grid distance is 31 km. and the nearest point distance is 18 km.
Fig. 26. Comparison of various interpolation methods to the actual ellipse. The results of the ellipse coefficient interpolation method (left), the velocity component interpolation method (middle), and the nearest point method (right). Only the M2 component is interpolated. The velocities at data point 6 were calculated with the data points 3, 8, and 11 (the data point numbers correspond with the data point numbers used in Fig. 23). The average grid distance is 61 km. and the nearest point distance is 27 km.
The analysed ellipse.

The interpolated ellipse.

Fig. 27. Comparison of various interpolation methods to the actual ellipse. The results of the ellipse coefficient interpolation method (left), the velocity component interpolation method (middle), and the nearest point method (right). Only the M2 component is interpolated. The velocities at data point 8 were calculated with the data points 4, 10, and 12 (the data point numbers correspond with the data point numbers used in Fig. 23). The average grid distance is 66 km. and the nearest point distance is 32 km.
Fig. 28. Comparison of various interpolation methods to the actual ellipse. The results of the ellipse coefficient interpolation method (left), the velocity component interpolation method (middle), and the nearest point method (right). Only the M2 component is interpolated. The velocities at data point 7 were calculated with the data points 4, 10, and 13 (the data point numbers correspond with the data point numbers used in Fig. 23). The average grid distance is 89 km. and the nearest point distance is 43 km.
Fig. 29. Comparison of various interpolation methods to the actual ellipse. The results of the ellipse coefficient interpolation method (left), the velocity component interpolation method (middle), and the nearest point method (right). Only the M2 component is interpolated. The velocities at data point 23 were calculated with the data points 3, 14, and 24 (the data point numbers correspond with the data point numbers used in Fig. 23). The average grid distance is 120 km. and the nearest point distance is 55 km.
Greater grid distances result in a diversity of interpolation results independent of the method used; with the interpolation of the ellipse coefficients as the best method. This agrees with the results obtained from the error calculations with the Poincaré wave. However, the errors are still too big for a useful application of this method.

When the accuracy of the tidal current data interpolation methods is important, a data point grid with a grid distance of less than about 10 km. is needed when the nearest point method is used, a data point grid with a grid distance of less than about 40 km. is needed when an interpolation method is used. These numbers were obtained from tests with current data from a part of the North Sea where the currents have large horizontal gradients (Fig. 23).

When the interpolation methods are used, the interpolation of the current components is preferable above the interpolation of the ellipse coefficients, because the accuracy when a data point grid distance of less than 40 km. is used, is almost the same, and because the amount of computer run time and memory needed for the interpolation of the current components is less (see also appendix 1).

7.5 Conclusions.
The wave length of the M2 tide in this part of the North Sea is approximately 1000 Km (the period multiplied with the square root of the waterdepth and the acceleration of gravity). A data point grid with a distance of less than 1% of the wave length is needed when the nearest point method is used, and a data point grid with a distance of less than 4% of the wave length is needed when the interpolation method is used.

The nearest point method needs an average of thirty operations per time step, the interpolation of current components needs thirty six operations per time step. These methods are highly competitive and gives better results (Fig. 30).
Two dimensional interpolation method.
Nearest point method.

Fig. 30. The path of an oil slick, assuming a constant velocity due to wind. The velocities calculated with the nearest point method show a discontinuity at the nearest point area borders. M2 and S2 ellipses were fitted into current data from a sea chart of the German Bight. At the left is the triangle area displayed, at the right the nearest point area.
8. REFERENCES

Admiralty Tide Tables 1982.

The Hydrographer of the Navy, Great Britain, 1981.

Blond, P.H. le and Mysak, L.A.

Waves in the oceans

Elsevier Scientific Publ. Comp.
Amsterdam * Oxford * New York 1978

Chung, T.J.

Finite element analysis in fluid dynamics


Defant, A.

Physical oceanography.


Hartling, J.M., Druffel, L.E., Hilbing, F.J.


Huijstee, J.J.A. van

Wiskundig model ter bepaling van getijstroomsnelheden langs de kust.

Delft University of Technology, Delft, The Netherlands, Mei 1983.

Huijstee, J.J.A. van

Marine spill simulation software set

Delft University of Technology, Delft, The Netherlands, February 1985

Kuipers, H.D.

Processes which influence the motions and characteristics of oil at sea.


Kuipers, H.D.

A simulation model for oil slicks at sea.

Lee, R.F.

Processes affecting the fate of oil in the sea.

from: Marine Environmental Pollution 1. Hydrocarbons. Editor: Geyer, R.A.

Elseviers Amsterdam, 1980.

Newman, W.M., Sproull, R.F.

Principles of interactive computer graphics.


Pedlosky, J.

Geophysical Fluid Dynamics


Sverdrup, H.U., Johnson, M.W. and Fleming, R.H.

The oceans, their physics, chemistry and general biology.

Prentice-Hall U.S.A. 1942

Veen, K. v.d.

Some preliminary results of current measurements at the KNMI station of ICES North Sea Pilot Current Meter Network.

This is an internal report of the KNMI (Royal Dutch Institute of Meteorology) and not to be considered as a publication.