Joint Defra/EA Flood and Coastal Erosion Risk Management R&D Programme

Development of estuary morphological models

Annex E: Hybrid modelling of managed realignment

R&D Project Record FD2107/PR

Environment Agency

defra
Department for Environment Food and Rural Affairs

HR Wallingford
Working with water

TR 157

Hybrid Modelling of Managed Realignment

Report TR 157
Release 2.0
July 2007

Document Information

Project	DDS0214_RLS_Development of Estuary Morphological Models
Report title	Hybrid Modelling of Managed Realignment
Client	Proudman Oceanographic Laboratory
Client Representative	John Huthnance
Project No.	DDS0214
Report No.	TR 157
Project Manager	Richard Soulsby
Project Sponsor	Richard Whitehouse

Document History

Date	Release	Prepared	Approved	Authorised	Notes
15/06/07	1.0	JS	MPD		Draft
27/07/07	2.0	JS	MPD	MPD	Final

Prepared [illegible]

Approved [illegible]

Authorised [illegible]

Summary

Hybrid Modelling of Managed Realignment

Report TR 157
July 2007

This report describes a model for prediction of the evolution of a managed realignment site under the action of tides and waves and sediment supply. Simple vegetation effects have been incorporated into the model and the model represents a framework for further developments into wave, vegetation and biological processes.

Validation of the managed realignment model was undertaken and sensitivity tests considered how variations in waves, friction and model resolution affect the predicted evolution of the model. Longer simulations were used to see how the growth of saltmarsh itself affects the evolution of the setback field. The results of the managed realignment model were compared to results from a simple ASMITA model.

The scope of this report did not consider the evolution of the breach itself as a result of extreme events, weathering and dessication but this should be considered as a potentially important effect.

Contents

Contents continued

Contents continued

1. Introduction

This report describes work undertaken at HR Wallingford for Defra project, FD2107, "Development of hybrid estuary morphological models" This project is part of Phase 2 the Estuaries Research Programme and has the overall aim of developing new modelling tools capable of delivering 50-year forecasts of morphology enabling estimates of the associated flood risks and habitat change in order to help Defra in its responsibilities relating to proposed estuarine management to accommodate climate change.

This report has been written for Work Package 3.3, "Development and application of re-alignment module". The objective of the study is the development of a model for prediction of local changes in morphodynamics resulting from managed realignment.

The remainder of this report consists of eight further chapters. Chapter 2 describes the background to the study. Chapter 3 presents a review of other relevant morphological models. The model is described in Chapter 4. A discussion of saltmarsh development is included in Chapter 5. The model is applied to the Case Study of Tollesbury Creek Managed realignment in Chapter 6. Chapter 7 consists of a sensitivity analysis of the various forcings and parameters used in Chapter 6. The Long term evolution of Tollesbury Creek is considered in Chapter 8 and the conclusions of the study are presented in Chapter 9.

2. Background to study

The tools available for resolving the issues particularly relating to the evolution of habitats created by managed realignment are not well developed, partly because of the site-specific complexity of these systems and the significant roles of tides, waves, sediment, vegetation and biology at small spatial and temporal scales. However, by adapting models which have been used in other related systems real progress could be made towards provision of a management tool. This refinement of existing modelling approaches to provide solutions to key management issues was envisaged in the Research Plan for the Phase 2 of the Estuaries Research Project (French and Reeve, 2002).

This report describes the development of a model for prediction of local changes in morphodynamics resulting from managed realignment. The methodology builds on the conceptual modelling approach to habitat development employed successfully by di Silvio (1989, 1990) and others for lagoon environments and which the Research Plan (French et al, 2002) suggested as a sound base for future progress. The overall approach can be described as hybrid – combining bottom-up and top-down predictive techniques – to describe the essential inlet functioning, and has the built in flexibility to incorporate the effects of waves, vegetation and biology and to allow future increases in complexity as the level of knowledge regarding these and other processes develops.

One of the important aspects to solve is that by definition a managed realignment site is a "virgin" terrestrial site with no representative hydraulic, sedimentological, vegetative or biological functioning upon which any prediction can be based. The method must therefore be viable in predicting *a priori* and at least qualitatively what will occur following breaching. This problem is discussed in Section 4.4 and has considerable influence as to the degree of empiricism which can be implemented in the model.

Notably the methodology is by no means restricted to use for management realignment schemes and could (and has by other researchers) be implemented in more general estuary systems.

This exercise, as well as fulfilling an important objective in the aims of the Estuaries Research Programme FD2107 project, could provide a valuable contribution to the development of the estuary management system envisaged in Phase 3 of the Estuaries Research Project.

3. Review of relevant morphological modelling approaches

3.1 INTRODUCTION

This chapter represents a short review of previously used modelling approaches which have been utilised in the prediction of changes in morphology in estuary systems.

3.2 DI SILVIO'S 2D METHOD

In this approach di Silvio creates a hybrid sediment transport model. This sediment transport model is characterised by:

- Time averaged flow and concentration
- A simplified erosion/deposition equation based on the concept of an equlibrium concentration
- The equilibrium concentration is based on empirical formulae

If T_x and T_y are the sediment flux then the erosion/deposition, E , at any point is given by,

$$\frac{\partial T_x}{\partial x} + \frac{\partial T_y}{\partial x} = E \tag{3.1}$$

where $T_x = h\left(CU - D_x \frac{\partial C}{\partial x} \right)$ and $T_y = h\left(CV - D_y \frac{\partial C}{\partial y} \right)$

The erosion/deposition is given by the first order approximation (Galapatti and Vreugdenhil, 1985),

$$E = w\,(C_i - C) \tag{3.2}$$

Where C is the concentration, C_i is the equilibrium concentration and w is a rate parameter dependent on grain size and the local tidal velocity.

di Silvio's model consisted of two basic equations solved numerically,

$$\frac{\partial}{\partial x}\left(hD_x \frac{\partial C}{\partial x} \right) + \frac{\partial}{\partial y}\left(hD_y \frac{\partial C}{\partial y} \right) = w(C - C_i) \tag{3.3}$$

$$\frac{dh}{dt} = w(C_i - C) + \alpha_e + \alpha_s \tag{3.4}$$

where α_e and α_s relate to sea level rise and land subsidence.

Di Silvio, on the basis that sediment transport takes the form $S \sim u^n$ where $4<n<6$ chose C_i to be $C_i = f_c/h^5$ for channels and $C_i = f_s/h_{HW}$ for shoals and intertidal areas.

3.3 WANG ET AL'S METHOD

Wang et al's approach is similar to that of di Silvio but here the estuary system is schematised as a series of cross-sections (essentially a 1-D model) but each cross-section is schematised into "channel", "lower flat" and "upper flat" sections which makes the model quasi-2D. As for di Silvio's method Wang et al use the simplified erosion/deposition law of Galapatti and Vreugdenhil (1985) but in the Wang model Equation 3.3 becomes:

$$\frac{\partial A_c c_c}{\partial t} + \frac{\partial A_c u c_c}{\partial x} + \frac{\partial}{\partial x}\left(A_c D_c \frac{\partial c_c}{\partial x} \right) = W_c w_s \left(c_{ce} - c_c \right) + F_{lc} \tag{3.5}$$

$$\frac{\partial A_l c_l}{\partial t} = W_l w_s \left(c_{le} - c_l \right) - F_{lc} + F_{hl} \tag{3.6}$$

$$\frac{\partial A_h c_h}{\partial t} = W_h w_s \left(c_{he} - c_h \right) - F_{hl} \tag{3.7}$$

where

A_c	is the cross-section area of the channel (below MSL), A_l is the cross-section area of the low tidal flat and A_h is the cross-section area of the high tidal flat;
c_c, c_l , and c_h	are the sediment concentrations (by volume) in the channel, low flat area and high flat area respectively;
c_{ce}, c_{le}, and c_{he}	are the equilibrium concentrations in the channel, low flat area and high flat area respectively;
W_c, W_l , and W_h	are the widths of the channel, low flat area and high flat parts, respectively;
F_{lc}	is the exchange between the channel and the lower flats;
F_{hl}	is the exchange between the lower flats and the higher flats;
D_c	is the diffusion coefficient describing the time-averaged dispersion of sediment through the channel cross-section;
U	is the residual current velocity through the channel cross-section.

Equation 3.5 is essentially the 1D advection-diffusion transport equation. The 2nd term of the left hand side of the equation is the term arising from residual transport and the 3rd term is the term arising from diffusion. Equations 3.6 and 3.7 are of the same form as Equation 3.5 except that it is assumed that there is no transport into the flats areas from upstream or downstream and only transport from the channel immediately adjacent to the flats.

The equilibrium concentration in the channel is calculated by,

$$c_{ce} = C_E \left(\frac{A_{ce}}{A_c} \right)^{n_c} \tag{3.8}$$

Where C_E is the long term overall equilibrium suspended sediment concentration and n_c is an empirical parameter.

Equation 3.8 is based on the premise that the equilibrium concentration is related to the velocity to some power and that the velocity will be larger/smaller if the cross-section area is smaller/larger than the equilibrium cross-section area, in this case derived from a best fit regime theory prism/area relationship. The equilibrium concentrations in the lower flat and upper flat areas are calculated using equations of a similar form to Equation 3.4 but involving water depth rather than area.

3.4 STIVE ET AL'S AGGREGATE MODELLING METHOD

A conceptual approach to sea level rise in coastal environments has been developed which considers the interaction between different parts of the coastal system at the macro level (Cowell et al, 2003). In order to remove the shortcomings of bottom-up models from studying long term system response these authors concentrated on the response of large-scale or aggregated coastal features. This resulted in a framework for developing a conceptual model of interaction between inlet, delta and the surrounding coast based on representation of simple box elements and simple descriptions of interaction between these elements. The approach was developed into the ASMITA model (Stive et al, 1998). For the sake of brevity the section below will restrict discussion to the basis for the ASMITA model in its simplest form and will show how a relatively simple analysis can reveal valuable insights in future long term macro-scale evolution.

The ASMITA model approach considers the interaction between tidal flats, channel delta and the surrounding coast (Figure 3.1). For simplicity we will consider only the interaction between a channel and the surrounding coast (essentially lumping the delta and tidal flats into the channel element).

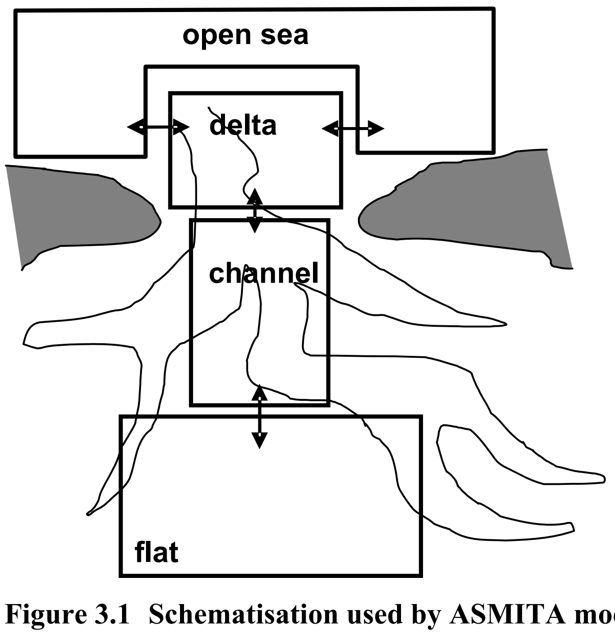

Figure 3.1 Schematisation used by ASMITA model

The erosion/deposition in the channel is given by the simplified sediment transport equation (Galapatti and Vreugdenhil, 1985),

$$\frac{\partial V}{\partial t} = w_s A_b \left(c_e - c\right) \qquad (3.9)$$

Where V is the volume of water in the channel; w_s is the settling velocity of the sediment; A_b is the surface area of the channel; c is the sediment concentration in the channel (in this case a volumetric concentration) and c_e is the equilibrium sediment concentration.

The interaction between the channel and the outside coast is given by δc and by continuity,

$$\delta(c - c_E) = w_s A_b \left(c_e - c\right) \qquad (3.10)$$

The equilibrium sediment concentration is the sediment concentration that would result in no overall morphological change. The ASMITA approach assumes that a long-term equilibrium concentration exists, which we will term c_E , and that the equilibrium concentration in the channel is a function both of c_E and of changes in velocity in the channel. It is assumed that the proportional change in the average velocity in the channel can be characterised by $(V_E/V)^r$ where V_E is the originalequilibrium volume and r is an empirical constant which takes a value in the region of 2 (for muddy sediment but would be expected to be higher for sandy sediment). This results in,

$$c_e = c_E \left(\frac{V_E}{V}\right)^r \qquad (3.11)$$

This assumption has the effect of increasingly making the channel a sink as the channel volume increases and increasingly making the channel an exporter of sediment as the channel reduces in volume.

By solving Equation 3.10 for c and using this result and Equations 3.9 and 3.11 it can be shown that the system will evolve as follows,

$$V' = V_0' \exp\left\{-\frac{t}{\tau}\right\} \qquad (3.12)$$

where $V' = V - V_E$ is the difference between the volume of the channel and its equilibrium value, t is time and τ is the time-scale of the evolution (the time taken for the initial "disturbance" from equilibrium to reduce by a factor e=2.718) given by,

$$\tau = \frac{1}{c_E . r}\left(\frac{V_E}{w_s A_b} + \frac{V_E}{\delta}\right) \qquad (3.13)$$

With the addition of sea level rise at a constant rate of ${}^{d\zeta}\!/_{dt}$ the Equation 3.9 becomes (Stive and Wang, 2003),

$$\frac{\partial V}{\partial t} = \frac{w_s A_b \delta C_E}{\delta + w_s A_b}\left(\left(\frac{V_E}{V}\right)^r - 1\right) + A_b \frac{d\zeta}{dt} \tag{3.14}$$

When sea level rise is constant it can be shown that a dynamic equilibrium can be established (whereby *V* tends to a constant but the channel is changing in its elevation with respect to a fixed datum at the rate of sea level rise) and that,

$$V_{SL} = \frac{V_E}{\left(1 - \frac{d\zeta}{dt} . \frac{\delta + w_s A_b}{w_s \delta C_E}\right)^{1/r}} \tag{3.15}$$

where V_{SL} is the equilibrium volume in response to sea level rise and V_E is the equilibrium volume without sea level rise.

This result shows that a channel system experiencing sea level rise can in general keep pace with sea level rise because, as the volume of the channel increases with the rise in water level the channel increasingly will become a sink for local sediment and the rate of vertical infill will approach that of the rise in water level. Note that this methodology establishes that this equilibrium is based on the size of the system, the sediment supply, the sediment type and the rate of sea level rise. However, the channel can only keep pace with sea level rise if the rate of sea level rise is below the critical value given by,

$$\frac{d\zeta}{dt} = \frac{w_s \delta C_E}{\delta + w_s A_b} \tag{3.16}$$

This analysis suggests that rollover will occur in some form for rates of sea level rise less than that given by Equation 3.16 and drowning will occur for rates of sea level rise higher than this. Estimates made by Van Goor et al (2002) for the critical rate of sea level rise for tidal inlets in the Netherlands were of the order of 10-20mm/yr.

3.5 DISCUSSION

3.5.1 Application of the di Silvio and Wang et al models to the managed realignment problem

The approaches of di Silvio and Wang et al represent a basic approach which offers great promise for application to the prediction of morphological evolution of managed realignment schemes. In particular, the simplified time-averaged erosion/deposition model of Galapatti and Vreugdenhil (1985) affords a great reduction in model complexity (and therefore run time) without sacrificing too much in the way of realistic processes. The 2D nature of sediment transport in the set back fields of managed realignment schemes means that di Silvio's model structure is more appropriate than Wang et al's but the main problem that exists with using either model is the reliance of these models on empiricism. These models require a considerable amount of data from existing estuary systems to evaluate the parameters needed for the relationships defining equilibrium concentration. Managed realignment is by definition concerned with a virgin terrestrial system where there is no information available *a priori*. A different approach is therefore required to establish these relationships.

3.5.2 Extending the availability of behavioural models

The di Silvio model and the Wang *et al* model described above are purpose built models which (at present) are not commercially available. The development of such a model is an onerous task which would preclude most end-users from benefiting from this type of approach. This study had an additional objective therefore to show how readily available software (a flow model and a mud transport model) can be used, with some minor modification and some post-processing, to create the same type of model.

3.5.3 Application of the ASMITA analysis to managed realignment

The analysis presented in Section 3.4 is useful because it provides a "back of the envelope" methodology for evaluating the time-scale of change within the setback system. However, the methodology requires some alteration to be applied to the managed realignment problem.

Here we schematise the "model" as a "channel" element (representing the setback field) and an "offshore" element (representing Tollesbury Fleet and the outside world). The ASMITA formulation assumes that the exchange, δ, between the setback field and Tollesbury Fleet is a constant value throughout the simulation. In the case of the setback field, however, all of the volume of the field to HWS is tidal volume and so as the field reduces on volume, so does δ. This means that to first order (ie neglecting tidal asymmetry effects) the current velocity in the set back field remains constant throughout the evolution. As a result the value of *r* which describes how the equilibrium concentration varies with changes in depth, becomes only a function of wave action. However, Equation 3.11 and 3.12 remain of the same form, the only change being in the value of *r*.

The parameter *r*, now representing the effect of changes in wave action with changes in water depth is not straightforward. Wave-generated shear stress is a function of friction and orbital velocity. Based on the approximation of Soulsby (2006) for the orbital velocity of irregular waves and the evaluation of friction under rough turbulent, smooth turbulent and laminar conditions as given by Soulsby (1997), to leading order wave-generated shear stress varies as h^{-p} where h takes a value in the range 0.5-0.8. It is therefore expected that the value of *r* should take a similar range.

This report will consider the case of the Tollesbury set back field (see Chapter 6). For this example the values of the parameters are as follows:

V_E :	not known *a priori* but is calculated to be $2.3\text{x}10^5$m for the Tollesbury site;
δ :	equal to V;
A :	$2\text{x}10^5$m;
C_E :	The equilibrium concentration is assumed to be equal to 0.05kg/m^3 but needs to expressed as the equivalent volume change on the bed per metre cubed of water. It is assumed that the dry density of the bed is 500kg/m^3 so C_E is $1\text{x}10^{-4}$ (no units);
W_s :	is the settling velocity, in this case assumed to be $3\text{x}10^{-4}$m/s.

This gives a range of values for the time scale as, 20-30 years. Note that this is the time scale needed for (just under) a three-fold reduction in the rate of change of volume. A ten-fold reduction would require (just over) double this time.

4. Description of the model

4.1 OVERALL STRUCTURE

The model has been developed for prediction of local changes in morphodynamics resulting from managed realignment. The methodology has built on the conceptual modelling approach to habitat development employed successfully by di Silvio (1989, 1990) but adjusted for the additional complexity of the managed realignment environment. This approach is a hybrid – combining bottom-up and top-down predictive techniques.

The model structure is based on a UNIX shell script which is simple and allows flexibility enabling the user to implement the software which they possess rather than proprietary software. For those users who prefer windows the UNIX shell script is simple to adapt to the windows equivalent (e.g. visual basic). The shell script controls the application of a flow model, input of information about waves, a program which derives the important morphological parameter of equilibrium concentrations and the time-averaged dispersion characteristics and a "di Silvio-type" sediment transport model (See Figure 1) originally developed by Galapatti and Vreugdenhil (1985).

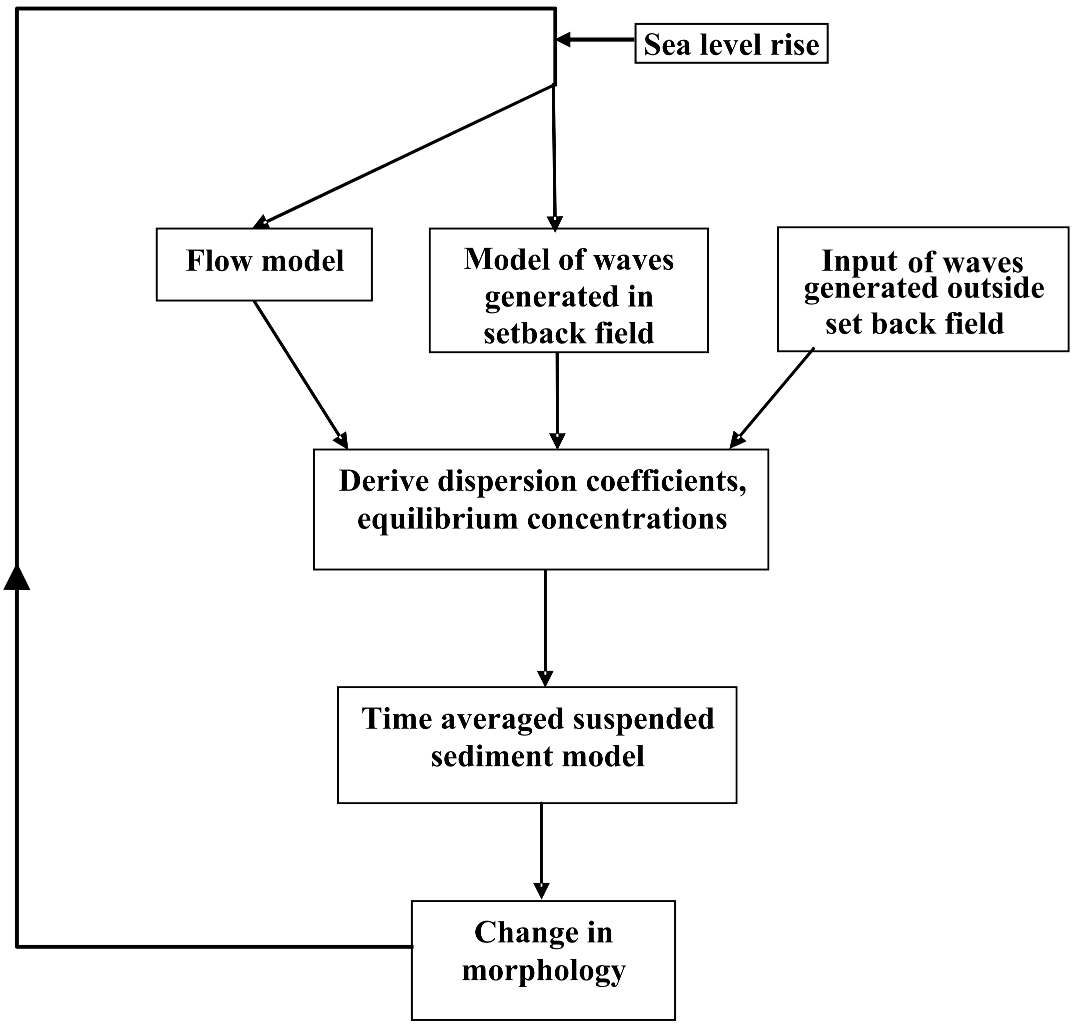
Sea level rise
Flow model
Model of waves generated in setback field
Input of waves generated outside set back field
Derive dispersion coefficients, equilibrium concentrations
Time averaged suspended sediment model
Change in morphology

Figure 4.1 Basic structure of morphological model

In greater detail the model structure is as follows:

1. Set up initial bathymetry
2. Work out time-averaged wave heights and periods at every point in model domain
3. Use TELEMAC-2D flow model to get flow conditions in set back field
4. Post-process the flow results file and wave results
   – to derive time-averaged spatial distribution of diffusion coefficients (Dronkers et al, 1981, see Appendix 7) and
   – to derive spatial distribution of (time-averaged) equilibrium concentrations (incorporating wave effects)
   – save derived values in a form which can be used as input to SUBIEF-2D model.
5. Run time-averaged sediment transport model [SUBIEF-2D]
   – use derived time-averaged diffusion coefficients and zero residual currents [ie diffusive process only]
   – Use Equation (2) as sediment transport equation with C_i (equilibrium concentration)
   – Update bathymetry during model simulation
6. Extrapolate predicted change in bathymetry from SUBIEF-2D model over a much longer time step and save to results file
7. Bathymetry is used as basis for another run of the TELEMAC-2D model – go to 2.

4.2 MODEL JUSTIFICATION

4.2.1 *Justification for the time averaging of flooding and emptying of realignment site*

At the core of the methodology described above is the idea that a set-back field which fills and empties can be represented by a time-averaged (tidally averaged) model. In reality the principle process is advection of sediment into the field, followed by deposition, rather than dispersion which is represented in the model.

The methodology concentrates on the situation whereby the set-back field is filled every spring tide to (local) High Water Springs by sediment laden water and so in the model the field is permanently full. In reality (on average) the field is half full and depths are half their maximum value. Thus without some adjustment, the model will overestimate the diffusion/dispersion (because depths are represented in the model as their maximum values) and will overestimate erosion/deposition (because in the model deposition can occur for the whole tide rather than just when wetting occurs). In fact the selection of appropriate time-averaged values of dispersion is not an exact science and it is enough in this case to reduce the dispersion values by a factor of two to account for the time-averaged depth being roughly half of the maximum depth. However, the selected settling velocity and erosion rate have to be increasingly reduced in shallower water to take account of the reduced time available in reality for deposition/erosion to take place.

4.2.2 *Justification for time averaging of diffusion*

Time averaging is valid if the diffusion is small with respect to the scale of the spatial variations of the velocity gradients (de Jong, 1998), ie, if

$$D << \frac{r_{\partial u}^2}{T}$$

where D is the diffusion, $r_{\partial u}$ is the radius inside which the spacial velocity gradient, $\frac{\partial^2 \tilde{u}}{\partial \tilde{x}^2}$ is relatively uniform, and T is the tidal period. In the case of the Tollesbury set back considered in this study this is true except for near to the entrance to the set back field.

It can be shown formally (de Jong, 1998) that if the above criterion is satisfied and there is negligible residual flow that time-averaged dispersion approximates to the corresponding time-varying dispersion. However, the model developed by de Jong intrinsically assumed that the tidal amplitude was small compared with the water depth. In the case of managed realignment a substantial part of the set back field will usuallydry. The model used in this study can therefore be considered as something of an "engineering" approximation – i.e. it has not been formally shown that a wetting/drying time-averaged model approximates to its time-varying equivalent.

4.3 METHOD FOR DERIVING THE TIME-AVERAGED DIFFUSION

As there is no residual transport in the model, the time-averaged transport into the field is given by Equation 3.3. The diffusion of sediment into the field is controlled by the value of the diffusion coefficient. For this study most of the simulations of evolution were undertaken using a space-varying distribution of time-averaged diffusion coefficient. The diffusion coefficient was assumed to be proportional to the square of the time-averaged current speed within the setback field (on the basis of work by Dronkers et al, 1982). The time-averaged sediment transport model then used the distribution of diffusion to model the movement of sediment into the setback field. The absolute magnitude of the diffusion coefficients was calibrated along with the other parameters used in the model.

4.4 METHOD FOR DERIVING THE EQUILIBRIUM CONCENTRATION

As mentioned in Chapter 2, a managed realignment site is a virgin terrestrial site with no representative hydraulic, sedimentological, vegetative or biological functioning upon which any empirical evaluation of the equilibrium concentration can be based. In the absence of a general empirical law governing the evolution of muddy tidal inlets the most obvious course is to use a simple analytical method based on process.

The equilibrium (time-averaged) concentration is given by equating the deposition occurring during slack water with the erosion occurring during the rest of the tide. This is summarised by the following equation,

$$\int_{\tau>\tau_e} M_e(\tau-\tau_e)\,dt = \int_{\tau<\tau_d} w_s C\left(1-\frac{\tau}{\tau_d}\right)dt = C_E \int_{\tau<\tau_d} w_s\left(1-\frac{\tau}{\tau_d}\right)dt \qquad (4.2)$$

The value of C_E at any location is thus dependent on the variation of current speed and wave action and the friction parameter (which give the value of bed shear stress), the thresholds of erosion and deposition, the settling velocity and the erosion rate.

It should be noted that there is inherent uncertainty in the values of friction, the thresholds of erosion and deposition, the settling velocity and the erosion rate. To some extent these values can be estimated from typical values of the literature but there will always be an appreciable amount of uncertainty in the value of C_E .

4.5 PROCESSES CURRENTLY NOT INCLUDED IN THE MODEL

Processes not currently included in the model for the purposes of this report include:

- The effect of biology on bed shear stress;
- Erosion of the initial bathymetry (re-erosion of deposited sediment is reproduced);
- Geotechnical processes, such as failure, slumping, dessication, etc., which are proceses which would enhance the erosion of the sea walls at the entrance to the setback site.

4.6 COMMERCIAL SOFTWARE USED IN MODEL

The morphological model uses the following commercial software:

- The TELEMAC-2D flow model, a finite element model which solves the shallow water equations. The model was designed by LNHE and is described in Appendix 1.
- The SUBIEF-2D suspended sediment model, the mud transport module of the TELEMAC suite, again designed by LNHE. The model is described in Appendix 2. For this study, however, the code was altered (see Appendix 3) to make the following changes:
  - the dispersion coefficients and equilibrium concentrations calculated from the flow model results were read
  - the calculation of erosion and deposition were changed from the normal formulations of Krone (1962) and Partheniades (1965) to Equation (3.2).

It is noted that the alteration of commercial software codes in this manner, even though relatively minor in scope, requires access to the code. A number of commercially available models do not allow such access to the code and therefore could not be used in the manner envisaged in this report.

4.7 ADDITIONAL SOFTWARE DEVELOPED FOR MODEL

4.7.1 The shell script

The shell script is presented in Appendix 4. The script essentially sets up the initial conditions and runs the various routines/models in sequence for as many time steps as required.

4.7.2 The dispcequ routine

The code for the *dispcequ* routine is presented in Appendix 5. The *dispcequ* routine performs the following functions:

- it reads the flow model results file
- it calculates the time series of shear stress due to currents at every node in the model domain
- it calculates the shear stress due to internally derived and (if the wave field is provided) externally derived waves
- it integrates the combined shear stress due to waves and currents through a user-defined period of time and, by balancing the erosion that would result from this wave stress with the deposition necessary to maintain equilibrium, calculates the

concentration required to maintain such an equilibrium at every node in the model domain.
- it calculates the time-averaged dispersion coefficient for sediment entering the set back field.

The *dispcequ* routine uses Young et al's (1996) method for wave evolution in shallow water to derive the internally generated waves (see Appendix 6). The format of the steering file is shown below. Essentially there is a line of text describing the input required, followed by the input, followed by a slash to indicate a blank line.

```
Description of steering file
/
flow results file
name
/
start and end times
time1,time2
/
physical roughness length
length
/
Windwaves to be included?
Yes/no
/
wind wave input to be worked out offline?
Yes/no
/
Name of wave climate file
name
/
how many wind speed classes are there?
Number of classes
Speed1
Speed2
..
Speedn
/
how many wind direction classes are there?
Number of classes
Dir1
Dir2
..
Dirn

/
what is the water level cut off for local waves?
Level (beneath which waves in field are considerd insignificant)
/
water depth limit for implementing effect of waves?
Depth (below which waves are considerd insignificant at any
point)
/
swell waves to be included?
No (swell wave capability not developed)
/
/if answer is "yes" then need to input more info here
/
erosion threshold
```

$\tau_{deposition}$, $\tau_{erosion}$ *(units are N/m²)*
```
/
erosion constant
Me1 , Me2 (second value needs to be present but not used by model)
/
settling velocity
speed in m/s
/
output file
name
/
```

4.7.3 The write_geom routine

The *write_geom* routine is presented in Appendix 7. Its purpose is to write temporary files containing the new bathymetry.

4.7.4 The write_bathy routine

The *write_bathy* routine is presented in Appendix 8. Its purpose is to write the prediction of bathymetry at each time step into the bathymetry results file.

4.7.5 The pick_morph routine

The *pick_morph* routine is presented in Appendix 9. Its purpose is to read a TELEMAC-2D Serafin-type file with lots of time steps and to save the first time step of the file.

4.7.6 Application of the code to other models

The main issues to consider when applying the basic methodology to other flow and sediment transport models are as follows:

- The erosion/deposition equation in the sediment transport code will need to be changed to Equation 3.2.
- The calculation of dispersion coefficient in the *dispcequ* program is currently hard-coded and site-specific and will need to be adapted for the site in question.
- The methodology presented above was developed for the (finite element) TELEMAC suite and assumes the results files of the flow model and the sediment transport model are in the following (binary) form:

Headers … (several lines containing the number of nodes and elements, number of variables in results files, variable names, node positions and node/element connectivity).

Time 1
Variable 1, *N* values
Variable 2, *N* values
..
Variable *M, N* values

Time 2
Variable 1, *N* values
Variable 2, *N* values

..
Variable *M*, *N* values
..

Time *T*
Variable 1, *N* values
Variable 2, *N* values
..
Variable *M*, *N* values

When reading in results files of a differing format the relevant parts of the code will have to be modified.

The code is written on the basis that the important variables are 1D arrays consisting of a value for each node. This may not be true of other models, for instance finite difference models where each value can be a 2D array where a value is assigned to the i^{th} row and j^{th} column of the model grid. To use the code presented the easiest way may therefore be to write conversion routines to transform the flow and sediment transport results files into files of 1D arrays and to transform them back for re-running the flow and sediment transport models. As this exercise is relatively computationally quick there will be a trivial impact on the length of the model run time.

5. Conceptual model of salt marsh development

When a seawall experiences deliberate breaching or failure, allowing tidal waters to enter a previously dry or freshwater site, the resulting evolution of the area behind the breach, which will be termed the setback field, is a result of the combination of a number of key processes:

- Flood tides move water into the managed realignment site carrying with them suspended sediments that subsequently deposit in the quiescent waters of the field.
- On ebb tides the currents are insufficient to resuspend deposited muds and silts resulting in sediment accumulation.
- Once the setback field is breached it is possible for waves to be generated internally within the field and externally outside of the field. This wave action will reduce and potentially prevent accumulation in certain areas (see below).
- In addition the breaching of the site causes increases in current speed in the approaches to the site, causing scour on the foreshore. This morphological change outside the site can be significant and cause adverse effects on existing habitat. However, this effect is not considered within the scope of this study.

As the setback field accumulates with sediment large areas of intertidal mudflats form. As the level of the mudflats rises the volume of water entering the setback field reduces together with the total volume of sediment entering the setback field. As a consequence the rate of accumulation in the field reduces.

Once parts of the field reach a high enough elevation relative to the tidal frame (the rule of thumb is that this level is similar to High Water Neaps) pioneer colonization by saltmarsh species can occur. Once established this colonization process becomes more rapid through lateral expansion of communities. Figure 5.1 (adapted from a similar figure by Williams and Orr, 2002) illustrates the progress of evolution from the in situ environment to mudflat, to pioneer colonisation to mature saltmarsh.

Once colonisation has occurred vegetated marsh occurs through lateral expansion of rhizomes from each established plant on the mudflat and from plants along the site perimeter (Williams and Orr, 2002). The presence of vegetation contributes to vertical accretion through sediment trapping and organic production and accumulation (Eisma and Dijkema, 1997). Once saltmarsh starts to form the tidal creek drainage system will become more defined within the setback field structure, due to imprinting by the establishing vegetation.

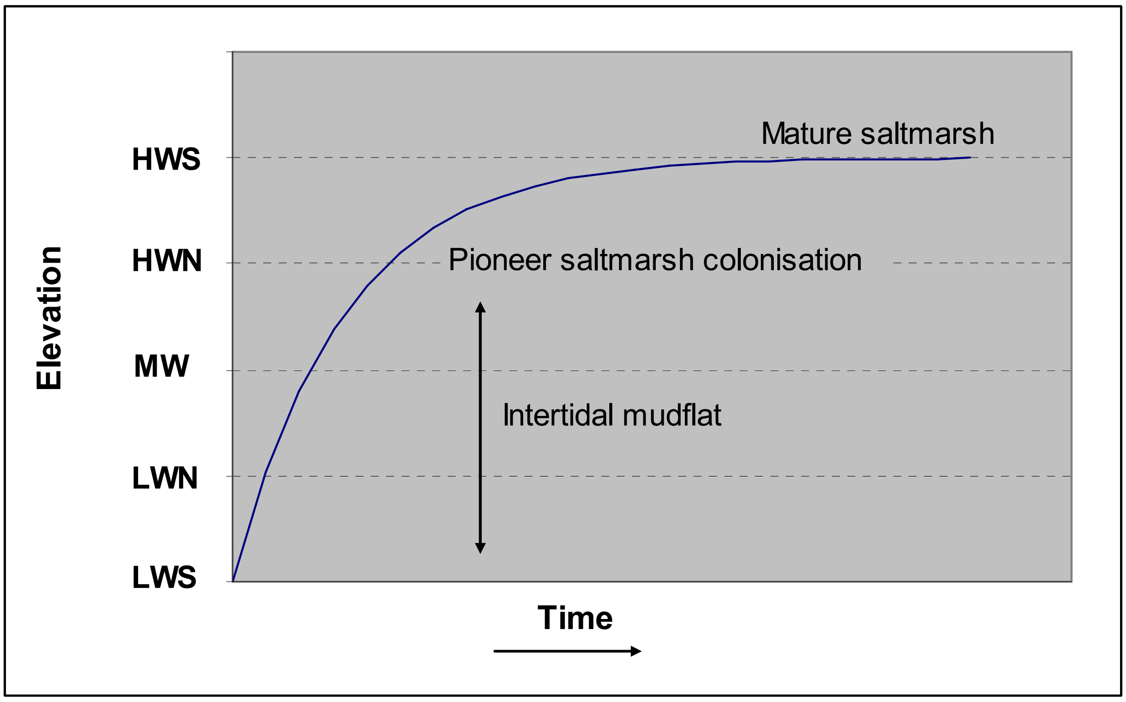

Figure 5.1 Conceptual model of saltmarsh evolution with time after breaching

Williams and Orr (2002) cite three important physical processes which act to hinder the evolution of a setback field to a vegetated marsh:

- *Restricted tidal exchange*. The rate of sedimentation is directly related to the amount of sediment brought in by the flood on every tide. A narrow breach entrance or an entrance with an elevated sill or poor drainage within the setback field will reduce the tidal exchange and therefore the mass of sediment available for sedimentation.
- *Limited sediment supply*. This is self explanatory. The rate of sedimentation in the setback field is limited by the rate at which this sediment can be supplied by the local system outside of the setback field.
- *Wave action*. Propagating waves, either generated internally within the field or externally outside the setback field create turbulence in the water column in addition to that produced by tidal currents. The wave disturbance itself is a function of fetch length and wind speed (see Appendix 6) but the turbulence generated is inversely related to water depth. Thus, in general, as the elevation of the setback rises with sediment accumulation the effect of waves will be enhanced.

This effect can significantly retard the evolution of the system and even prevent areas of saltmarsh forming as illustrated in Figure 5.2 (from Williams and Orr, 2002).

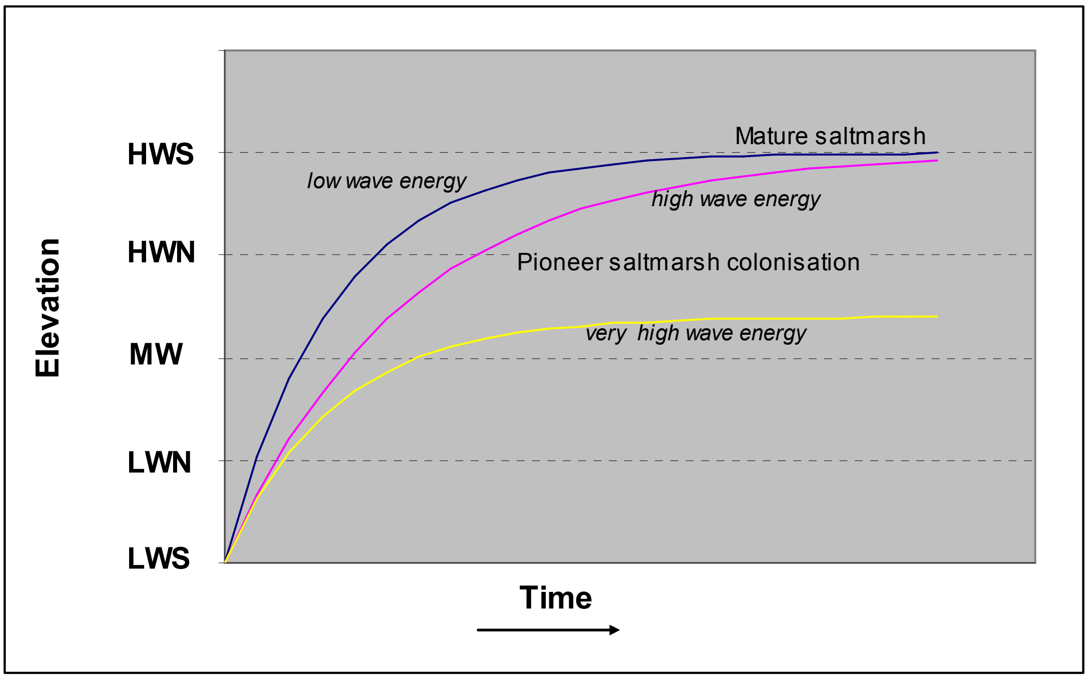

Figure 5.2 Conceptual effect of waves on tidal saltmarsh evolution

6. Case study: Tollesbury Creek

6.1 INTRODUCTION

The case study chosen to pilot the model was the managed realignment in Tollesbury Creek. The tidal creek investigated links a set back field with the Tollesbury Fleet Estuary (Figure 6.1), part of the Blackwater Estuary in South East England. This managed retreat experiment was part of a MAFF (now Defra) funded study following breaching of the set back field in 1995.

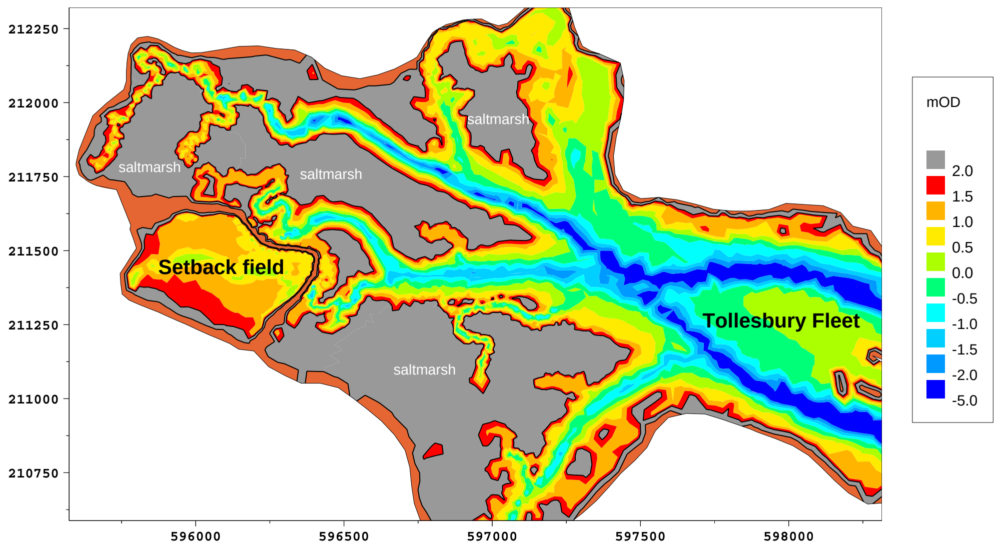

Figure 6.1 Tollesbury Fleet and the managed realignment site

The Tollesbury managed realignment site is located at the end of a 1km long creek, itself part of Tollesbury Fleet. Mean spring tidal range within the creek is of the order of 4m, ranging from –1mOD to +3mOD. The intertidal area of the creek is composed of a complex network of saltmarsh cliffs and small-scale creeks and most of the creek dries out at low water (Figure 6.2). Water depths within the Creek vary up to 4.5m. Peak currents are of the order of 0.3-0.5m/s and wave heights are small.

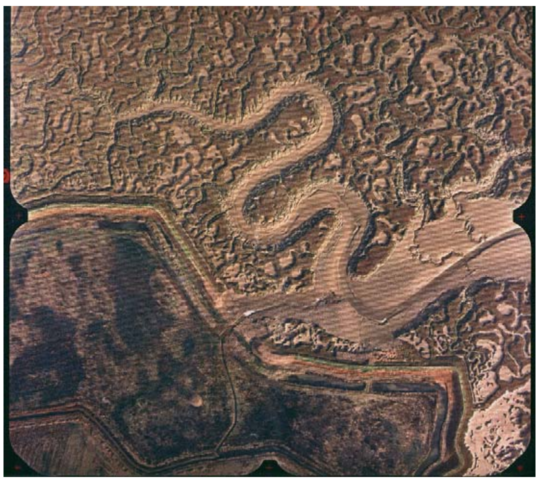

Figure 6.2 Aerial photo of Tollesbury Creek and the managed realignment site

The set back field is shown in Figures 6.1 and 6.2. The field is slopes gently upwards towards the west with "borrow dykes" cut around the perimeter of the field. The remains of an old drainage channel runs east to west across the field where it used to run out through the south west perimeter of the field. For the breach a further drainage channel was cut to link the mouth to the old drainage channel.

For the breach itself a JCB was used to dig out the entrance through the seawall (Figure 6.3). The base of the sea wall is clay based and there has reportedly been little erosion of the entrance itself following the initial breach design.

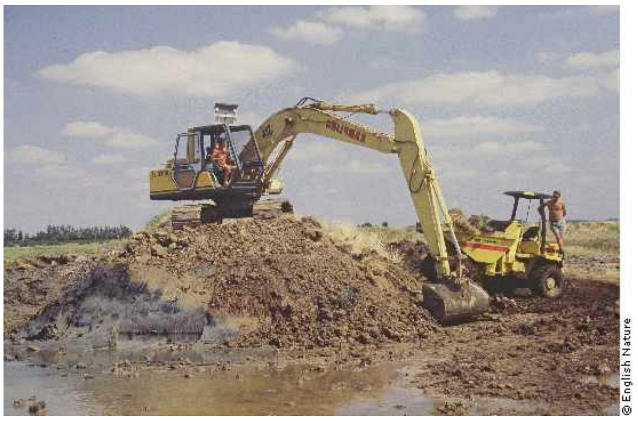

Figure 6.3 Breaching of Tollesbury Creek Realignment

6.2 OBSERVATIONS OF BATHYMETRIC CHANGE

CEH(2000) gives details of measurements of bathymetric change at 25 Locations in the setback field between 1995 and 2000. These locations are shown in Figure 6.4. The measurements are summarised in Tables 6.1 and 6.2.

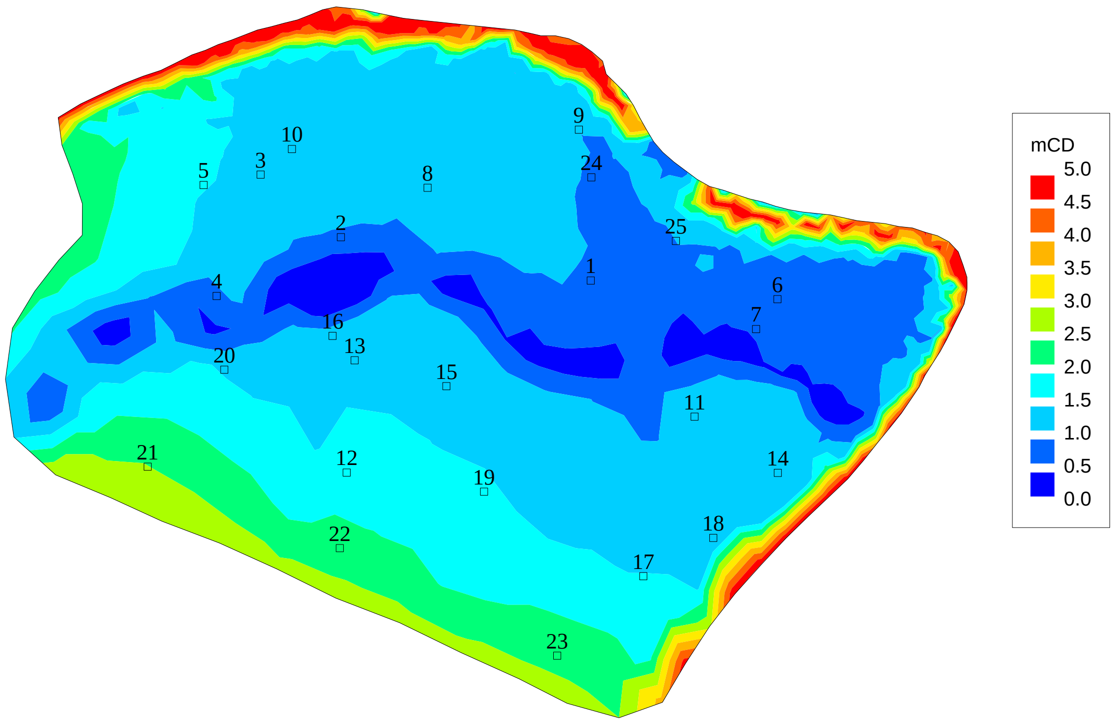

Figure 6.4 Locations of bathymetric measurement points

Table 6.1 Mean increase in surface level at individual transects within the experimental site at Tollesbury from August 1995

Transect	Mean increase in surface level to January 1998 (mm)	Mean increase in surface level to November 2000 (mm)
1	+ 5	+ 65
2	+ 86	+ 169
3	+ 100	+ 222
4	+ 43	+ 128
5	+ 63	+ 103
6	+ 34	+ 67
7	+ 55	+ 87
8	+ 78	+ 217
9	+ 34	+ 74
10	+ 87	+ 202
11	+ 78	+ 189
12	+ 18	+ 54
13	+ 102	+ 193
14	+ 67	+ 165
15	+ 43	+ 133
16	+ 50	+ 158
17	- 9	+ 7
18	+ 74	+ 137
19	+ 29	+ 121
20	+ 16	+ 50

Table 6.2 Mean increase in surface level at individual transects within the experimental site at Tollesbury from April 1999 to November 2000

Transect	Mean increase in surface level (mm)
21	- 1
22	- 1
23	+ 2
24	- 8
25	- 30

On average the mean measured accretion rate within the setback field over the first 5 years following the breach was around 25mm/yr, varying from 20mm/yr in 1997/1998 to 30mm/yr in 1995/1996.

The model was implemented to try and achieve the measured changes in bathymetry.

6.3 CALIBRATION DATA

The data required to drive the morphological model includes:

- The tidal boundary condition at the mouth of the set back field
- The time-averaged suspended sediment concentration at the mouth of the set back field
- The time-averaged diffusion coefficient
- Settling velocity
- Threshold of erosion
- Erosion rate constant
- Bed density
- Friction coefficient

The tidal boundary condition. The tidal boundary condition for the flow model was taken from the results of the flow modelling of the effects of the breach described in Chesher et al, (1995). The predicted tidal profile at the entrance from the model results was used as the boundary condition for the present study. The tidal profile used is shown in Figure 6.5.

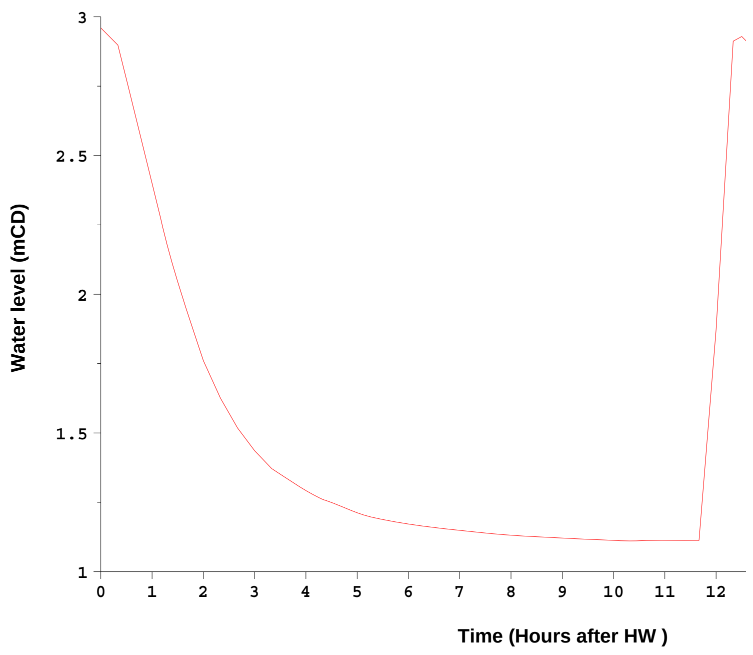

Figure 6.5 Tidal boundary conditions used in flow model

Suspended sediment Concentrations. These were not available and so the value chosen in the end was 50mg/l based a number of measurements made in the Blackwater Estuary but at other locations (HR Wallingford, 2001):

- Measurements by HR Wallingford taken at three locations in the nearby Salcott Creek over a period of 5 weeks (2 locations) and 4 months (one location), and at 0.1, 0.2m and 0.5m above the bed, found that mean suspended sediment concentrations varied between 50mg/l and 270mg/l.
- Measurements at nearby Brightlingsea recorded in March 2000 found typical spring tide peak values of around 70mg/l and typical neap tide peak values of around 25mg/l.
- Measurements at Maldon at the head of the Blackwater in April 1998 found typical spring tide peak values of under 50mg/l with occasional peak values of 100-150mg/l.

Time-averaged diffusion coefficient. For all runs the diffusion coefficient was was assumed to be proportional to the square of the time-averaged current speed (Dronkers et al, 1982) calculated from the flow model results. The diffusion was set to be high in the region of the entrance to the setback field.

Settling velocity. This was set arbitrarily at 0.3mm/s.

Threshold of erosion. The threshold of erosion is used to derive the equilibrium concentration values. The threshold was set arbitrarily at 0.05N/m^2.

Erosion rate constant. The erosion rate used in the erosion/deposition of fresh sediment was set at 0.000075 kg/s/m^2.

Bed density. The dry density (ie the mass concentration) of material deposited on the bed of the set back field was set arbitrarily to 500kg/m^3.

Friction coefficient.
Initially the friction coefficient was set to 0.00005m which corresponds to smooth muddy conditions of freshly deposited sediment. A sensitivity run (see Section 7.3) was undertaken for a higher value (0.003m) more appropriate to the initial conditions in the field.

6.4 RESULTS OF CALIBRATION

The results of the calibration are shown in Figure 6.6 which compare the predicted morphological change with the observed changes for the first 5 years respectively. It can be seen that the model performance is not unreasonable but that the observed data is insufficient to ascertain whether the model is truly predicting the correct pattern of morphological change in the set back field.

For the purposes of this report, however, the model was assumed to be calibrated (as far as possible with the available data) on the basis of Figure 6.6.

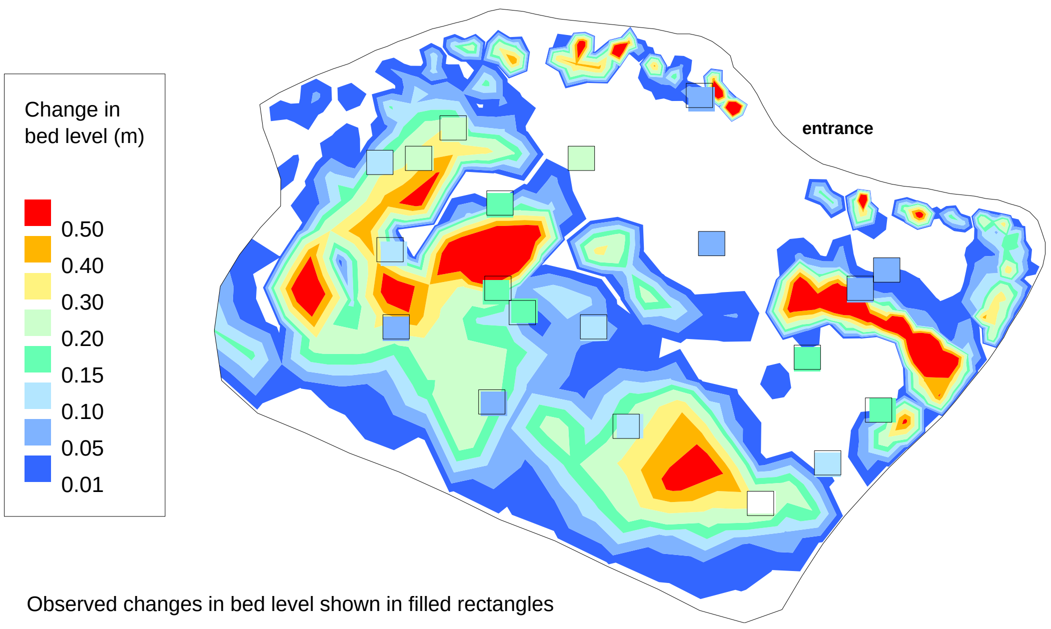

Figure 6.6 Comparison of model predictions against observations of changes in bed level between August 1995 and November 2000

In addition the model was also compared against LiDAR data obtained from the Environment Agency for the year 2002. The pre-breach (1995) bathymetry of the field (not derived from LiDAR) is shown in Figure 6.4. The 2002 LiDAR data set is shown in Figure 6.7. The corresponding change in bathymetry is shown in Figure 6.8. The benefits of the having entire spatial coverage of the setback field is obvious when trying to understand the evolution of the field and to calibrate the morphological model. However, the difference between these measurement methodologies means that the accuracy (both vertical and horizontal) of the individual measurements is reduced. The usefulness of this measure of bathymetric change is therefore in identifying the general pattern of change across the field, which is lacking in the spot measurements shown in Figure 6.6. For this reason Figure 6.8 is presented with different (coarser) contour classes.

The LiDAR/survey comparison shows that the main features of change are accretion in the drainage channel running through the setback field, the accretion in the west corner of the field and the accretion in the drainage channels along the north and south edges of the field. The LiDAR/survey comparison also indicates erosion in the central/southern part of the field. This erosion may be more of a by-produce of the survey methodologies. The erosion at the mouth is more likely to be real, however.

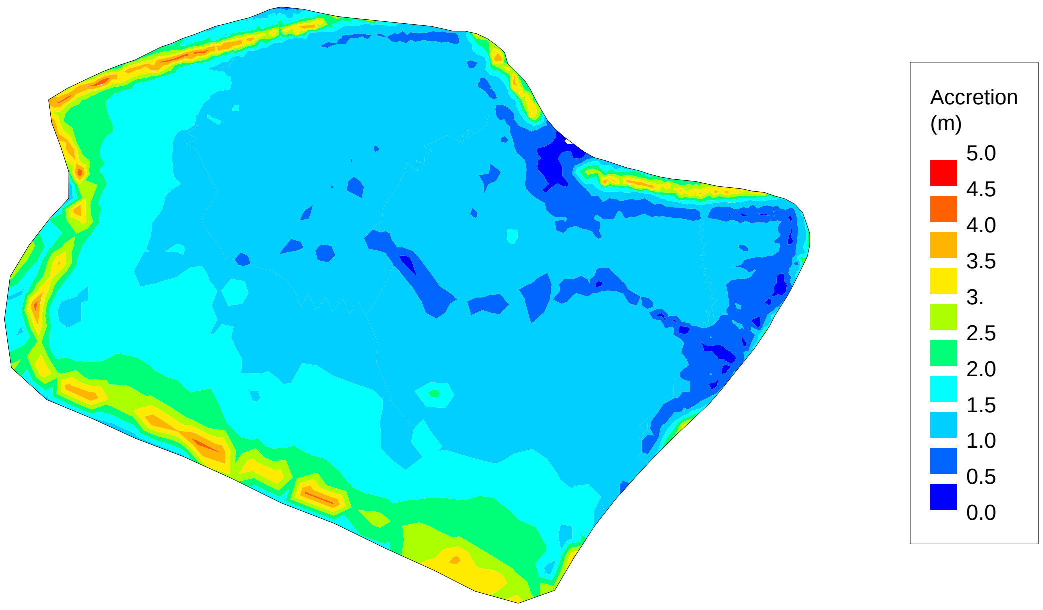

Figure 6.7 2002 LiDAR data

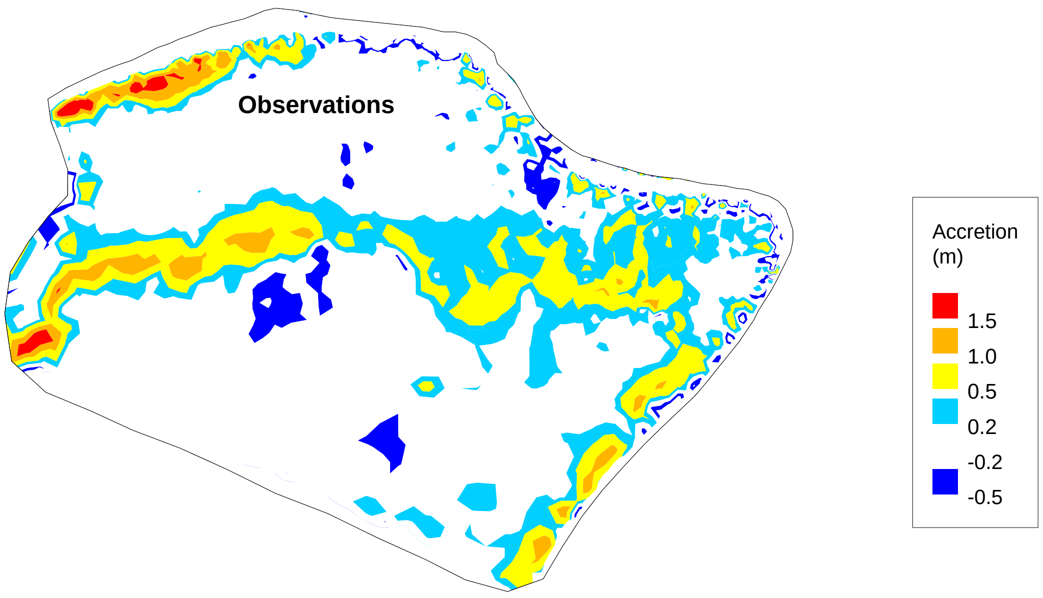

Figure 6.8 Difference between 1995 survey and 2002 LiDAR data

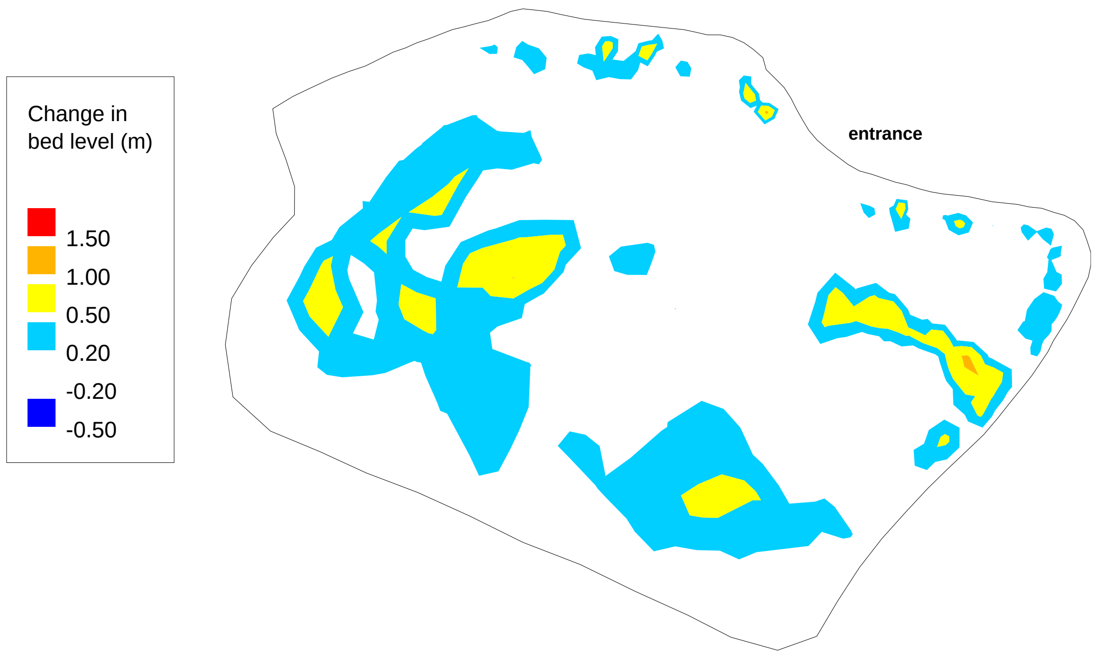

Figure 6.9 Model prediction of 2002 bathymetry

The comparison of the model prediction to the 2002 bathymetry shows that the model predicts the deposition in the main drainage channel and to some extent the deposition in the west corner of the field, but does not predict the deposition in the drainage channels in the north and south of the field. These channels are not represented in the bathymetry due to the coarse resolution.

In addition the model predicts a patch of deposition in the southern central part of the field. This does not occur according to the LiDAR/survey data.

Section 7.2 shows a comparison of the model with a much finer grid, where the model prediction is much closer to Figure 6.8.

7. *Sensitivity analysis*

7.1 EFFECT OF WAVES GENERATED INTERNALLY IN SET BACK FIELD

Here the result shown in Figure 6.6 was compared to the corresponding result without any internally generated waves. The results are shown in Figure 7.1. It can be seen that the waves are instrumental in preventing accretion just inside of, and to either side of, the set back field entrance.

The combined bed shear stress due to waves and currents was calculated by adding the bed shear stress due to currents to that due to waves. Orbital wave velocities were calculated using the methodology of Soulsby and Smallman (1986). The bed shear stress generated by small wind waves in shallow water is sensitive to whether the waves are laminar, smooth turbulent or rough turbulent (Soulsby, 1997). For this study it was assumed that the presence of a current laminarised the wave field in shallow conditions.

The jet-like movement of water through the entrance produces large circulations on either side of the entrance, but owing to the entrance geometry, particularly on the east side of the entrance (The right-hand side of the entrance in Figure 7.2). Without the shear stress from internal waves these eddies result in areas of accretion.

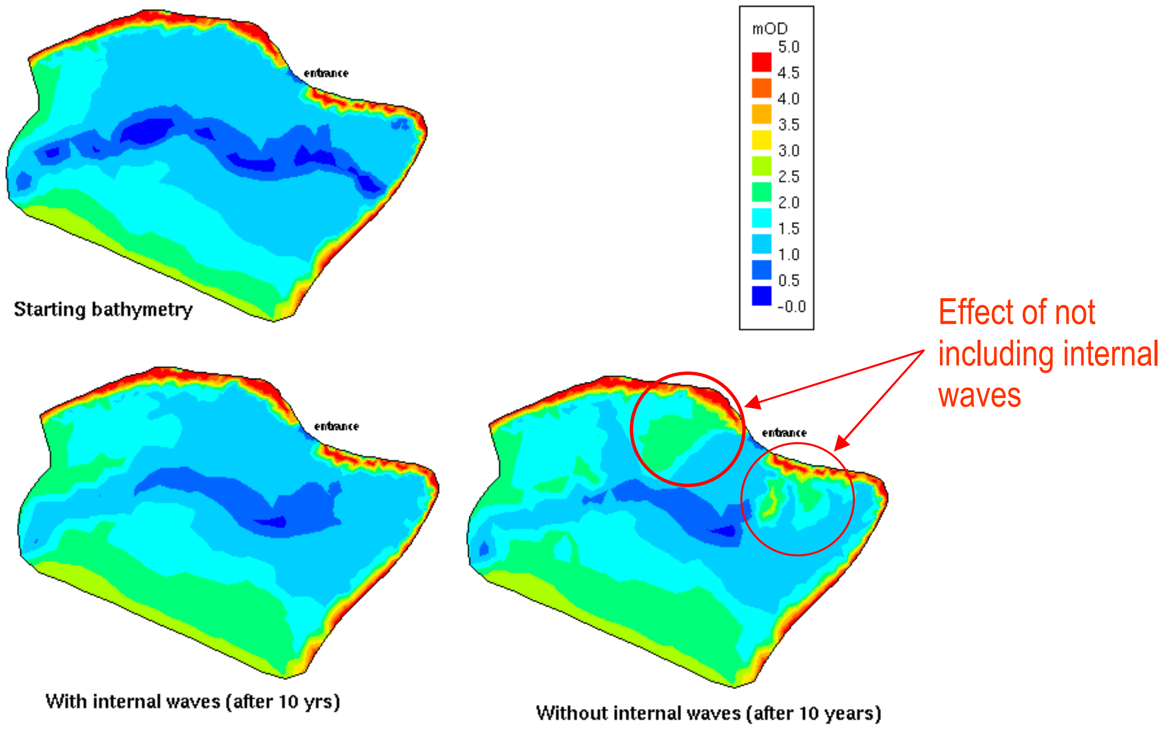
entrance
Starting bathymetry
mOD
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
-0.0
Effect of not including internal waves
entrance
With internal waves (after 10 yrs)
entrance
Without internal waves (after 10 years)

Figure 7.1 Effect of internal waves of evolution of set back field

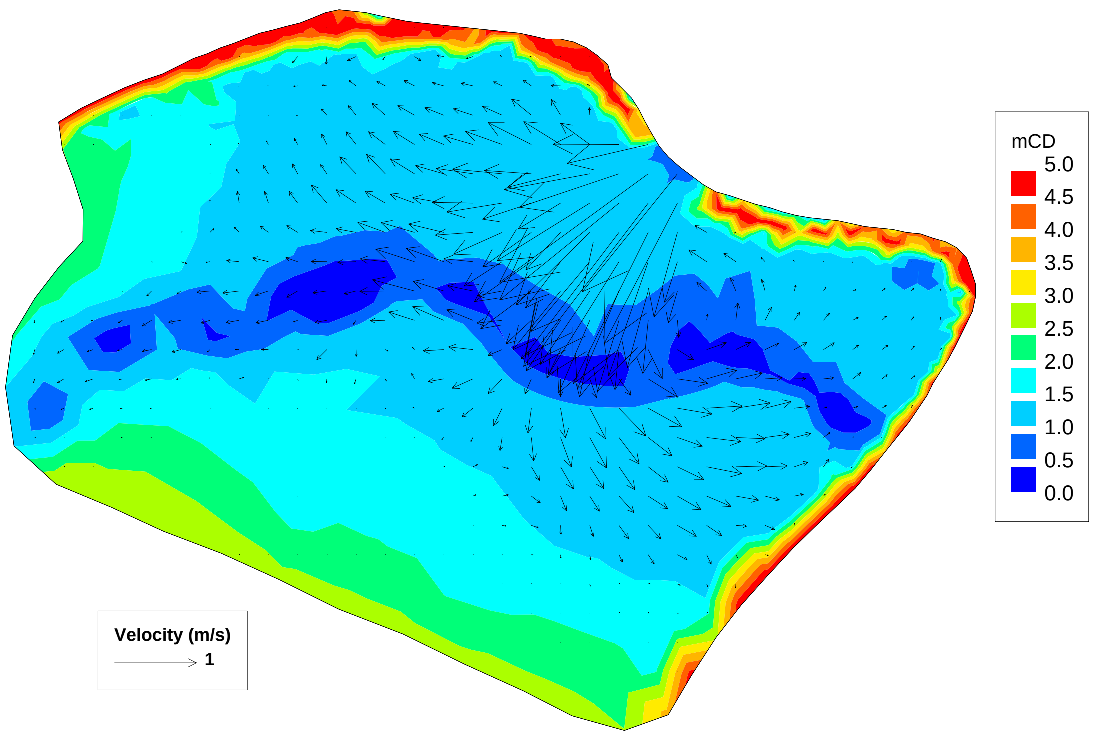

Figure 7.2 Circulations generated by flow through the entrance

7.2 EFFECT OF INCREASED GRID RESOLUTION

Here the baseline result was compared to the corresponding result with a significantly finer grid (the number of nodes in the model domain was increased threefold). The results are shown in Figure 7.3. It can be seen that to start with the increased resolution picks up the borrow channels aligned next to the northern and south western sea walls. The deposition in the enhanced resolution result occurs in these borrow dykes and in the east and west ends of the east/west drainage channel. For the low resolution prediction the deposition is more evenly spread in the east/west drainage channel with small amounts of deposition spread over most of the set back field.

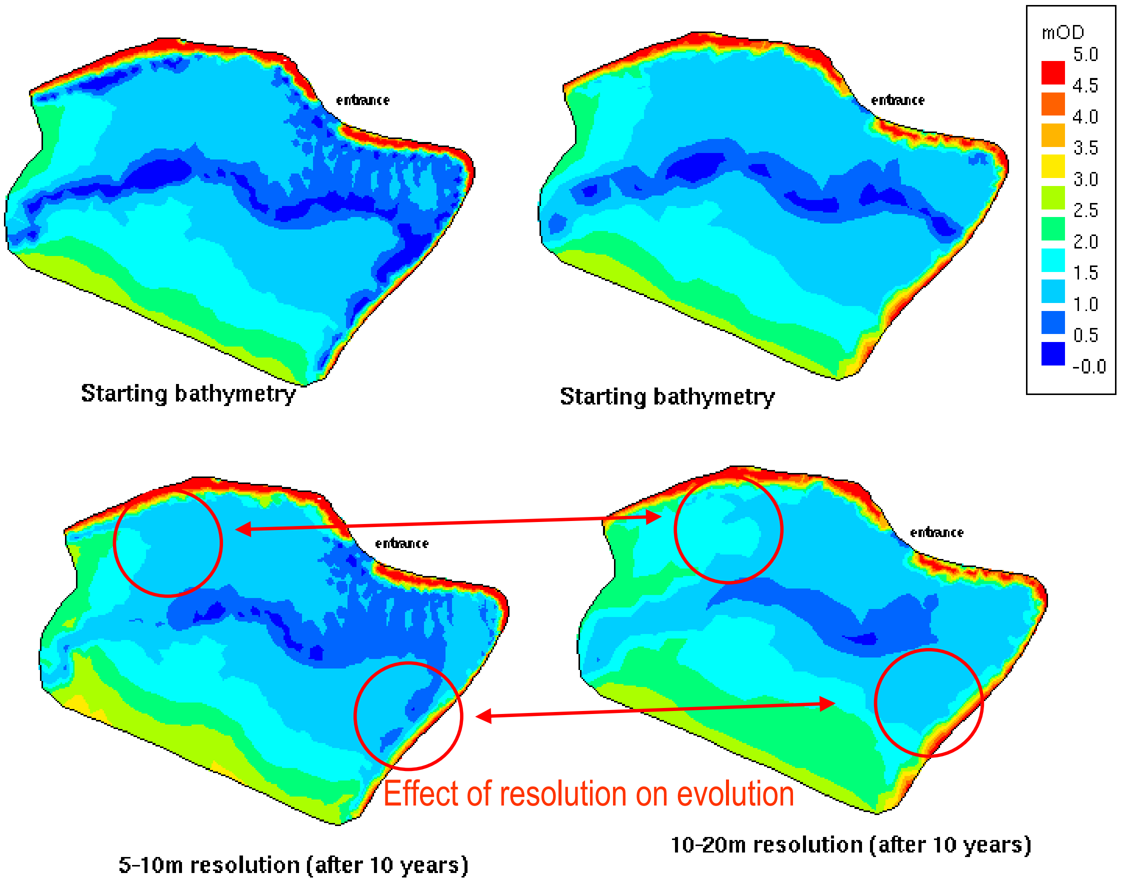

Figure 7.3 Effect of model grid resolution on predicted evolution

The result displayed in Figure 7.3 was compared to to the LiDAR/survey data (Figure 7.4) described in Chapter 6. For this the predicted evolution (increase in bed level) after 7 years (top right of Figure 7.4) was compared against the evolution (increase in bed level) calculated from the 2002 LiDAR and the 1995 bathymetry (bottom left of Figure 4). The comparison shows that the higher resolution model predicts all the main features of deposition and reproduces the magnitude of deposition reasonably.

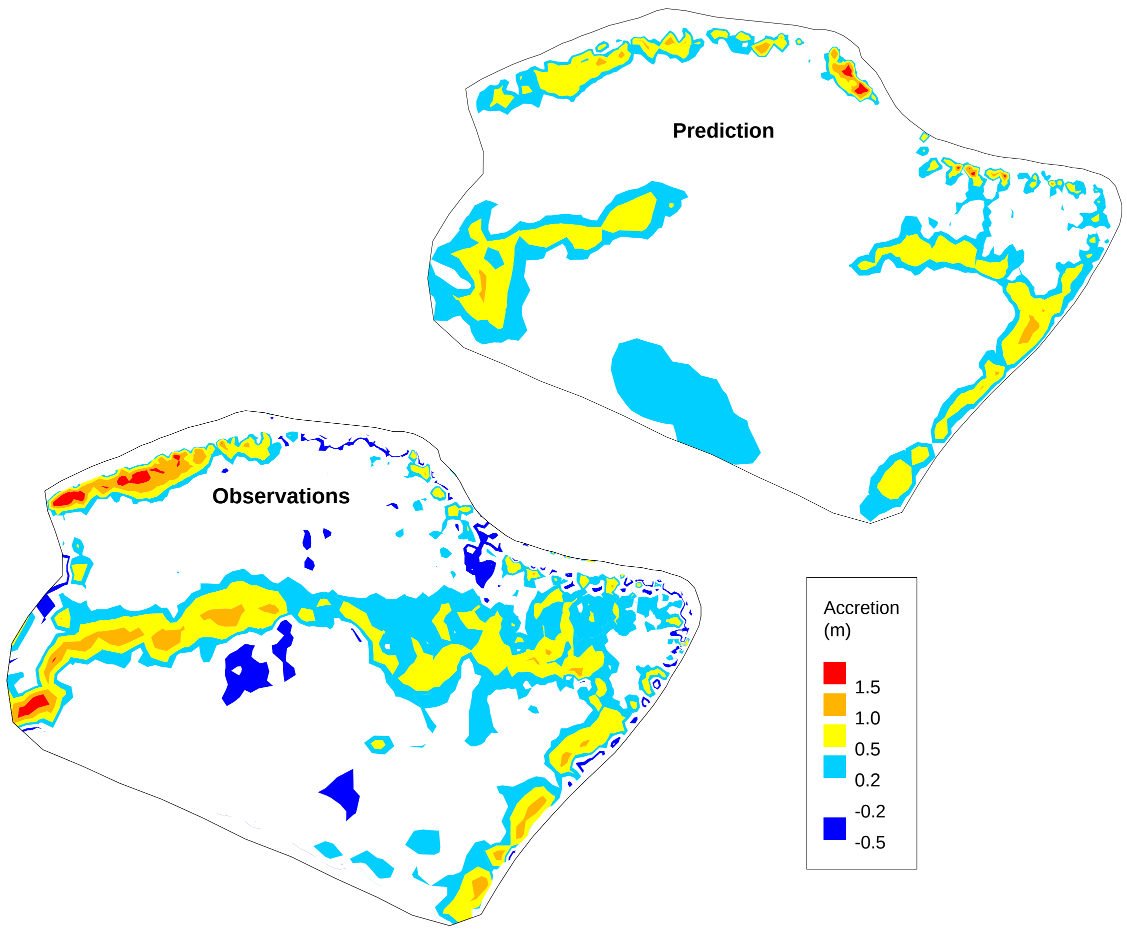

Figure 7.4 Comparison of prediction of 2002 bathymetry with data derived from a comparison of LiDAR and survey data

7.3 EFFECT OF CHANGE IN BED ROUGHNESS

The baseline result was compared to the corresponding result with significantly higher friction. The friction was increased to a physical roughness length of 3mm (compared to 0.3mm for the baseline result). The results of the two morphological predictions are compared in Figure 7.5. It can be seen that while the baseline result predicts deposition in the extremities of the field the high roughness result predicts deposition in the middle of the field. This makes sense as current flows are slower for the high roughness result and therefore sediment is not able to travel as far towards the extremities of the field.

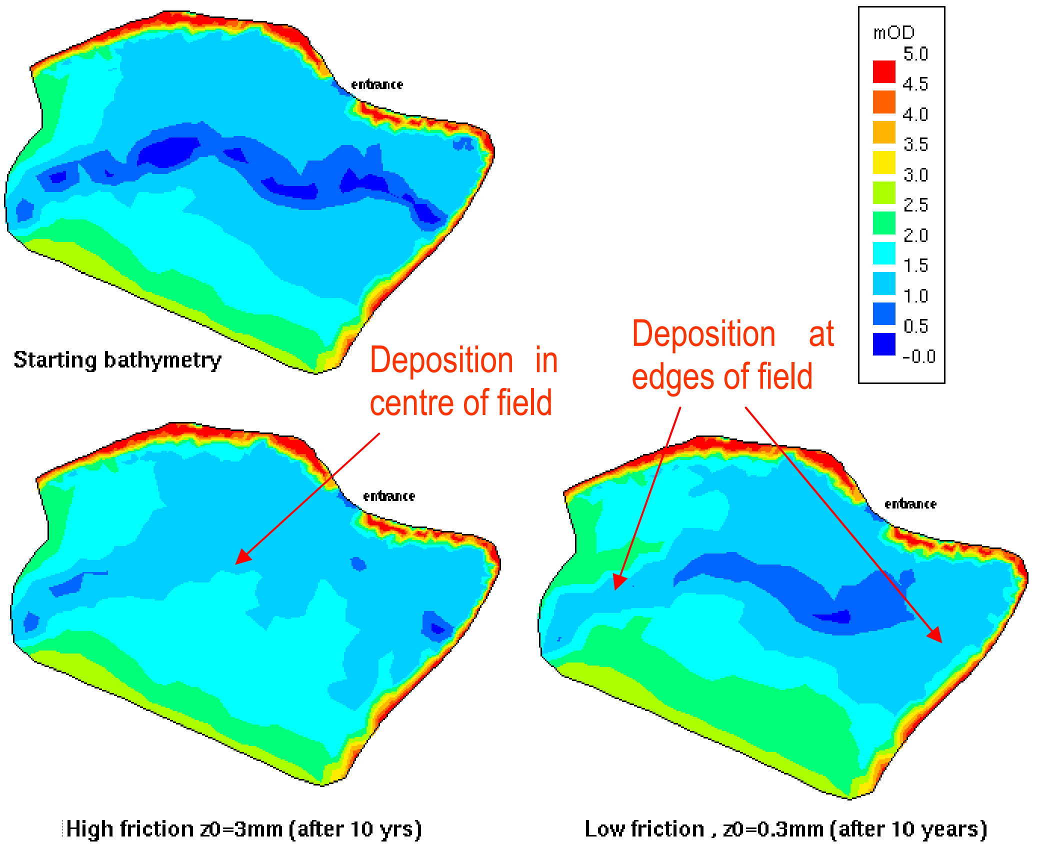
mOD
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
-0.0
entrance
Starting bathymetry
Deposition in centre of field
Deposition at edges of field
High friction z0=3mm (after 10 yrs)
Low friction , z0=0.3mm (after 10 years)

Figure 7.5 Effect of varying the model friction parameter on predicted evolution

7.4 EFFECT OF WAVES FROM OUTSIDE SET BACK FIELD

The effect of waves from outside the setback field is principally a function of the proportion of time that waves of a reasonable size can penetrate into the set back field.

Waves of a reasonable size will have a significant eroding effect on the shallower areas of the set back field but often, as at Tollesbury, only a small proportion of wind directions can generate waves that enter the setback field from outside, and of these only a small proportion generate a reasonable wave. The median wave height (H_s) entering Tollesbury set back field from outside is of the order of a few centimetres but extreme waves are larger than 0.5m (Chesher et al, 1995). Because of the small width of the entrance with respect to the size of the field the wave heights entering the setback field quickly reduce due to diffraction. This can be easily demonstrated using the diffraction diagrams produced by Goda (1985). Therefore the effect of waves entering the setback field from outside is constrained to the area close to the mouth, where currents, and hence levels of bed shear stress, are already very high. For this reason, at Tollesbury, it appears that outside waves do not play a major part in the evolution of the set back field. However, extreme waves from outside will have been important, and potentially still are important, in the development of the breach cross-section where episodic events - extreme waves and water levels, together with tidal currents, are the important drivers acting on the geotechnical integrity of the sea wall at either side.

8. Longer term evolution

8.1 PREDICTION OF EVOLUTION WITHOUT VEGETATION EFFECTS

On the basis of the calibration shown in Figure 6.6 a longer term simulation was undertaken. The results are shown in Figure 8.1. The longer term simulation suggests that there will be only a small area over the level of Mean High Water Neaps (MHWN), and hence little salt marsh, for 20 years and the subsequent development of area over HWN, and hence the growth of saltmarsh, will be slow. The area where salt marsh could occur appears to develop most strongly in the western part of the set back field. However, it must be understood that the uncertainty in this estimate is significant.

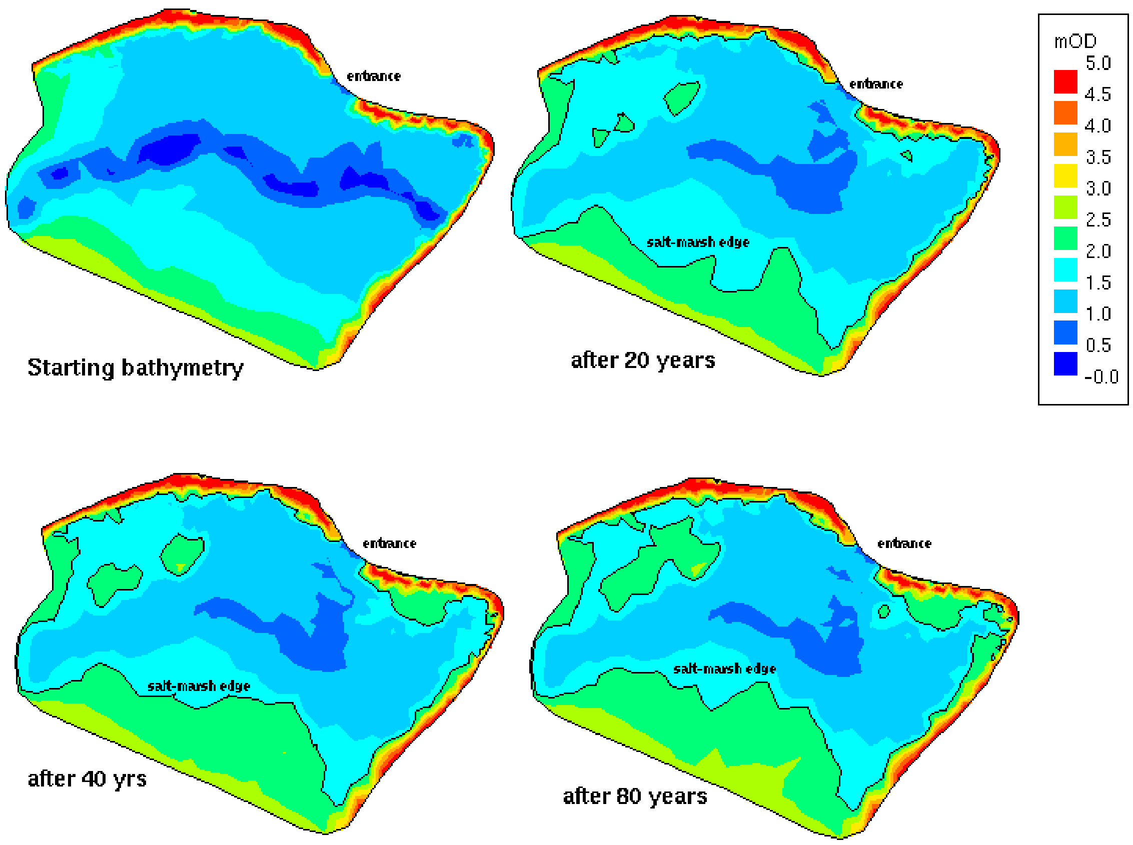

Figure 8.1 Predicted evolution of the set back field over the longer term

8.2 PREDICTION OF EVOLUTION INCLUDING VEGETATION EFFECTS

The effect of vegetation on morphology is complex but in crude terms the saltmarsh can be thought of as reducing wave action and current speeds and therefore acting as a depositionary environment for sediment.

To investigate the influence of salt-marsh vegetation on the evolution of the set-back field a simple sensitivity test was undertaken. Available sediment was assumed to deposit (all of it) on "salt marsh areas" (which were assumed to be all areas above High Water Neaps). This assumption will over-estimate the effect of saltmarsh on sediment deposition – since in reality the saltmarsh will not be 100% efficient as a sediment trap. However the test represents a useful indicator of the effect of vegetation.

Note that this effect only initiated when there are significant areas of saltmarsh – ie significant areas above MHWN.

The results of the vegetation test are shown in Figure 8.2. The vegetation effect (as modelled) produces different evolution but does not necessarily accelerate salt-marsh growth. In particular the saltmarsh tends to concentrate evolution in particular areas.

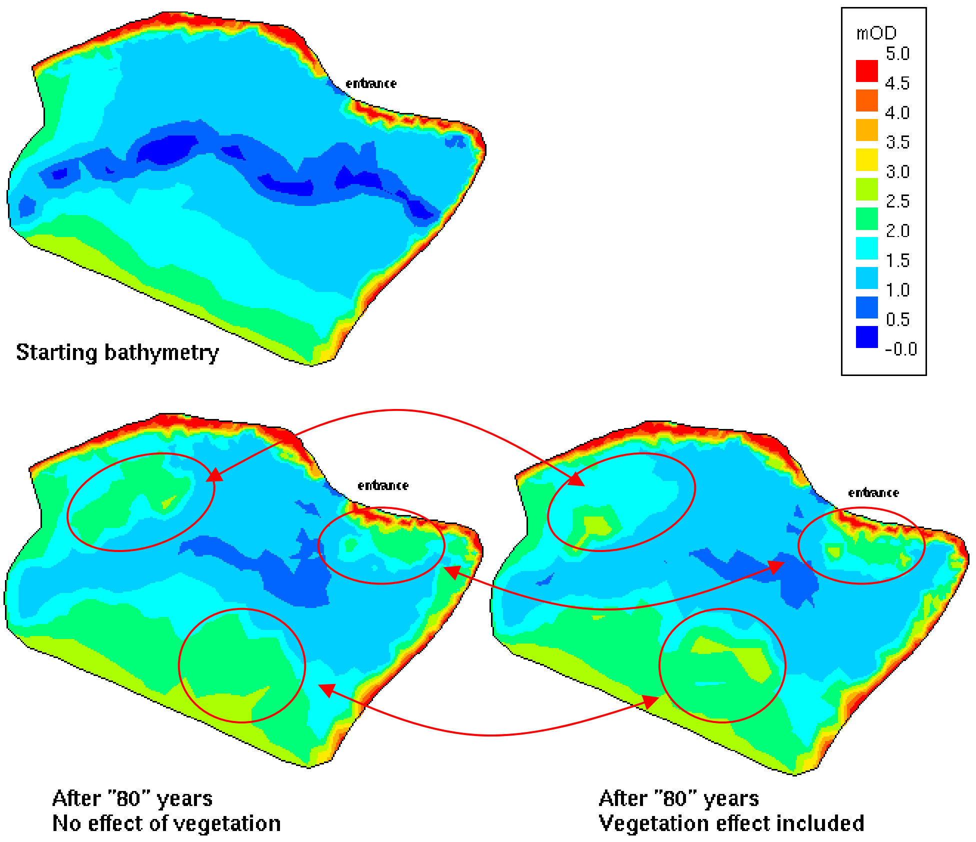

Figure 8.2 Predicted evolution of the set back field over the longer term with and without vegetation effects

8.3 PREDICTION OF EVOLUTION INCLUDING SEA LEVEL RISE

For this simulation the test described in Section 8.1 was repeated but this time the tidal levels imposed on the flow model were allowed to rise during the simulation at a rate of 6mm/yr. This rate of sea level rise broadly constitutes the Defra guidance on sea level rise over the next 50 years, although more recently this guidance has become more specific taking into account the predicted increase in sea level rise from the current rate (approximately 2mm/yr) to a much higher rate by the end of the century (15mm/yr).

A comparison of the predicted evolution without the effect of sea level rise and the predicted evolution taking sea level rise into account is shown in Figure 8.3. The figure shows that the setback field experiences more accretion as a result of sea level rise but that the pace of the accretion is not able to keep pace with the sea level rise and in particular the extent of salt marsh (defined by the level of high water neap tides which itself rises as the sea level rises) is reduced from 8.2ha (after 80 years of evolution without sea level rise) to 5.9ha (after 80 years of evolution with sea level rise). The equilibrium volume predicted by the model for the scenario without sea level rise is 2.3×10^5 m^3 while the the scenario with sea level rise is 2.7×10^5 m^3. The ASMITA analysis presented in Section 3.4 predicts that the dynamic equilibrium under the process of sea level rise would be around 16% higher than the scenario without sea level rise which agrees with the model prediction.

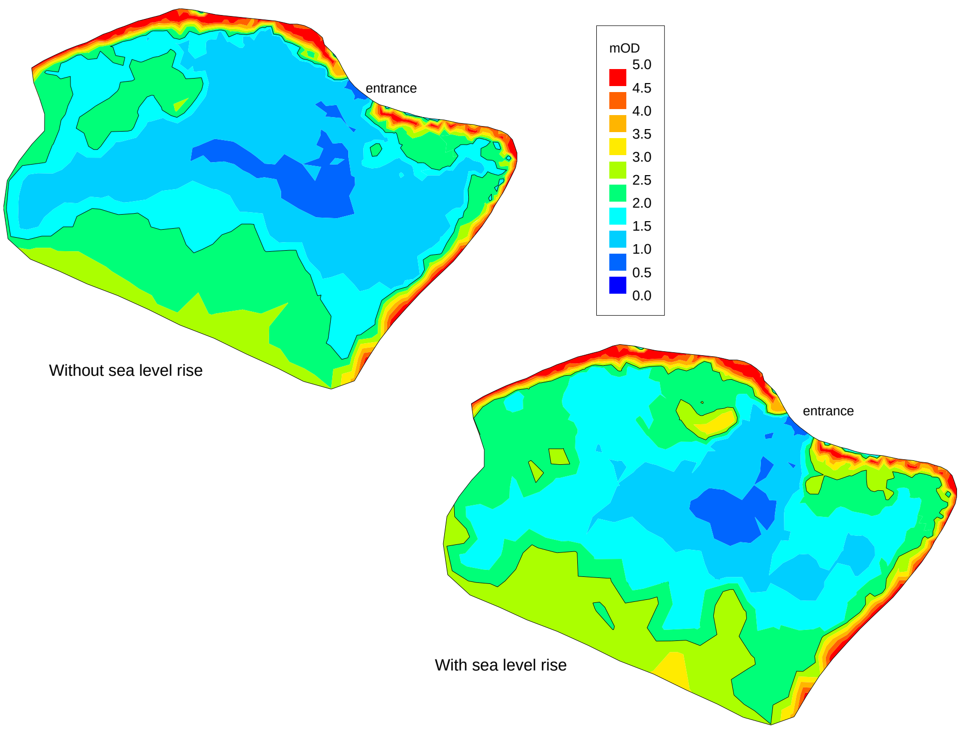

Figure 8.3 Comparison of predicted evolution of the setback site with and without sea level rise

Note that both the morphological model and the ASMITA prediction of the effects of sea level rise are done on the basis that all of the boundary conditions (sediment supply, tidal range, volume of water exchanged between the managed realignment site and Tollesbury Creek on each tide) remain constant during the period considered. In practice there is every reason to suppose these will change, either as a result of sea level rise or anthropogenic intervention over this period.

9. Conclusions

The model described in this report has been shown to be able to predict the evolution of a managed realignment site under the action of tides and waves and sediment supply.

Simple vegetation effects have been incorporated into the model and the model represents a framework for further developments into wave, vegetation and biological processes.

Validation of the managed realignment model was undertaken as far as possible on the basis of some point measurements made over a period of 6 years. However this type of data appears to be unsatisfactory for calibration and validation owing to the large spatial variation (even on a local scale) in bathymetric change. It is concluded that LiDAR data should be used wherever possible for this purpose.

The scope of this report did not consider the evolution of the breach itself as a result of extreme events, weathering and dessication but this should be considered as a potentially important effect. The wider breach will reduce the velocities through the breach and lengthen the flood tide. This would have the effect of introducing more sediment into the managed realignment site, but potentially reducing the amount of this sediment that settled in the site. In addition the wider breach could lead to larger waves entering the setback field from outside, which would tend to reduce the rate of accumulation and reduce the potential for salt marsh growth.

The sensitivity tests undertaken have considered how variations in waves, friction and model resolution affect the predicted evolution of the model. Longer simulations were used to see how the growth of saltmarsh itself affects the evolution of the setback field.

The results of the managed realignment model were similar to those of the ASMITA model except that the managed realignment model was able to supply the detail of evolution while the ASMITA model (in the linear form) was only able to provide an overall view of the evolution of the setback field.

Future research projects could consider incorporating the effects of widening of the breach on evolution of the site, scour of the foreshore of the approaches to the managed realignment and the inclusion of more detailed vegetation and/or biological effects on the site evolution. In addition it should be considered whether a more complex form of ASMITA, incorporating separate elements for mudflat and saltmarsh could provide a reliable estimate of evolution without the need for detailed modelling.

10. References

Chesher T.J., Price D.M. and Southgate H.N. (1995) MAFF Set Back Experiment, Tollesbury Creek, Report on model investigation of breaching scenarios, HR Wallingford Report SR 413, January 1995.

Cowell P J, Stive M J F, Niedoroda A W, de Vriend H J, Swift D J P, Kaminsky G M, and Capobianco M (2003) The Coastal-Tract (Part1): A conceptual approach to aggregated modelling of low-order coastal change, Journal of Coastal Research, Volume 19, number 4, pp812-827.

Di Silvio G (1989) Modelling the morphological evolution of tidal lagoons and their equilibrium configurations, XXIII Congress of the IAHR, Ottawa, August 1989.

Di Silvio G (1998) Interactions between marshes, channels and shoals in a tidal lagoon investigated by a 2-D morphological model, Interaction 3rd International Conference on Hydro-Science and -Engineering Brandenburg University of Technology at Cottbus Cottbus/Berlin, Germany, August 31 - September 3, 1998.

Di Silvio G and Gambolati G (1990) Two-dimensional model of the long-term morphological evolution of tidal lagoons, VIII International Conference on Computational Methods in Water Resources, Venice, Italy, June 11-15, 1990.

Dronkers J, van Os A.G. and Leendetse J J (1982) Predictive salinity modelling of the Oosterschelde with hydraulic and mathematical models, Delft Hydraulics Laboratory Publication Number 264, April 1982.

Eisma D and Djikema K S (1997) The influence of salt marsh vegetation on sedimentation. Pages 403-414 in D.Eisma, editor. Intertidal deposits. CRC Press, Boca Raton, Florida.

Fischer H B, List E J, Koh R C Y Imberger J and Brocks N K (1979) Mixing in inland and coastal waters, Academic Press, In. , New York.

French, J, Reeve, D & Owen, M (2002) Defra/EA FCD R&D Programme, Estuaries Research Programme Phase 2, Research Plan.

Goda, Y (1985) Random Seas and Design of Maritime Structures, University of Tokyo Press. Tokyo, Japan.

Programme, Phase 2 Research Plan FD2115, Report prepared for Defra and the Environment Agency, April 2002.

Gallappatti R and Vreugdenhil C B (1985) A depth-integrated model for suspended transport, Journal of Hydraulic Research, 23 (4), pp359-377.

HR Wallingford (2001) Sustainable Flood Defences. Monitoring of Retreat and Recharge Sites. Project Number MRD 21110, Abbott's Hall, Numerical Modelling, HR Wallingford Report EX 4367, August 2001.

Krone, R.B. (1962). Flume studies of the transport of sediment in estuarial shoaling processes final report. Hydraulic Engineering and Sanitary Engineering Research Laboratory, University of California, Berkeley, USA.

Partheniades E (1965) Erosion and deposition of cohesive soils, Proceedings of the American Society of Civil Engineers (ASCE), Volume 91 (HY1), pp105-139.

Soulsby R L (1997), Dynamics of Marine Sands A manual for practical applications, Thomas Telford Publications.

Soulsby R L (2006) Simplified calculation of wave orbital velocities, HR Wallingford Report TR 155.

Stive M J F Capobianco M, Wang Z B, Ruol P and Buijsman M C (1998) Morphodynamics of a tidal lagoon and the adjacent coast. In: Dronkers J and Scheffers M (Editors), Physics of Estuaries and Coastal Seas, pp397-407, Balkema, Rotterdam.

Stive, M. J. F., Wang, Z. B. (2003) Morphodynamic modelling of tidal basins and coastal inlets. In: Advances in coastal modelling. C. Lakhan (Ed.), Ch. 13, pp. 367–392.

Van Goor M A, Stive M J F, Wang ZB and Zitman T J (2002) Influence of relative sea level rise on coastal inlets and tidal basins. Proceedings, Coastal Dynamics ’01, ASCE, Lund, Sweden, pp 242-252.

Wang Z B, Karssen B, Fokkink R J and Langerak A (1998) A dynamic-empirical model for estuarine morphology, Physics of estuarine and coastal seas, Dronkers and Cheffers (eds), Balkema, Rotterdam, 1998, ISBN 905410965 3.

Appendices

Appendix 1 TELEMAC-2D Model Description

Description of model and main areas of application

TELEMAC-2D is a sophisticated flow model, developed by LNH in Paris, for free surface flows. It solves the 2D depth-integrated shallow water equations which are used to model flows in rivers, estuaries and seas. TELEMAC uses finite element techniques so that very flexible unstructured triangular grids can be used. It has been developed under a quality assurance system including application to a standard set of validation tests.

The model can simulate depth integrated tidal flows in estuaries and seas including the presence of drying banks. It can also simulate flows in rivers including turbulence structures resulting from flow obstructions and transcritical flows. The model has been used with up to 30000 elements.

The advantage of using finite elements lies primarily in the possibility of using a very flexible grid. This is superior to using an orthogonal curvilinear grid as the user has more control over grid refinement resulting in accurate representations of complex coastlines.

The applications of TELEMAC have included studies of ports, coastal management, floods in rivers, cooling water dispersion and infill of navigation channels.

Theoretical background and solution methods

TELEMAC solves the shallow water equations on an unstructured finite element grid (usually with triangular elements). The various variables (bed elevation, water depth, free surface level, u and v velocity components) are defined at the nodes (vertices of triangles) and linear variation of the water and bed elevation and of the velocity within the triangles is assumed.

When the model is used a timestep is chosen and the computation is advanced for the required number of timesteps. There is no particular limit on the timestep for a stable computation but it is best to ensure that the Courant number based on propagation speed is less than about 10. It is found that if the solution is nearly steady then few computational iterations are required at each step to achieve the required level of accuracy, which in TELEMAC is computed according to the actual divergence from the accurate solution. The computation at each timestep is split into two stages, an advective step and a propagation-diffusion step.

The advective step is computed using characteristics or streamwise upwind Petrov-Galerkin. The characteristic step makes it possible for the code to handle such problems as flow over a bump giving rise to locally supercritical flow and eddies shedding behind flow obstructions.

The finite element method used is based on a Galerkin variational formulation. The resulting equations for the nodal values at each timestep are solved using an iterative method based on pre-conditioned conjugate gradient (pcg) methods so that large problems are solved efficiently. Several pcg solvers are coded and a selection is available to the user. The complete matrix is not assembled, rather an element by element method is used so that most of the operations are carried out on the element matrices; this is computationally more efficient, both in speed of execution and in memory requirements. Rather than using Gauss quadrature exact analytical formulae are used for the computation of matrices. Symbolic software was used to draw up the formulae used. The software makes it possible to carry out a second iteration of the solution at each timestep in order to represent the nonlinear terms in a time centred way, otherwise these terms are treated explicitly.

Boundary conditions are applied at solid boundaries where a zero normal flow and slip or nonslip boundary conditions are applied. At open boundaries a selection of possibilities can be invoked depending on whether the flow is subcritical or supercritical, or whether a wave absorbing boundary using a Riemann invariant is needed. A water discharge along a boundary segment can also be applied and the software distributes the flow along the segment chosen. This facility is valuable when running models of river reaches and the discharge in a cross-section may be known rather than the velocity at each point in the cross-section.

The model can be run with a Cartesian grid for modelling rivers, estuaries and small areas of sea, with the ability to apply a uniform Coriolis parameter, or on a spherical grid for larger areas of sea in which case the Coriolis parameter is computed from the latitude at each node. The effect of a wind blowing on the water surface and causing a setup or wind induced current or of an atmospheric pressure variation causing an inverted barometer effect can be included, as can a k-epsilon model of turbulence if required.

The bed friction can be specified via a Chezy, Strickler or linear coefficient, or a Nikuradse roughness length. A variable friction coefficient over the model area is a possibility. Sidewall friction can also be included if required. Viscosity can be imposed as a given eddy viscosity value or a k-epsilon model can be used.

TELEMAC-2D includes the capability to simulate the transport of a tracer substance. The tracer is again computed using an advective step followed by a propagation/diffusion step. Tracer boundary conditions can be applied at model inflow boundaries. The tracer calculation has been used in order to simulate cooling water dispersion and mud transport. Sources of water and/or of tracer can be specified in terms of the discharge required and the x and y coordinates of the location.

Model Input

TELEMAC requires as input a finite element grid of triangles covering the area to be modelled. Bathymetric data from which the bed elevation at each node can be computed is also required. A file of keyword values is used to steer the computation (supplies bed roughness, timestep, duration of run etc).

The finite element grid may be generated using the MATISSE grid generator, which is part of the TELEMAC suite. The software STBTEL (part of the TELEMAC suite) is used to read the output file from the grid generation software. The bathymetry is input using a digitising tablet and the SINUSX software is used to capture the bathymetry data. The data is stored in a form to be read into the TELEMAC system and depths interpolated to the model nodes.

SINUSX is a powerful interactive graphical software that can be used to check and amend the input data. Bathymetric curves can be duplicated, deleted, smoothed, moved etc.

Model Output

The user can select a range of output parameters including u and v velocity, u and v discharge, water level, bed level, water depth, tracer concentration and Froude number.

The TELEMAC output is contained in a single binary file which can be input to the graphics post-processor RUBENS. A listing file contains reflection of the input keywords and information on timestep reached, number of iterations to convergence etc. This file can be used to monitor the progress of a run.

Results from any part of the TELEMAC system are processed using the interactive graphics system RUBENS. This is a powerful and friendly environment where colour and black and white figures can be produced interactively. By pointing and clicking time history plots, cross sections, vector plots and contour plots of any parameter at any position can be produced. Moreover parameters other than those input can be calculated in RUBENS and plotted.

Appendix 2 Subief-2D model description

Introduction

SUBIEF code makes it possible to study the transport of one or more tracers within a two-dimensional free surface ow. These tracers can be diluted or in suspension in water. In this last case, SUBIEF can calculate their deposition on the bottom and their resuspension by erosion. The tracers can possibly inter-react; not only can they be transported by the current but also they can vary with time. For each tracer, we can thus take into account various physico-chemical phenomena: transport by the current, diffusion, deposition, erosion, reaction, and define the necessary source terms.

SUBIEF was developed from the finite element structure of TELEMAC-2D. It is a two-dimensional code: the concentration of each transported variable is supposed constant vertically over the water column. SUBIEF is a code uncoupled from the hydrodynamics. Hydrodynamic calculations are initially carried out by TELEMAC-2D. SUBIEF reads the results file of TELEMAC-2D, to build the convective field in which to solve one or more transport equations. Nevertheless, compatibility between the hydrodynamic computation and the transport computation are of primary importance. This is true in particular for the conservation of the mass of tracer, which is related to the conservation of the water mass. The quality of the results of SUBIEF thus depends strongly on the quality on the hydro dynamic computation carried out before, even if one models the transport of a simple passive tracer. For obvious reasons, we cannot hope to correctly reproduce the bottom evolution, starting from a poor quality computation of the current field.

Theoretical Aspects

SUBIEF-2D solves the transport, deposition, and erosion of tracer in following equation:

$$\frac{\partial c}{\partial t} + \vec{u}.\nabla c = div(\vec{K}.\nabla c) + \frac{Q_e - Q_d}{h} + \sum_{pointsr}^{sources} (c_s - c)\frac{Q_L(r)}{\int_\Omega h\psi_r d\Omega}$$

where:

c	Concentration of tracer in suspension (g/l)
K	Coefficient of dispersion (m^2/s)
u	Velocity of flow (m/s)
h	Depth of water (m)
Q_e	Erosion flux ($kgm^{-2}s^{-1}$)
Q_d	Deposition flux ($kgm^{-2}s^{-1}$)
c_s	Concentration of tracer at point source r (g/l)
Q_L	Liquid discharge of the source r (m^3/s)
S	Sources or sinks to be defined by the user (g/l/s)
ψ_r	Basis function (finite elements) at point source

The deposition and erosion fluxes are calculated using the Krone and Partheniades formulae respectively:

$$Q_d = W_c c \left[1 - \left(\frac{u^*}{u_d^*}\right)^2\right] \quad \text{and}$$

$$Q_e = M \left[\left(\frac{u^*}{u_e^*}\right)^2 - 1\right]$$

Where:

W_c	Settling Velocity (m=s).
M	Constant of Partheniades.
u^*_d ; u^*_e	Critical velocities of deposition and erosion (m/s).
u^*	Velocity bottom friction (m/s).

The evolution of the bottom (i.e. the variation of the bathymetry) is therefore:

$$E = \frac{Q_d - Q_e}{C_{sf}} DT$$

Where:
DT is the time step.
C_{sf} is the concentration of the deposit on the bed (kg/m^3).

The equation of the tracer can be solved several times successively within the same timestep. One can thus model the behaviour of several tracers. For each tracer, one can choose not to take into account the term of advection by the current or dispersion.

The schematisation of the water column used by SUBIEF is summarised in Figure 1.

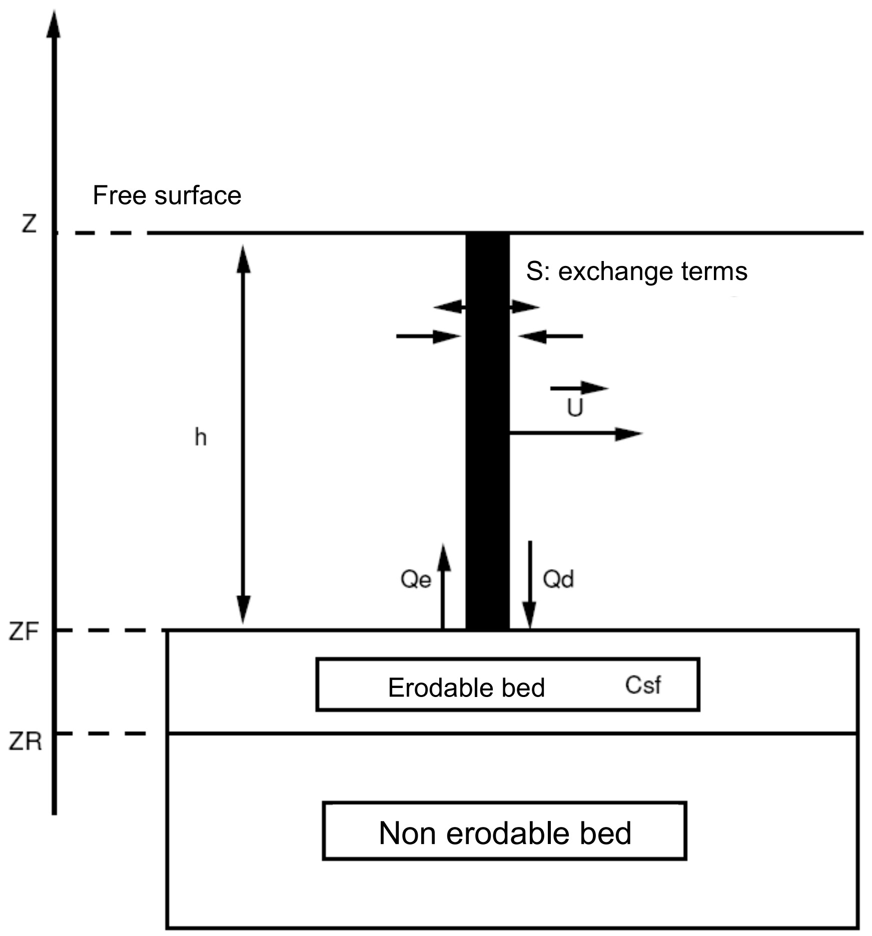

Figure 1 Schematisation of the water column used by SUBIEF-2D

Appendix 3 Modifications to the SUBIEF-2D model

```
C                       *****************
                        SUBROUTINE SEDINI
C                       *****************
```

Need to include in declaration

```
      double precision conc(npoin)
      logical found
```

and before setting ESOMT include following text

```
      inquire(file='../sed.xyz',exist=found)
      if (.not.found) then
       do i=1,10
         write(*,*) 'FATAL ERROR: no sediment xyz file'
       enddo
       stop
      endif

        open(20,status='old',file='../sed.xyz')
        rewind(20)
        do i=1,npoin
            read(20,*) conc(i),ESOMT%R(i)
            print *,'sed.xyz ',conc(i),ESOMT%R(i)
        enddo
```

and to set initial concentration conditions

```
C AFFECTATION DES CONDITIONS INITIALES
C ---------------------------------------------------------------------
      DO 60 I = 1, NTRAC
      DO 60 K = 1, NPOIN
c     CS%ADR(I)%P%R(K) = CINI(INT(TRA01%R(K)),I)
      if(i.eq.1) CS%ADR(I)%P%R(K)=conc(K)

60    CONTINUE
```

```
C                       *****************
                        SUBROUTINE FLUSED
C                       *****************
```

This routine is included in full with changes highlighted

```
     *( FLUDP  , FLUER  , TSIMP  , CS    , HN     , Q      , TRA01  ,
     *  TRA02  , GRAV   , CHESTR , XWC   , XM     , VITCD  , WCVAR  ,
     *  WC     ,VITCE   , KFROT  , NPOIN , HMIN , ZF , X , Y,
     *  cequ, pkstress, ESOMT)
C
C**********************************************************************
C  SUBIEF VERSION 5.1    13/12/00             C MOULIN (LNH) 30 87 83 81
C
C
C**********************************************************************
C
C     FONCTION  :  CE SOUS-PROGRAMME CALCULE :
C                    - LE FLUX DE DEPOT
C                    - LE FLUX D'EROSION
```

```
C                    - LA PARTIE IMPLICITEE DU FLUX DE DEPOT
C                     ( SOIT :   WC * ( 1 - (U*/U*d)**2 )
C
C REMARQUE : SI L'ON NE VEUT PAS CALCULER TSIMP, LE REMPLACER PAR FLUDP
C --------   DANS L'APPEL A FLUSED.
C
C-----------------------------------------------------------------------
C                             ARGUMENTS
C .________________.____.______________________________________________
C !      NOM       !MODE!                   ROLE                       !
C !________________!____!______________________________________________!
C !    FLUDP       !<-- !  FLUX DE DEPOT                               !
C !    FLUER       !<-- !  FLUX D'EROSION                              !
C !    TSIMP       !<-- !  PARTIE IMPLICITE DU FLUX DE DEPOT           !
C !    CS          ! -->!  CONCENTRATION AU TEMPS N                    !
C !    HN          ! -->!  HAUTEUR D'EAU AU TEMPS N                    !
C !    Q           ! -->!  DEBIT AU TEMPS N                            !
C !    TRA01,TRA02 !<-->!  TABLEAUX DE TRAVAIL                         !
C !    GRAV        ! -->!  COEFFICIENT DE GRAVITE   (9.81 M.S-2)       !
C !    CHESTR      ! -->!  COEFFICIENT DE FROTTEMENT DU FOND PAR POINT !
C !    XWC         ! -->!  VALEUR DE LA VITESSE DE CHUTE (CONSTANTE)   !
C !    XM          ! -->!  CONSTANTE DE PARTHENIADES                   !
C !    VITCD       ! -->!  VITESSE CRITIQUE DE DEPOT                   !
C !    WCVAR       ! -->!  LOGIQUE VRAI SI VITESSE DE CHUTE VARIABLE   !
C !    WC          ! -->!  VITESSE DE CHUTE VARIABLE (TABL. DIM. NPOIN)!
C !    VITCE       ! -->!  VITESSE CRITIQUE D'EROSION                  !
C !    KFROT       ! -->!  LOI DE FROTTEMENT   (1:CHEZY  2:STRICKLER)  !
C !    NPOIN       ! -->!  NOMBRE DE POINTS DU MAILLAGE                !
C !    HMIN        ! -->!  HAUTEUR D'EAU MINIMALE                      !
C !________________!____!______________________________________________!
C MODE : -->(DONNEE NON MODIFIEE), <--(RESULTAT), <-->(DONNEE MODIFIEE)
C-----------------------------------------------------------------------
C
C PROGRAMME APPELANT : PRESOU , EVOLUT
C PROGRAMMES APPELES : OV , OVD , PLANTE
C
C***********************************************************************
C
      USE BIEF
      IMPLICIT NONE
      INTEGER LNG,LU
      COMMON/INFO/LNG,LU
C
c     TYPE(BIEF_MESH) :: MESH
      TYPE(BIEF_OBJ) :: HN,Q,TRA01,TRA02,CHESTR

      INTEGER NPOIN
C
      DOUBLE PRECISION  FLUDP(NPOIN) ,CS(NPOIN), ZF(NPOIN)
      DOUBLE PRECISION  FLUER(NPOIN) ,WC(NPOIN)    ,TSIMP(NPOIN)
      DOUBLE PRECISION  X(NPOIN)     , Y(NPOIN)
C
      LOGICAL WCVAR
C
      DOUBLE PRECISION  A1    , GRAV ,VITCD , VITCE , XM , XM2
      DOUBLE PRECISION  XWC , HMIN, cequ(npoin), pkstress(npoin),bit
      DOUBLE PRECISION  ESOMT(npoin)
      INTEGER           I    , KFROT
C
      INTRINSIC SQRT , MAX
C =====================================================================
C
```

```
C  CALCUL DE LA VITESSE CRITIQUE :
C  -----------------------------
C
      A1 = 7.D0/6.D0
      CALL OS( 'X=C      ' , TRA02  , HN , HN , SQRT(GRAV) )
      CALL OS( 'X=Y/Z     ', TRA01 , TRA02 , CHESTR , 0.D0 ,
     *                 IOPT=2,INFINI=0.D0,ZERO=1.D-3)
C
C
      IF (KFROT.EQ.1) THEN
        CALL OS( 'X=CXY/Z  ' , TRA01 , Q , HN , 1.D0 ,
     *                 IOPT=2,INFINI=0.D0,ZERO=1.D-3)
C
      ELSEIF (KFROT.EQ.2) THEN
        CALL OS( 'X=Y**C  ', TRA02 , HN , HN    , A1 )

        CALL OS('X=CXY/Z ', TRA01 , Q  , TRA02 , 1.D0,
     *                 IOPT=2,INFINI=0.D0,ZERO=1.D-3)
C
      ELSE
        IF (LNG.EQ.1) WRITE(LU,110)
        IF (LNG.EQ.2) WRITE(LU,111)
        CALL PLANTE(0)
        STOP
      ENDIF

      bit=0.0001

C

C  CALCUL DE FLUDP ET TSIMP :
C  ------------------------
C
      DO 100 I=1,NPOIN
C
       if(HN%R(i).gt.0.05) then
        IF (CS(I).GT.(cequ(i)+bit)) THEN
            FLUDP(I) =  (CS(i)-cequ(i))*XWC
C
        ELSE
C
          FLUDP(I) = 0.D0
C
        ENDIF
       else

         FLUDP(I) = 0.D0
C
        ENDIF
C
 100  CONTINUE
C
C
C  CALCUL DE FLUER :
C  ---------------
C

      DO 200 I=1,NPOIN
C

       if(HN%R(i).gt.0.05) then
       IF (CS(I).LT.(cequ(i)-bit)) THEN
```

```
          FLUER(I) = (cequ(i)-CS(i))*XWC
        else
          FLUER(I) = 0.D0
        ENDIF
       else
          FLUER(I) = 0.D0
       ENDIF


200   CONTINUE

C
110    FORMAT(1X,'FLUSED : LOI DE FROTTEMENT INCONNUE')
111    FORMAT(1X,'FLUSED : UNKNOWN LAW FOR FRICTION')
C
      RETURN
      END


C                        *****************
                         SUBROUTINE EVOLUT
C                        *****************
```

The only change here is the call with the cequ, pkstress variables and the changed call to FLUSED

```
     *( E      , CS1    , CS     , FLUDP  , FLUER  , HN     , ZF   ,
     *  Q      , TRA02  , TRA03  , TRA04  , GRAV   , ZR     ,
     *  XWC    , XM     , VITCD  , VITCE  , NPOIN  , WCVAR  , TRA05,
     *  WC     , CSF    , KFROT  , CHESTR , DT     , TETA   , HMIN ,
     *  MESH, cequ, pkstress)
C
```

And

```
      CALL FLUSED
     *       ( FLUDP%R ,FLUER%R ,FLUDP%R, CS1%R,
     *         HN ,Q ,TRA02 ,TRA03 ,
     *         GRAV  ,CHESTR,XWC  ,XM  ,VITCD ,WCVAR ,WC%R ,
     *         VITCE , KFROT, NPOIN, HMIN , ZF , MESH%X%R ,
     *         MESH%Y%R, cequ, pkstress, E%R)


C                        ******************
                         SUBROUTINE SUBIEF2D
C                        ******************
C
```

Need to include in declarations

```
      double precision cequ(npoin), pkstress(npoin), old_geom(npoin)
```

and at end of routine

```
      open(13,status='new',file='../endgeom.xyz')
      do i=1,npoin
        write(13,*) ZF%R(i)-old_geom(i),ZR%R
c       print *,ZF%R(i)-old_geom(i)
      enddo
```

```
C                      *****************
                       SUBROUTINE PRESOU
C                      *****************
```

The only change here is the call with the cequ, pkstress variables and the changed call to FLUSED

```
     *( FLUDP  , FLUER , TSIMP , SMH , CS , HN ,Q,TRA02,
     *  TRA03  , TSEXP , GRAV  , CHESTR, XWC   , XM     , VITCD    ,
     *  WCVAR  , WC    , VITCE , KFROT , NPOIN , ZF ,ZR , CSF , DT ,
     *  TETA   , HMIN  , X , Y ,Z  ,
     *  NELMAX, IELM   , ITRA     ,
     *  NREJET , DSCE  , ISCE  , DCHIM , LV    ,
     *  MESH   , MSK   , MASKEL , OPTSOU, cequ , pkstress, E)
C
```
and
```
          CALL FLUSED
     *      ( FLUDP%R ,FLUER%R ,TSIMP%R, CS%ADR(ITRA)%P%R,
     *        HN ,Q ,TRA02 ,TRA03 ,
     *        GRAV  ,CHESTR,XWC  ,XM  ,VITCD ,WCVAR ,WC%R ,
     *        VITCE , KFROT, NPOIN, HMIN , ZF, X, Y, cequ, pkstress,
     *        E%R)
C
C                        *****************
                         SUBROUTINE DISPER
C                        *****************
```

This routine is included in full with highlighted alterations

```
     *( DISP, KX , KY , KZ , XCONV , YCONV , H , CHESTR , ZF , TRA01 ,
     *  TRA02 , X ,Y , NPOIN , AT  , DT  , GRAV  ,
     *  XKX   , XKY  , OPTDIF , nelem, cequ, pkstress)
C
C***********************************************************************
C  SUBIEF VERSION 5.1    13/12/00            C MOULIN (LNH) 30 87 83 81
C
C***********************************************************************
C
C     FONCTION  : CALCUL DES COEFFICIENTS DE DISPERSION
C
C
C----------------------------------------------------------------------
C                              ARGUMENTS
C .________________.____.______________________________________________.
C !      NOM       !MODE!                   ROLE                       !
C !________________!____!______________________________________________!
C !   KX,KY,KZ     !<-- ! COEFF DU TENSEUR  DE DISPERSION (DIM. NPOIN) !
C !   XCONV,YCONV  !<-->! COMPOSANTES DE LA VITESSE                    !
C !   H            !<-->! HAUTEUR D'EAU                                !
C !   ZF           !<-->! COTE DU FOND                                 !
C !   S            !    ! SURFACE LIBRE (INUTILISEE)                   !
C !   TRA01,TRA02  ! -->! TABLEAUX DE TRAVAIL                          !
C !   X,Y          ! -->! COORDONNEES DES POINTS DU MAILLAGE           !
C !   NPOIN        ! -->! NOMBRE DE POINTS DU MAILLAGE                 !
C !   AT           ! -->! TEMPS                                        !
C !   DT           ! -->! PAS DE TEMPS                                 !
C !   GRAV         ! -->! ACCELERATION DE LA PESANTEUR                 !
C !   XKX,XKY      ! -->! DISPERSION SELON X ET Y (CONSTANTES)         !
C !   OPTDIF       ! -->! OPTION POUR LA DISPERSION :                  !
C !                !    ! 1 : KU ET KT CONSTANTS                       !
C !                !    ! 2 : DISPERSION DU TYPE                       !
C !                !    !     K * UETOILE * H (EN CHAQUE POINT)        !
```

```
C !                 !    ! 3 : DISPERSION RELUE DANS LE RESULTAT DE     !
C !                 !    !     TELEMAC-2D                               !
C !_________________!____!______________________________________________!
C MODE : -->(DONNEE NON MODIFIEE), <--(RESULTAT), <-->(DONNEE MODIFIEE)
C-----------------------------------------------------------------------
C SOUS-PROGRAMME APPELANT : SUBIEF
C SOUS-PROGRAMMES APPELES :
C **********************************************************************
C
      USE BIEF
      IMPLICIT NONE
      INTEGER LNG,LU
      COMMON/INFO/LNG,LU
C
      TYPE(BIEF_OBJ) :: TRA01,TRA02,KX,KY,KZ,DISP
      INTEGER NPOIN
C
      DOUBLE PRECISION   H(NPOIN)
      DOUBLE PRECISION   XCONV(NPOIN) , YCONV(NPOIN)
      DOUBLE PRECISION   ZF(NPOIN)    , CHESTR(NPOIN)
      DOUBLE PRECISION   X(NPOIN)     , Y(NPOIN)
      DOUBLE PRECISION   AT , GRAV    , DT , XKX , XKY
      DOUBLE PRECISION   COST , SINT , NORMV , UETH , PUIS , C

      integer chdisp,nelem,i
      double precision cequ(npoin),pkstress(npoin)
      double precision Dx(npoin),Dy(npoin)
      character*32 dispfile
      logical found
C
      INTEGER    OPTDIF , K
      INTRINSIC  SQRT
C
C     ***     DECLARATIONS DUES A IMPLICIT NONE   ***
C
C
C-------------------------------------------------------------------
C ALLOCATION DES TABLEAUX DE DISPERSION LONGITUDINALE (TRA01)
C                                    ET TRANSVERSALE (TRA02)
C ------------------------------------------------------------------
C
      IF ((OPTDIF.EQ.2).OR.(OPTDIF.EQ.1)) THEN
C
        IF (OPTDIF.EQ.2) THEN
C LA DISPERSION EST DU TYPE XKX * U(CISAILLEMENT) * H
          PUIS = 5.D0/6.D0
          DO 10 K=1,NPOIN
            NORMV = SQRT(XCONV(K)**2 + YCONV(K)**2)
            UETH = SQRT(GRAV) * (H(K)**PUIS) * NORMV / CHESTR(K)
            TRA01%R(K) = XKX * UETH
            TRA02%R(K) = XKY * UETH
10        CONTINUE
C
        ELSE

c       read dispersion file
      dispfile='../dispcequ.res'
        chdisp=30
      inquire(file=dispfile,exist=found)
c     read *,i
      if (.not.found) then
111     write(*,*) 'FATAL ERROR: no dispersion file'
        stop
      endif
        open(chdisp,file=dispfile,form='UNFORMATTED')

        call read_tel_hdr(chdisp,npoin,nelem)

        call readflow(chdisp,Dx,Dy,cequ,pkstress,npoin,nelem)

C
c         CALL OS('X=Y     ', TRA01  , D , TRA02 , XKX )
c         CALL OS('X=Y     ', TRA02  , D , TRA01 , XKY )

        do i=1,npoin
             TRA01%R(i)=dsqrt(Dx(i)**2+Dy(i)**2)
             TRA02%R(i)=0.1*TRA01%R(i)
        endif

        enddo
C
        ENDIF
C
C
C CALCUL DES COEFFICIENTS DE DISPERSION
C -------------------------------------
C
C TRA01 CONTIENT LA DISPERSION DANS LE SENS DU COURANT
C TRA02 CONTIENT LA DISPERSION TRANSVERSALE
C
        DO 20 K=1,NPOIN
C
c change for morphology: get NORMV and COST/SINT from Dx/Dy
           NORMV = SQRT(Dx(K)**2 + Dy(K)**2)
C
           IF (NORMV.GE.1.D-6) THEN
c            COST = XCONV(K)/NORMV
c            SINT = YCONV(K)/NORMV
c NB TRA01%R(i) is magnitude of dispersion
             COST = Dx(K)/TRA01%R(K)
             SINT = Dy(K)/TRA01%R(K)

           ELSE
             COST = 0.D0
             SINT = 0.D0
           ENDIF
C
       KX%R(K) = (TRA01%R(K) - TRA02%R(K))*(COST**2)  +  TRA02%R(K)
       KY%R(K) = (TRA02%R(K) - TRA01%R(K))*(COST**2)  +  TRA01%R(K)
       KZ%R(K) = (TRA01%R(K) - TRA02%R(K))*COST*SINT
C
20      CONTINUE
C
      ELSEIF(OPTDIF.EQ.3) THEN
C
        CALL OS( 'X=Y     ' , KY , KX    , TRA01 , C     )
        CALL OS( 'X=C     ' , KZ , TRA01 , TRA01 , 0.D0  )
C
      ELSE
        IF (LNG.EQ.1) WRITE(LU,30) OPTDIF
        IF (LNG.EQ.2) WRITE(LU,31) OPTDIF
30      FORMAT(1X,'DISPER : OPTION POUR LA DISPERSION NON PREVUE: ',1I6)
31      FORMAT(1X,
     *  'DISPER : OPTION FOR THE DISPERSION NOT AVAILABLE : ',1I6)
        CALL PLANTE(0)
        STOP
      ENDIF
C
           CALL OV_2('X=Y     ',DISP%R,1,KX%R,1,KX%R,1,C,
     *                          DISP%MAXDIM1,DISP%DIM1)
           CALL OV_2('X=Y     ',DISP%R,2,KY%R,1,KY%R,1,C,
     *                          DISP%MAXDIM1,DISP%DIM1)
           CALL OV_2('X=Y     ',DISP%R,3,KZ%R,1,KZ%R,1,C,
     *                          DISP%MAXDIM1,DISP%DIM1)
      RETURN
      END


c ************************************************************

      subroutine read_tel_hdr (chan,npoin,nelem)


c Function:  reads the header part of a Seraphin format TELEMAC results
c            file.


c-----------------------------------------------------------------------
c    Name     |   i/o    |    Description
c-----------------------------------------------------------------------
c   chan      |   -->    |    channel number to read from
c   npoinmax  |   -->    |    Maximum number of nodes
c   nelmax    |   -->    |    Max no. of elements
c   ndpmax    |   -->    |    Max no. of nodes per element
c   nptfrmax  |   -->    |    Max no. of boundary nodes
c   nvarmax   |   -->    |    Max no. of variables
c   title     |  <--     |    text describing project
c   nvar      |  <--     |    Number of variables in file
c   text      |  <--     |    Text describing each variable
c   nelem     |  <--     |    Number of elements
c   npoin     |  <--     |    Number of nodes
c   ndp       |  <--     |    Number of nodes per element
c   ikle      |  <--     |    Mesh structure array
c   boundflag |  <--     |    Indicates whether a point is on boundary
c   nptfr     |  <--     |     Number of modes on boundary
c  x,y        |  <--     |    Node locations
c-------------------------------------------------------------------------


      implicit none

      integer chan
      integer nvar,ndp
      integer npoin,nelem,nptfr
      integer ikle(nelem*3),boundflag(npoin)
      integer ikles(nelem*3)

      double precision x(npoin),y(npoin)
      real   w(npoin)

      character*80 title
      character*32 text(2)

c Local variables

      integer i,nvar1,nvar2,ielem,junk(10)


c NB this subroutine assumes channel already opened to correct file
```

```
      ndp=3
      rewind (chan)
c read title
      read(chan) title(1:72)
c read number of variables
      read(chan) nvar1,nvar2
c nvar2 not used at present
      nvar = nvar1
c read text for each variable
      do 10 i = 1,nvar
        read(chan) text(i)(1:32)
10    continue
c read 10 unused integers
      read(chan) (junk(i),i=1,10)
c mesh info - 4 integers
      read(chan) nelem,npoin,ndp,junk(1)
c mesh structure
      read(chan) (ikles(i),i=1,nelem*ndp)
c no need to reorder [ikles(3,nelem) --> ikle(nelem,3)]
c     do 20 i = 1,ndp
c       do 25 ielem = 1,nelem
c         ikle(ielem+(i-1)*nelem) = ikles(i+(ielem-1)*ndp)
c25      continue
c20    continue

c read boundary flags
      read(chan) (boundflag(i),i=1,npoin)
c no need to work out number of boundary nodes
c     nptfr = 0
c     if (npoin .ge.1) then
c       do 30 i = 1,npoin
c         if(boundflag(i).ne.0) nptfr = nptfr + 1
c30      continue
c     endif

c read coordinates of nodes
      read(chan) (w(i),i=1,npoin)
c convert to double precision
c     do 40 i = 1,npoin
c       x(i) = dble(w(i))
c40    continue
      read(chan) (w(i),i=1,npoin)
c convert to double precision
c     do 50 i = 1,npoin
c       y(i) = dble(w(i))
c50    continue


      end

c **********************************************************
      subroutine readflow (chan,Dx,Dy,cequ,pkstress,npoin,nelem)


c Function: to read dispersion/cequ file

c    Name          |   i/o      |     Description
c------------------------------------------------------------------
c   chan           |    -->     |  Channel for disp file
c  Dx,Dy           |   <--      |  time-averaged dispersion coeffs
c  cequ            |   <--      |  time-averaged equilm. concentrations
c  work            |    --      |  real*4 workspace
```

```
c  npoinmax       |    -->     |  First dimension of arrays D,cequ
c  npoin          |    -->     |  number of nodes
c  time           |   <-->     |  time associated with data
c------------------------------------------------------------------------

      implicit none

      integer chan,npoin
      integer nelem

      double precision Dx(npoin),Dy(npoin),cequ(npoin),pkstress(npoin)

      real work(npoin)

c Local variables
      integer i
      real time
      logical eof

c--------------------------------------------------------------------------
--
c read data
      read(chan) time

c           read *,i

            read(chan) (work(i),i=1,npoin)
            do i = 1,npoin
              Dx(i) = dble(work(i))
            enddo
            read(chan) (work(i),i=1,npoin)
            do i = 1,npoin
              Dy(i) = dble(work(i))
            enddo
c           read *,i
            read(chan) (work(i),i=1,npoin)
            do i = 1,npoin
              cequ(i) = dble(work(i))
            enddo
           read(chan) (work(i),i=1,npoin)
            do i = 1,npoin
              pkstress(i) = dble(work(i))
            enddo
c           print *,'finished'

      return


      end
```

Appendix 4 Shell script

```
source v5p2_slow                                 choice of UNIX compiler
echo 'initialising starting conditions'
cp tollsmall.geom.nodrain tollsmall.geom         set initial bathymetry
cp tollsmall.geom.nodrain bathy.res
cp sed.xyz.ini sed.xyz                     set initial sediment conditions
cp geofile.ini geofile                     set initial underlying geology

jloops=41                                        define number of time steps

j=1

echo $j

until test $j -eq $jloops                  start of time loop
do

echo "running telemac2d"
telemac2d -s cas_tele                            run flow model

echo "Producing modified telemac file"
pickatime_morph < pick.str                 select water levels from time zero
                                           of flow file

echo "Processing flow/wave results"
dispcequ11 < dispcequ11.str                derive dispersion coefficients and
                                           equilibrium concentration

mv telemac.res telemac.res$j                     save flow model results

echo "running subief2d"
subief2d -s cas_sub6                             run mud transport model

rm sed.xyz

echo "writing geom/start file"

write_geom4 < write_geom4.str                           Change morphology
mv dispcequ.res dispcequ.res$j    save dispersion/equm.conc. results
cp geofile geofile$j                                    save geology file
mv geofile.new geofile

cp tollsmall.geom tollsmall.geom$j                      save bathymetry file

echo "re-writing bathymetry file"
write_bathy < write_bathy.str              save morphological result file

mv endgeom.xyz endgeom.xyz$j                     work file manipulation
mv newgeom.xyz newgeom.xyz$j                     work file manipulation
rm bathy.res
mv bathy_new.res bathy.res
mv subief.res subief.res$j                 save sediment transport file

j=`expr $j + 1`

echo $j

done
```

Appendix 5   dispcequ routine

```
      program dispcequ11

c             -------
c  reads in a Serafin results file (unstructured mesh) and calculates
c  dispersion coefficients and equilm. concentrations at each point
c------------------------------------------------------------------

      implicit none
      integer istat,nvar,k,ibid(2),i
      integer nelmax,npoinmax,nelem,npoin,maxvars,nstoremax
      integer nstore,nt,ipoin,ndp,nptfr
      parameter (nelmax=60000,npoinmax=60000,nstoremax=600,maxvars=8)


      real w(npoinmax)
      double precision xbid(2),times(nstoremax),time,bid
      double precision f(npoinmax,maxvars)
      double precision x(npoinmax),y(npoinmax)
      double precision hn(npoinmax),zs(npoinmax)
      double precision Dx(npoinmax),Dy(npoinmax)
      double precision cequ(npoinmax),pkstress(npoinmax)
      double precision tstart,tend
      integer ikle(nelmax*3),ikles(3,nelmax),i,ielem
      integer ib(10), ivar
      integer itrav(npoinmax)
      integer numtimes
      character*32 infile
      character*32 texte(maxvars),outfile
      character*32 textdispx,textdispy,textcequ,textcequ2
      character*32 textavspd
      character*72 titre
      character*80 titsel



c extra variables for realignment model
c waves
      integer iel,j,nclass,ndir
      integer ifabor(nelmax*3),iwork(npoinmax),iwork2(npoinmax)
      integer iwork3(npoinmax*30),idim3
      integer frequ(50,50),total_frequ

      double precision hs,tp,fetch,slope
      double precision wdir(50),U(50),p(50,50),wavedepthlim
      double precision windwavelim,avdepth
      logical swell,windwave,winputfile1,ifail,winput1,winput2
      character*32 windfile,windclimatefile
c currents
      double precision speed,z0,speedtotx(npoinmax),speedtoty(npoinmax)
      double precision avspeedx(npoinmax),avspeedy(npoinmax)
c stress
      double precision w1stress,w2stress(100,100),cstress,cstress2,Cd
      double precision threshold,total_stress(npoinmax)
      double precision threshold2,total_stress2(npoinmax)
c sediment transport
      double precision rateconstant,rateconstant2,ws
c character
      character*72 line
      character*10 yesno_windwave,yesno_winput1,yesno_winput2
      character*10 yesno_swell
c general
      double precision endtime,start_time
      logical eof
      external eof
c Read steering file

c read header line
9     read(5,999) line
999   format(a72)
      if(line(1:1).eq.'/') goto 9

c read results input file
10     read(5,1000) line
1000   format(a72)
       if(line(1:1).eq.'/') goto 10

      read(5,*) infile
      open(12,file=infile,form='UNFORMATTED')

      rewind 12

c read start/endtimes
11    read(5,1001) line
1001  format(a72)
      if(line(1:1).eq.'/') goto 11
      read *,start_time,endtime

c read friction coefficient
12    read(5,1002) line
1002  format(a72)
      if(line(1:1).eq.'/') goto 12
      read *,z0

c read whether windwaves selected
13    read(5,1003) line
      print *,line
1003  format(a72)
      if(line(1:1).eq.'/') goto 13

100   read(5,*) yesno_windwave
      print *,yesno_windwave
        if(yesno_windwave(1:3).eq.'YES'.or.
     &                  yesno_windwave(1:3).eq.'yes') then
               windwave=.TRUE.
        elseif(yesno_windwave(1:2).eq.'NO'.or.
     &                  yesno_windwave(1:2).eq.'no') then
               windwave=.FALSE.
        else
      print *,'****** ERROR: are wind waves required? Yes/No? *******'
            go to 100
        endif

c if there are windwaves then read next four lines (which should not
appear
c otherwise) which contain NWAVE representative wind directions and wind
speeds
      if(windwave) then
14      read(5,1004) line
      print *,line
1004  format(a72)
      if(line(1:1).eq.'/') goto 14
```

```
150    read(5,*) yesno_winput2
       print *,yesno_winput2
         if(yesno_winput2(1:3).eq.'YES'.or.
      &                  yesno_winput2(1:3).eq.'yes') then
                winput2=.TRUE.
         elseif(yesno_winput2(1:2).eq.'NO'.or.
      &                    yesno_winput2(1:2).eq.'no') then
                winput2=.FALSE.
         else
       print *,'** ERROR: wind waves to be read from file? Yes/No? **'
       print *,'error ',yesno_winput2,winput2
         go to 150
         endif
         if(.not.(winput2)) then
       print *,'winput ', winput2
15     read(5,1005) line
       print *,line
1005   format(a72)
       if(line(1:1).eq.'/') goto 15
           read(5,*) windclimatefile
           print *,windclimatefile

16     read(5,1006) line
       print *,line
1006   format(a72)
       if(line(1:1).eq.'/') goto 16
           read(5,*) nclass
           print *,nclass
           do i=1,nclass
             read(5,*) U(i)
           enddo

17     read(5,1007) line
       print *,line
1007   format(a72)
       if(line(1:1).eq.'/') goto 17
           read(5,*) ndir
           print *,ndir
           do j=1,ndir
             read(5,*) wdir(i)
           enddo

c read water level limit for implementing effect of wind waves
18       read(5,1008) line
         print *,line
1008   format(a72)
         if(line(1:1).eq.'/') goto 18
         read(5,*) windwavelim
         print *,windwavelim

         endif
c read water depth limit for implementing effect of all waves
19      read(5,1009) line
        print *,line
1009   format(a72)
       if(line(1:1).eq.'/') goto 19
       read(5,*) wavedepthlim
       print *,wavedepthlim

c end of if windwave loop
       endif

c read whether swell waves selected
```

```
20     read(5,1010) line
       print *,line
1010  format(a72)
      if(line(1:1).eq.'/') goto 20
200   read(5,*) yesno_swell
      print *,yesno_swell
        if(yesno_swell(1:3).eq.'YES'.or.
     &                    yesno_swell(1:3).eq.'yes') then
              swell=.TRUE.
        elseif(yesno_swell(1:2).eq.'NO'.or.
     &                    yesno_swell(1:2).eq.'no') then
              swell=.FALSE.
        else
      print *,'****** ERROR: are swell waves required? Yes/No? *******'
        go to 200
        endif

c read whether swell waves to be input from file
      if(swell) then
21    read(5,1011) line
1011  format(a72)
      if(line(1:1).eq.'/') goto 21

300   read(5,*) yesno_winput1
        if(yesno_winput1(1:3).eq.'YES'.or.
     &                 yesno_winput1(1:3).eq.'yes') then
              winput1=.TRUE.
        elseif(yesno_winput1(1:2).eq.'NO'.or.
     &                 yesno_winput1(1:2).eq.'no') then
              winput1=.FALSE.
        else
      print *,'** ERROR: are swell waves read from file? Yes/No? **'
        go to 300
        endif

      if(winput1) then
22      read(5,1012) line
1012    format(a72)
        if(line(1:1).eq.'/') goto 22

        read(5,*) winputfile1

      else
28      read(5,1018) line
1018    format(a72)
        if(line(1:1).eq.'/') goto 28
        read *,hs,tp
      endif

c endif for swell routine
      endif

c read erosion thresholds (fresh mud and eroding old mud)
23    read(5,1013) line
1013  format(a72)
      if(line(1:1).eq.'/') goto 23

      read(5,*) threshold,threshold2

c read erosion rate constants (fresh mud and eroding old mud)
24    read(5,1014) line
1014  format(a72)
      if(line(1:1).eq.'/') goto 24
      read(5,*) rateconstant,rateconstant2

c read settling velocity
25    read(5,1015) line
1015  format(a72)
      if(line(1:1).eq.'/') goto 25

      read(5,*) ws

c read outputfile
26    read(5,1016) line
1016  format(a72)
      if(line(1:1).eq.'/') goto 26
      read(5,1017) outfile
1017  format(a32)


      print *, 'Steering file read correctly'

c *********** read info from flow results file
******************************


c... title
      call lit(x,w,itrav,titre,72,'CH',12,'STD',istat)
      write(*,*) 'Title of study:'
      write(*,*) titre
c... number of variables

      call lit(xbid,w,ib,titre,2,'I ',12,'STD',istat)
      nvar = ib(1)
c     write(*,*) ib(1),ib(2)

c... text for each variable
      do k=1,nvar
        call lit(xbid,w,ibid,texte(k),32,'CH',12,'STD',istat)
      enddo

c... 10 integer parameters

      call lit(xbid,w,ib,titre,10,'I',12,'STD',istat)


c... nelem etc

      call lit(xbid,w,ib,titre,4,'I',12,'STD',istat)
      nelem = ib(1)
      npoin = ib(2)
      ndp   = ib(3)
c     write(*,*) nelem,' elements, ',npoin,' nodes  ',ndp

c... ikle

      call lit(xbid,w,ikles,titre,nelem*ndp,'I',12,'STD',istat)

c... convert ikles to ikle
      do 110 i = 1,ndp
       do 115 ielem = 1,nelem
          ikle(ielem+(i-1)*nelem) = ikles(i,ielem)
115    continue
110   continue
```

```
c reads array itrav ( = 0 if interior node, = node no. if on boundary)

      call lit(xbid,w,itrav,titre,npoin,'I',12,'STD',istat)

c     write(*,*) itrav(1), itrav(200), itrav(2430)
c calculate number of boundary nodes:

      nptfr = 0
      if (npoin .ge.1) then
        do 120 i = 1,npoin
          if(itrav(i).ne.0) nptfr = nptfr + 1
120     continue
      endif

c     write(*,*) nptfr, ' nodes on boundary'


c... node coordinates

      call lit(x,w,ib,titre,npoin,'R4',12,'STD',istat)
      call lit(y,w,ib,titre,npoin,'R4',12,'STD',istat)



      if (npoin.gt.npoinmax) then
        write(*,*) 'Too many nodes'
        stop
      elseif (nelem.gt.nelmax) then
        write(*,*) 'Too many elements'
        stop
      endif


c****************************************************************
c Print out variable descriptions

      write(*,*) 'The following variables have been read in:'
      do 610 i = 1,nvar
        write(*,*) i,'  ',texte(i)
610   continue

c     write(*,*) 'Enter earliest time to be considered'
c     read(*,*) tstart
c     write(*,*) 'Enter latest time to be considered'
c     read(*,*) tend

c----------------------------------------------------------------
c initialise
      numtimes = 0
      do 475, ipoin = 1,npoin
         Dx(ipoin) = 0.0
         Dy(ipoin) = 0.0
         cequ(ipoin) = 0.0
         pkstress(ipoin) = 0.0
         speedtotx(ipoin)=0.0
         speedtoty(ipoin)=0.0
         total_stress(ipoin)=0.0
         total_stress2(ipoin)=0.0

475   continue

      print *,'Initialising dispersion and equ. concentration'
      if(windwave.and.(.not.winput2)) then
c if using wind waves but not working out stresses offline
c then will need boundary node information to calculate fetches
c Calculate IFABOR - numbers of neighbouring elements
         idim3 = npoin*10
c telemac routine
         call voisin(ifabor,nelem,nelem,3,ikle,iwork,iwork2,iwork3,
     &                     idim3,npoin)
c also will need to read wind climate
         open(30,status='old',file=windclimatefile)
         rewind(30)
         total_frequ=0
         do i=1,nclass
           read(30,*) (frequ(i,j),j=1,ndir)
           do j=1,ndir
             total_frequ=total_frequ+frequ(i,j)
           enddo
         enddo

         do i=1,nclass
           do j=1,ndir
             p(i,j)=float(frequ(i,j))/float(total_frequ)
           enddo
         enddo

      else
            nclass=1
            ndir=1
            frequ(1,1)=1.0
            p(1,1)=1.0
      endif

c---------------------------------------------------------------
c   read times and results
       nstore = 0



500    if (eof(12)) goto 600

          call lit(time,w,itrav,titre,1,'R4',12,'STD',istat)
          if (time.gt.endtime) then
              goto 600
          endif
          nstore = nstore + 1
          do 510 i = 1,nvar
c           write(*,*) name
            call lit(f(1,i),w,itrav,titre,npoin,'R4',12,'STD',istat)
            if(i.eq.4) then
              do j=1,npoin
                zs(j)=f(j,i)
              enddo
            elseif(i.eq.3) then
             do j=1,npoin
                hn(j)=f(j,i)
              enddo
            endif
510       continue

c--------------------------------------------------------------
c work out stresses experienced through the tide
```

```
         if (time.ge.start_time.and.time.le.endtime) then
         write(*,*) 'Time ',time,' being processed'
         numtimes = numtimes+1

         do ipoin = 1,npoin

c calculate dispersion coeffs
            speed = dsqrt(f(ipoin,1)**2+f(ipoin,2)**2)

            speedtotx(ipoin)=speedtotx(ipoin)+abs(f(ipoin,1))
            speedtoty(ipoin)=speedtoty(ipoin)+abs(f(ipoin,2))

c calculate shear stress from currents
            if(hn(ipoin).gt.0.05) then
              Cd=(0.4/(log(hn(ipoin)/z0)-1))**2
              cstress=1025*speed**2*Cd
            else
              cstress=0.d0
            endif

c calculate excess shear stress from swell waves
            if(swell) then

              if(hn(ipoin).gt.wavedepthlim) then
                if(winput1) then
c read waves from file but unlike wind waves need to step through time
                   print *,'****** Model not developed - stop *****'
                   stop
c                  call getswellwave(ipoin,hn(ipoin),hs,tp)
                endif
                call wavestress(w1stress,hs,tp,hn(ipoin),z0,slope)
              else
                w1stress=0.d0
              endif
            else
              w1stress=0.d0
            endif

c calculate shear stress from wind waves
            if(windwave) then

c use wave stress distributions worked out offline as for Tollesbury
               if(winput2) then
                call getwindwavestress(x(ipoin),y(ipoin),
     &                              cstress+w1stress,threshold,w2stress)
            if(ipoin.eq.327) then
c           print *,'wave stress',w2stress,x(ipoin),y(ipoin)
            endif

c work out wind waves from fetch and Young et al's algorithm
              elseif(zs(ipoin).gt.windwavelim.and.
     &             hn(ipoin).gt.wavedepthlim) then
                 w2stress=0.0
                 do j=1,ndir
c wind direction, speed and duration should have been defined in the
c steering file for nwave representative winds
      call whichelem(iel,x(ipoin),y(ipoin),x,y,npoin,nelem,ikle,ifail)
c work out fetch
                 call getfetch(wdir(j),x(ipoin),y(ipoin),iel,x,y,ikle,
     &                        ifabor,nelem,npoin,ndp,fetch,zs,
     &                        windwavelim,avdepth,hn)
                 if(ipoin.eq.568.and.j.eq.13.and.int(time).eq.44400)
     &               print *,'fetch ',fetch,avdepth
```

```
c use young et al's algorithm to get wave height and period

                 do i=1,nclass
                 call getwindwave(U(i),fetch,avdepth,hs,tp)
c              if(ipoin.eq.568.and.j.eq.13.and.int(time).eq.44400)
c     & print *,'hs wstress hn',i,hs,hn(ipoin)
               hs=min(hs,0.5*hn(ipoin))
              call wavestress(w2stress(i,j),hs,tp,hn(ipoin),z0,-99.9)
c                if(ipoin.eq.568.and.j.eq.13.and.int(time).eq.44400)
c     & print *,'hs wstress hn',i,hs,w2stress(i,j),hn(568),tp,avdepth
                 enddo
                 enddo


               else
                 do i=1,nclass
                   do j=1,ndir
                      w2stress(i,j)=0.d0
                   enddo
                 enddo
               endif
            else
                 do i=1,nclass
                   do j=1,ndir
                      w2stress(i,j)=0.d0
                   enddo
                 enddo
            endif

        do i=1,nclass
          do j=1,ndir
            total_stress(ipoin)=total_stress(ipoin)+
     &  max(cstress+w1stress+w2stress(i,j)-threshold,0.d0)*p(i,j)
                if(ipoin.eq.474.and.i.eq.2.and.j.eq.11) print *,
     &           'w1stress @ 474 ',cstress,w1stress,w2stress(i,j)
            pkstress(ipoin)=max(max(pkstress(ipoin),
     &  (cstress+w1stress+w2stress(i,j))-threshold,0.d0)
c       if(ipoin.eq.474) print*,cstress,w1stress,w2stress(i,j),
c     &        p(i,j)
          enddo
        enddo

         enddo
         print *,'tau1,tau2',
     &    total_stress(474)/numtimes,
     &    total_stress2(474)/numtimes,total_stress(474),numtimes

         else
           write(*,*) 'Time ',time,' outside range'
         endif
c------------------------------------------------------------


       goto 500

600   continue

c at end of file : work out means
      if (numtimes.eq.0) then
        write(*,*) 'No stored results in required time range'
        stop
      endif
      print *, 'calculating dispersion coefficients and'
      print *, 'equilibrium values of concentration'
      do 615 ipoin = 1,npoin
        avspeedx(ipoin) = speedtotx(ipoin)/float(numtimes)
        avspeedy(ipoin) = speedtoty(ipoin)/float(numtimes)
        Dx(ipoin) = 200.0*(avspeedx(ipoin)/0.34)**2
     &                + 0.1*(avspeedy(ipoin)/0.34)**2
        Dy(ipoin) = 200.0*(avspeedy(ipoin)/0.34)**2
     &                + 0.1*(avspeedx(ipoin)/0.34)**2
        cequ(ipoin) = total_stress(ipoin)*rateconstant/ws
     &              /float(numtimes)
c       if(ipoin.eq.402) print *,'avspd',total_stress(ipoin),
c    &         rateconstant,ws
615   continue
c****************************************************************

      print *,'writing output file'


      textdispx = 'DISPERSION_X'
      textdispy = 'DISPERSION_Y'
      textcequ = 'CONC_EQU'
      textcequ2 = 'PKSTRESS'

c****************************************************************
c write information to output results file
      open(13,file=outfile, form='UNFORMATTED')

      titsel = titre // '        '
      call ecrit(x,itrav,titsel,80,'CH',13,'STD',istat)
c 2 variables in file: D, cequ
      ib(1) = 4
      ib(2) = 0
      call ecrit(xbid,ib,titre,2,'I',13,'STD',istat)
      call ecrit(xbid,ibid,textdispx,32,'CH',13,'STD',istat)
      call ecrit(xbid,ibid,textdispy,32,'CH',13,'STD',istat)
      call ecrit(xbid,ibid,textcequ,32,'CH',13,'STD',istat)
      call ecrit(xbid,ibid,textcequ2,32,'CH',13,'STD',istat)
      ib(1) = 1
      do 651 i=2,10
651     ib(i) = 0
      call ecrit(xbid,ib,titre,10,'I',13,'STD',istat)
      ib(1) = nelem
      ib(2) = npoin
      ib(3) = ndp
      ib(4) = 0
      call ecrit(xbid,ib,titre,4,'I',13,'STD',istat)
      call ecrit(xbid,ikles,titre,nelem*ndp,'I',13,'STD',istat)
      call ecrit(xbid,itrav,titre,npoin,'I',13,'STD',istat)

c...                 write node coordinates
      call ecrit(x,ib,titre,npoin,'R4',13,'STD',istat)
      call ecrit(y,ib,titre,npoin,'R4',13,'STD',istat)


      time = 0.d0
      call ecrit(time,itrav,titre,1,'R4',13,'STD',istat)
      call ecrit(Dx,itrav,titre,npoin,'R4',13,'STD',istat)
      call ecrit(Dy,itrav,titre,npoin,'R4',13,'STD',istat)
      call ecrit(cequ,itrav,titre,npoin,'R4',13,'STD',istat)
      call ecrit(pkstress,itrav,titre,npoin,'R4',13,'STD',istat)
      stop

      close(12)
      close(13)

      end

c *******************************************************************

      subroutine getwindwave(U,F,hn,hs,tp)

      double precision U,F,hn,hs,tp,delta
      double precision A1,B1,A2,B2,X,g
      double precision nondimE,nondimV,E

      g=9.81

      delta=g*hn/U**2
      X=g*F/U**2
      A1=0.493*delta**0.75
      B1=0.00313*X**0.57
      A2=0.331*delta**1.01
      B2=0.0005215*X**0.73
      nondimE=0.00364*(TANH(A1)*TANH(B1/TANH(A1)))**1.74
      nondimV=0.133*(TANH(A2)*TANH(B2/TANH(A2)))**(-0.37)
      E=nondimE*(U**4)/g**2
      hs=4*sqrt(E)
      tp=U/nondimV/g
c     if(hs.gt.0.1) then
c       print *,'hn,U,F,hs,tp'
c       print *,hn,U,F,hs,tp
c     endif

      return

      end

c ******************************************************************


      subroutine getfetch(wdir,xold,yold,iel,x,y,ikle,
     &                    ifabor,nelem,npoin,ndp,dx,zs,
     &                    windwavelim,avdepth,hn)


c Function:    This subroutine starts from the position of a node and
c              establishes the distance to the boundary.
c              It also crudely calculates the average fetch water depth
c              NB modified from SEDPLUME-RW "trackpart" routine


c Bill Roberts    Jan 1993

c  Variable  |   Mode   |  Description
c-----------------------------------------------------------------------
c  xnew      |  <--->   |  x-coord of new particle position
c  ynew      |  <--->   |  y-coord    "    "
c  xold      |   --->   |  x-coord of old particle position
c  yold      |   --->   |  y-coord  ...
c  iel       |  <--->   |  element containing the particle
c  x         |   --->   |  x-coord of each node
c  y         |   --->   |  y-coord of each node
c  ikle      |   --->   |  mesh structure array
```

```
c  ifabor    |   --->   |  array of elements neighbouring each element
face
c  nelem     |   --->   |  number of elements
c  npoin     |   --->   |  number of nodes
c  ndp       |   --->   |  number of nodes per element
c------------------------------------------------------------------------
      implicit none

      integer nelem,ndp,npoin
      integer iel, ikle(nelem,ndp),ifabor(nelem,ndp)
      integer newel,side,oldel
      integer i1,i2,i3,i,numtimes

      double precision xnew,ynew,xold,yold,x(npoin),y(npoin)
      double precision det1,det2,det3
      double precision alpha1,alpha2,alpha3
      double precision x1,x2,x3,y1,y2,y3
      double precision eps
      double precision length,dx
      double precision wdir,dx,zs(npoin),windwavelim
      double precision avdepth,totdepth,hn(npoin)

      parameter (eps=-1.0d-6)
      numtimes=0
      totdepth=0.d0


c Save original element
      oldel = iel

c find position of xnew,ynew outside of model
      xnew=xold+dsin((wdir+180.d0)/360.d0*2*3.142)*1000000.0
      ynew=yold+dcos((wdir+180.d0)/360.d0*2*3.142)*1000000.0

10    i1 = ikle(iel,1)
      i2 = ikle(iel,2)
      i3 = ikle(iel,3)

      x1 = x(i1)
      x2 = x(i2)
      x3 = x(i3)
      y1 = y(i1)
      y2 = y(i2)
      y3 = y(i3)
      det1 = (x3-x2)*(ynew-y2) - (y3-y2)*(xnew-x2)
      det2 = (x1-x3)*(ynew-y3) - (y1-y3)*(xnew-x3)
      det3 = (x2-x1)*(ynew-y1) - (y2-y1)*(xnew-x1)
      if ((dabs(xnew-xold).lt.1.d-3).and.
     &    (dabs(ynew-yold).lt.1.d-3)) then
         alpha1 = 0.d0
         alpha2 = 0.d0
         alpha3 = 0.d0
      else
         alpha1 = ((ynew-yold)*(x1-xold) - (xnew-xold)*(y1-yold))/
     &              ((xnew-xold)*(y2-y1) - (ynew-yold)*(x2-x1))
         alpha2 = ((ynew-yold)*(x2-xold) - (xnew-xold)*(y2-yold))/
     &              ((xnew-xold)*(y3-y2) - (ynew-yold)*(x3-x2))
         alpha3 = ((ynew-yold)*(x3-xold) - (xnew-xold)*(y3-yold))/
     &              ((xnew-xold)*(y1-y3) - (ynew-yold)*(x1-x3))
      endif
c                    note - side1 uses det3: node1 -> node2
c                           side2 uses det1: node2 -> node3
c                           side3 uses det2: node3 -> node1
```

```
      if ((det3.lt.0.0d0).and.(alpha1.ge.0.0d0).and.
     &                                    (alpha1.le.1.0d0)) then
         side = 1
         length = dsqrt((x2-x1)**2+(y2-y1)**2)
      elseif ((det1.lt.0.0d0).and.(alpha2.ge.0.0d0).and.
     &                                   (alpha2.le.1.0d0)) then
         side = 2
         length = dsqrt((x3-x2)**2+(y3-y2)**2)
      elseif ((det2.lt.0.0d0).and.(alpha3.ge.0.0d0).and.
     &                                    (alpha3.le.1.0d0)) then
         side = 3
         length = dsqrt((x3-x1)**2+(y3-y1)**2)
      else
c...                                                       In this
element
         side = 0
      endif

      if (side.ne.0) then
        newel = ifabor(iel,side)
        if (newel.eq.-1.or.newel.eq.0) then
c...                                  Solid or open boundary
          if(side.eq.1) then
               dx=0.5*((x1-xold)**2+(y1-yold)**2)**0.5+
     &                       0.5*((x2-xold)**2+(y2-yold)**2)**0.5
          elseif(side.eq.2) then
               dx=0.5*((x3-xold)**2+(y3-yold)**2)**0.5+
     &                       0.5*((x2-xold)**2+(y2-yold)**2)**0.5
          elseif(side.eq.3) then
               dx=0.5*((x3-xold)**2+(y3-yold)**2)**0.5+
     &                       0.5*((x1-xold)**2+(y1-yold)**2)**0.5
          endif

        else
c...                                   New element - check this one out
          iel = newel
          numtimes=numtimes+1
          if(side.eq.1) then
             totdepth=totdepth+0.5*(hn(i1)+hn(i2))
             dx=0.5*((x1-xold)**2+(y1-yold)**2)**0.5+
     &                        0.5*((x2-xold)**2+(y2-yold)**2)**0.5
             if(hn(i1).gt.0.05.and.hn(i2).gt.0.05) then
               goto 10
             endif
          elseif(side.eq.2) then
             totdepth=totdepth+0.5*(hn(i3)+hn(i2))
             dx=0.5*((x3-xold)**2+(y3-yold)**2)**0.5+
     &                        0.5*((x2-xold)**2+(y2-yold)**2)**0.5
             if(hn(i3).gt.0.05.and.hn(i2).gt.0.05) then
               goto 10
             endif
          elseif(side.eq.3) then
             totdepth=totdepth+0.5*(hn(i3)+hn(i1))
             dx=0.5*((x3-xold)**2+(y3-yold)**2)**0.5+
     &                         0.5*((x1-xold)**2+(y1-yold)**2)**0.5
             if(hn(i3).gt.0.05.and.hn(i1).gt.0.05) then
               goto 10
             endif
          endif

        endif
      else
```

```
          print *,'****** LOST IN TRACKPART ***********'
          stop
        endif

        avdepth=totdepth/float(numtimes)

        return

        end

c***********************************************************************
****

        subroutine whichelem(elem,x0,y0,x,y,npoin,nelem,ikle,lfail)


c Function:  finds the element containing the point (x0,y0)
c            method is to test each element in turn, checking that the
point
c            lies on the correct side of each face in turn.


c Bill Roberts     Jan 1993



        implicit none

        integer elem,npoin,nelem, ikle(nelem,3), i
        integer i1,i2,i3
        double precision x0,y0,x1,y1,x2,y2,x3,y3
        double precision det1,det2, det3
        double precision x(npoin), y(npoin)
        logical lfail


        lfail = .false.
        do 100 i = 1, nelem

          i1 = ikle(i,1)
          i2 = ikle(i,2)
          i3 = ikle(i,3)

          x1 = x(i1)
          x2 = x(i2)
          y1 = y(i1)
          y2 = y(i2)
          x3 = x(i3)
          y3 = y(i3)

          det1 = (x2-x1)*(y0-y1) - (y2-y1)*(x0-x1)
          det2 = (x3-x2)*(y0-y2) - (y3-y2)*(x0-x2)
          det3 = (x1-x3)*(y0-y3) - (y1-y3)*(x0-x3)

          if(det1.ge.0.d0  .and. det2.ge.0.d0 .and. det3.ge.0.d0)then
            elem = i
            return
          endif

100    continue


       lfail = .true.
```

```
        elem = -1
        return

        end

c*********************************************************************

       subroutine getwindwavestress(x,y,stress,threshold,w2stress)

       integer i
       double precision x,y
       double precision w2stress,total,totalweight,weight,dissq
       double precision p(20,20),xwv(20),ywv(20),tauw(20)

       data (xwv(i),i=1,6)/ 595859,596092,596342,595838,596065,596155/
       data (ywv(i),i=1,6)/ 211511,211592,211439,211382,211438,211263/
       data (tauw(i),i=1,6)/ 0.002,0.008,0.009,0.000,0.001,0.005/
c      data (p(1,i),i=1,10)/2.08,1.14,0.43,0.24,0.16,0.17,0.02,0.03,
c     &          0.01,0.02,0.00
c      data (p(2,i),i=1,10)/5.33,2.47,0.95,0.71,0.55,0.42,0.35,0.26,
c     &      0.15,0.16,0.02
c      data (p(3,i),i=1,10)/10.49,3.16,1.52,0.99,0.76,0.32,0.35,0.19,
c     &      0.10,0.13,0.05
c      data (p(4,i),i=1,10)/0.78,0.53,0.38,0.15,0.05,0.10,0.13,0.02,
c           0.01,0.01,0.00
c      data (p(5,i),i=1,10)/10.39,2.59,1.16,0.49,0.43,0.22,0.06,0.07,
c     &             0.07,0.01,0.00
c      data (p(6,i),i=1,10)/2.61,1.58,0.83,0.42,0.33,0.15,0.15,0.13,
c     &          0.06,0.06,0.03

       total=0.d0
       totalweight=0.d0
       do i=1,6
          dissq=((xwv(i)-x)**2+(ywv(i)-y)**2)
          weight=1.d0/dissq
          totalweight=totalweight+weight
          total=total+tauw(i)*weight
       enddo
c      print *,weight,totalweight,total,w2stress
       w2stress=total/totalweight

       return

       end

c ****************************************************************

       subroutine wavestress(wstress,hs,tp,hn,z0,slope)

       double precision tn,hn,t1,tp,A1,A2,Urms,Uw,fw
       double precision wstress,limHs,slope,z0,hs

       if(hn.lt.0.05) then

         wstress=0.d0
         return

       else

c revise wave depth according to Le Hir et al
c        if(slope.gt.-99.and.slope/fw.lt.1) then
c          limHs=(15.d0*3.142/4.d0)*slope/fw*hn
c          hs=min(hs,limHs)
```

```
c       endif

        tn=max((hn/9.81)**0.5,0.000001)
        t1=tn/tp
        A1=(6500.0+(0.56+15.54*t1)**6)**0.16666667

        Urms=max(0.25/(1+A1*t1**2)**3*hs/tn,0.0001)
        Uw=1.41*Urms
        A2=Uw*Tp/2/3.142

      endif

c smooth turbulent law
c     fw=max(0.0521*(Uw*A2/(0.00000136))**(-0.187),0.0001)

c laminar law
      fw=max(2.d0*(Uw*A2/(0.00000136))**(-0.5),0.0001)

      wstress=max(0.5*1025.0*fw*Uw**2,0.00001)

      if(hs.gt.0.1) then
c     print *,'hs,tp,hn,tn,t1,A1,Uw,fw,wstress'
c     print *,hs,tp,hn,tn,t1,A1,Uw,fw,wstress
      endif

      return

      end
```

Appendix 6 Young and Verhagen's method for wave evolution in shallow water

Young and Verhagen (1996) developed a simple model of fetch-limited wind waves, parameterising the wave height, total energy and peak frequency of waves as a function of fetch length, wind speed and average depth of the water along the fetch.

Initially the following non-dimensionalised parameters are defined:

non-dimensional energy	$\varepsilon = \dfrac{g^2 E}{U_{10}^4}$
non-dimensional frequency	$\nu = \dfrac{f_p U_{10}}{g}$
non-dimensional depth	$\delta = \dfrac{gd}{U_{10}^2}$
non-dimensional fetch	$\chi = \dfrac{gx}{U_{10}^2}$

where E is the total wave energy, f_p is the frequency of the spectral peak, g is the gravitational acceleration, U_{10} is the wind speed at a reference height of 10m, x is the fetch length, and d is the average water depth along the fetch.

The non-dimensional parameters are related as follows,

$$\varepsilon = 3.64x10^{-3}\left\{\tanh A_1 \tanh\left[\frac{B_1}{\tanh A_1}\right]\right\}^{1.74}$$

where $A_1 = 0.493\ \delta^{0.75}$ *and* $B_1 = 3.13 \text{ x } 10^{-3}\ \delta^{0.75}$, and ,

$$\nu = 0.133\left\{\tanh A_2 \tanh\left[\frac{B_2}{\tanh A_2}\right]\right\}^{-0.37}$$

where $A_2 = 0.331\ \delta^{1.01}$ *and* $B_2 = 5.215 \text{ x } 10^{-4}\ \chi^{0.73}$

Young I R and Verhagen L A (1996) The growth of fetch limited waves in water of finite depth. Part 1. Total energy and peak frequency, Coastal Engineering, volume 29, pp47-78.

Appendix 7 write_geom routine

```
      program write_geom


c             -------
c  reads in a Selafin results file (3-node triangular elements)
c  and outputs a single time level
c             -------

      implicit none
      integer nptfr,ndp,istat,nvar,k,ibid(2)
      integer nelmax,npoinmax,nelem,npoin,selafin,maxvars
      integer nstore,nstoremax,nt
      parameter (nelmax=60000,npoinmax=35000,nstoremax=90,maxvars=25)
      parameter(selafin=3)


      real w(npoinmax)
      integer downslope(npoinmax)
      double precision xbid(2),times(nstoremax),time,xtime
      double precision f(npoinmax,maxvars)
      double precision x(npoinmax),y(npoinmax)
      double precision u(npoinmax),v(npoinmax),h(npoinmax)
      double precision zs(npoinmax),zf(npoinmax)
      double precision cequ(npoinmax),pkstress(npoinmax)
      double precision Dx(npoinmax),Dy(npoinmax)
      double precision cs(npoinmax),esomt(npoinmax),dz,dz_onetide
      double precision dz_onetide,stab_slope
      double precision zr_weath(npoinmax),zr_clay(npoinmax),zr
      double precision Me_weath,duration,slope,dep_slope
      double precision threshold_weath,dens_weath,distance
      integer ikle(nelmax*3),ikles(3,nelmax),i,ielem
      integer ib(10), ivar
      integer itrav(npoinmax)
      integer numtimes,morph_number,chan
      character*32 infile,dispfile
      character*32 texte(maxvars),textmean,textmax,textmin,outfile
      character*16 descmean, descmax, descmin
      character*72 titre,line
      character*80 titsel
      logical eof,found
      external eof


8     read(5,998) line
998   format(a72)
      if(line(1:1).eq.'/') goto 8

9     read(5,999) line
999   format(a72)
      if(line(1:1).eq.'/') goto 9

      read(*,1001) infile
      open(12,file=infile,form='UNFORMATTED')

10     read(5,1002) line
1000   format(a72)
      if(line(1:1).eq.'/') goto 10
      read(5,1001) outfile
1001  format(a32)
```

```
12      read(5,1002) line
1002    format(a72)
        if(line(1:1).eq.'/') goto 12
        read(*,*) morph_number

13      read(5,1003) line
1003    format(a72)
        if(line(1:1).eq.'/') goto 13
        read(*,*) threshold_weath

14      read(5,1004) line
1004    format(a72)
        if(line(1:1).eq.'/') goto 14
        read(*,*) Me_weath

15      read(5,1005) line
1005    format(a72)
        if(line(1:1).eq.'/') goto 15
        read(*,*) dens_weath

16      read(5,1006) line
1006    format(a72)
        if(line(1:1).eq.'/') goto 16
        read(*,*) duration

      rewind 12

c *********** read info from file ******************************

c... title

      call lit(x,w,itrav,titre,72,'CH',12,'STD',istat)
      write(*,*) 'Title of study:'
      write(*,*) titre
c... number of variables

      call lit(xbid,w,ib,titre,2,'I ',12,'STD',istat)
      nvar = ib(1)
c     write(*,*) ib(1),ib(2)

c... text for each variable
      do 100,k=1,nvar
        call lit(xbid,w,ibid,texte(k),32,'CH',12,'STD',istat)
100   continue

c... 10 integer parameters

      call lit(xbid,w,ib,titre,10,'I',12,'STD',istat)


c... nelem etc

      call lit(xbid,w,ib,titre,4,'I',12,'STD',istat)
      nelem = ib(1)
      npoin = ib(2)
      ndp   = ib(3)
      write(*,*) nelem,' elements, ',npoin,' nodes  ',ndp

c... ikle

      call lit(xbid,w,ikles,titre,nelem*ndp,'I',12,'STD',istat)
```

```
c... convert ikles to ikle
      do 110 i = 1,ndp
       do 115 ielem = 1,nelem
          ikle(ielem+(i-1)*nelem) = ikles(i,ielem)
115    continue
110   continue

      write(*,*) ikle(1),ikle(1+nelem),ikle(1+2*nelem)

c... ipobo
c reads array itrav ( = 0 if interior node, = node no. if on boundary)

      call lit(xbid,w,itrav,titre,npoin,'I',12,'STD',istat)

      write(*,*) itrav(1), itrav(200), itrav(2430)
c calculate number of boundary nodes:

      nptfr = 0
      if (npoin .ge.1) then
        do 120 i = 1,npoin
          if(itrav(i).ne.0) nptfr = nptfr + 1
120     continue
      endif

      write(*,*) nptfr, ' nodes on boundary'


c... node coordinates

      call lit(x,w,ib,titre,npoin,'R4',12,'STD',istat)
      call lit(y,w,ib,titre,npoin,'R4',12,'STD',istat)

      write(*,*) x(370),y(370)

c****************************************************************
c Print out variable descriptions

      write(*,*) 'The following variables have been read in:'
      do 610 i = 1,nvar
        write(*,*) i,'  ',texte(i)
610   continue


c****************************************************************
c write header information to output results file
      open(13,file=outfile, form='UNFORMATTED')

c current test example, output file contains max of h only
      titsel = titre // '        '
      call ecrit(x,itrav,titsel,80,'CH',13,'STD',istat)
      ib(1) = 1
      ib(2) = 0
      call ecrit(xbid,ib,titre,2,'I',13,'STD',istat)
      texte(1)='BOTTOM          M               '

      call ecrit(xbid,ibid,texte(1),32,'CH',13,'STD',istat)

      ib(1) = 1
      do 651 i=2,10
651     ib(i) = 0
      call ecrit(xbid,ib,titre,10,'I',13,'STD',istat)
      ib(1) = nelem
      ib(2) = npoin
      ib(3) = ndp
      ib(4) = 0
      call ecrit(xbid,ib,titre,4,'I',13,'STD',istat)
      call ecrit(xbid,ikles,titre,nelem*ndp,'I',13,'STD',istat)
      call ecrit(xbid,itrav,titre,npoin,'I',13,'STD',istat)
      call ecrit(x,ib,titre,npoin,'R4',13,'STD',istat)
      call ecrit(y,ib,titre,npoin,'R4',13,'STD',istat)
c--------------------------------------------------------------
c   read times and results

       numtimes=0
500    if (eof(12)) goto 600
          call lit(time,w,itrav,titre,1,'R4',12,'STD',istat)
          write(*,*) 'Reading time ',time
          numtimes=numtimes+1
            call lit(u,w,itrav,titre,npoin,'R4',12,'STD',istat)
            call lit(v,w,itrav,titre,npoin,'R4',12,'STD',istat)
            call lit(h,w,itrav,titre,npoin,'R4',12,'STD',istat)
            call lit(zs,w,itrav,titre,npoin,'R4',12,'STD',istat)
            call lit(esomt,w,itrav,titre,npoin,'R4',12,'STD',istat)
            call lit(cs,w,itrav,titre,npoin,'R4',12,'STD',istat)

       if(numtimes.eq.1) then
         do i=1,npoin
           zf(i)=zs(i)-h(i)
         enddo
       endif

       write(*,*) time
       goto 500

600   continue

      print *,'read subief file'

c------------------------------------------------------------------
c       read dispersion file

      dispfile='./dispcequ.res'
      inquire(file=dispfile,exist=found)
      read *,i
      if (.not.found) then
111     write(*,*) 'FATAL ERROR: no dispersion file'
       goto 111
      endif

      open(23,file=dispfile,form='UNFORMATTED')

        call read_tel_hdr (23,npoin,nelem,3)

        call readflow(23,Dx,Dy,cequ,pkstress,w,npoin,nelem)

      print *, 'read dispersion file'

c--------------------------------------------------------------------

c write output


      call ecrit(0.0,itrav,titre,1,'R4',13,'STD',istat)

      open(20,status='old',file='endgeom.xyz')
      open(21,status='new',file='newgeom.xyz')
      open(22,status='new',file='sed.xyz')
      open(30,status='old',file='geofile')
      open(31,status='new',file='geofile.new')

c identify all the downslope nodes
      call findslope(npoin,nelem,zf,x,y,ikle,downslope)

      write(21,*) morph_number
      do i=1,npoin
        read(20,*) dz_onetide,zr

c read zr_weath,zr_clay in geo file
        read(30,*) zr_weath(i),zr_clay(i)

c NB need to read in Me_weath,dens_weath,threshold_weath and duration of
c exposure from steering file
        if(dz_onetide.gt.0.or.zf(i).gt.zr_weath(i)) then

c extrapolate modelled bed level change over a large number of tides
          dz=morph_number*dz_onetide

c disallow deposition on (hard-coded) slope
          slope=zf(i)-zf(downslope(i))
c hard code threshold slope for deposition arbitrarily
          dep_slope=stab_slope*0.5
          if(slope.lt.dep_slope) then

c limit rise in bed level to allowed slope
c           dz=min((slope-dep_slope)*distance,dz)

c deposition/erosion of mud
            zf(i)=zf(i)+dz

          endif

c only erode down to weathered layer
          zf(i)=max(zf(i),zr_weath(i))

c can't deposit above HW
          if(zf(i).gt.zs(i)) zf(i)=zs(i)

        endif

c account for erosion of weathered bed (clay bed assumed to be unerodable)
        if(zf(i).le.zr_weath(i).and.zf(i).gt.zr_clay(i)) then
          zf(i)=max(zf(i)-Me_weath*morph_number*duration
     &    *(max(pkstress(i)-threshold_weath,0.d0))
     &    /dens_weath,zr_clay(i))
           zr_weath(i)=zf(i)
        endif

        write(21,*) zf(i)
        cs(i)=max(cs(i),0.d0)
        write(22,*) cs(i),max(zf(i)-zr,0.d0)
      enddo

      call geo(npoin,pkstress,downslope,threshold_weath,zf,
     &               x,y,stab_slope,zr_clay)

      do i=1,npoin
         write(31,*) zr_weath(i),zr_clay(i)
      enddo
      call ecrit(zf,itrav,titre,npoin,'R4',13,'STD',istat)
      print *,'written geom file'
      print *,'written geofile'

      stop

      close(12)
      close(13)
      close(20)
      close(21)
      close(22)
      close(23)
      close(30)
      close(31)

      end

c-------------------------------------------------------------
      subroutine read_tel_hdr (chan,npoin,nelem,ndp)


c Function:  reads the header part of a Seraphin format TELEMAC results
c            file.


c-----------------------------------------------------------------------
c    Name     |   i/o    |    Description
c-----------------------------------------------------------------------
c   chan      |   -->    |    channel number to read from
c   npoinmax  |   -->    |    Maximum number of nodes
c   nelmax    |   -->    |    Max no. of elements
c   ndpmax    |   -->    |    Max no. of nodes per element
c   nptfrmax  |   -->    |    Max no. of boundary nodes
c   nvarmax   |   -->    |    Max no. of variables
c   title     |  <--     |    text describing project
c   nvar      |  <--     |    Number of variables in file
c   text      |  <--     |    Text describing each variable
c   nelem     |  <--     |    Number of elements
c   npoin     |  <--     |    Number of nodes
c   ndp       |  <--     |    Number of nodes per element
c   ikle      |  <--     |    Mesh structure array
c   boundflag |  <--     |    Indicates whether a point is on boundary
c   nptfr     |  <--     |     Number of modes on boundary
c  x,y        |  <--     |    Node locations
c-------------------------------------------------------------------------


      implicit none

      integer chan
      integer nvar,nvarmax,ndp2
      integer npoin,nelem,ndp,nptfr
      integer npoin2,nelem2
      integer ikle(nelem*ndp),boundflag(npoin)
      integer ikles(nelem*ndp)

      double precision x(npoin),y(npoin)
      real*4 w(npoin)

      character*80 title
      character*32 text(2)

c Local variables
```

```
      integer i,nvar1,nvar2,ielem,junk(10)


c NB this subroutine assumes channel already opened to correct file


      rewind (chan)
c read title
      read(chan,end=100,err=999) title(1:72)
c read number of variables
      read(chan,end=100,err=999) nvar1,nvar2
c nvar2 not used at present
      nvar = nvar1
c read text for each variable
      do 10 i = 1,nvar
        read(chan,end=100,err=999) text(i)(1:32)
10    continue
c read 10 unused integers
      read(chan,end=100,err=999) (junk(i),i=1,10)
c mesh info - 4 integers
      read(chan,end=100,err=999) nelem2,npoin2,ndp2,junk(1)
      if(nelem2.ne.nelem.or.
     &           npoin2.ne.npoin.or.ndp2.ne.ndp) then
         print *,'*** ERROR - different mesh ***'
         stop
      endif

c mesh structure
      read(chan,end=100,err=999) (ikles(i),i=1,nelem*ndp)
c no need to reorder [ikles(3,nelem) --> ikle(nelem,3)]
c     do 20 i = 1,ndp
c       do 25 ielem = 1,nelem
c         ikle(ielem+(i-1)*nelem) = ikles(i+(ielem-1)*ndp)
c25      continue
c20    continue

c read boundary flags
      read(chan,end=100,err=999) (boundflag(i),i=1,npoin)
c no need to work out number of boundary nodes
c     nptfr = 0
c     if (npoin .ge.1) then
c       do 30 i = 1,npoin
c         if(boundflag(i).ne.0) nptfr = nptfr + 1
c30      continue
c     endif

c read coordinates of nodes
      read(chan,end=100,err=999) (w(i),i=1,npoin)
c convert to double precision
c     do 40 i = 1,npoin
c       x(i) = dble(w(i))
c40    continue
      read(chan,end=100,err=999) (w(i),i=1,npoin)
c convert to double precision
c     do 50 i = 1,npoin
c       y(i) = dble(w(i))
c50    continue

      return

100   print *,'*** ERROR - End of file encountered ***'
      stop
      return

999   print *,'*** ERROR - Error reading file ***'
      stop

      return

      end

c-------------------------------------------------------
      subroutine readflow (chan,Dx,Dy,cequ,pkstress,work,
     &                    npoin,nelem)


c Function: to read dispersion/cequ file

c    Name         |   i/o      |     Description
c------------------------------------------------------------------------
c   chan          |    -->     |  Channel for disp file
c  D              |   <--      |  time-averaged dispersion coeffs
c  cequ           |   <--      |  time-averaged equilm. concentrations
c  work           |    --      |  real*4 workspace
c  npoinmax       |    -->     |  First dimension of arrays D,cequ
c  npoin          |    -->     |  number of nodes
c  time           |   <-->     |  time associated with data
c------------------------------------------------------------------------

      implicit none

      integer chan,npoin
      integer nelem

      double precision Dx(npoin),Dy(npoin)
      double precision cequ(npoin),pkstress(npoin)

      real work(npoin)

c Local variables
      integer i
      real time
      logical eof

c--------------------------------------------------------------------------
--
c read data
      read(chan,end=998,err=999) time

            read(chan,end=998,err=999) (work(i),i=1,npoin)
            do 20 i = 1,npoin
              Dx(i) = dble(work(i))
20          continue
            read(chan,end=998,err=999) (work(i),i=1,npoin)
            do 30 i = 1,npoin
              Dy(i) = dble(work(i))
30          continue
            read(chan,end=998,err=999) (work(i),i=1,npoin)
            do 40 i = 1,npoin
              cequ(i) = dble(work(i))
40          continue
            read(chan,end=998,err=999) (work(i),i=1,npoin)
            do 50 i = 1,npoin
              pkstress(i) = dble(work(i))
```

```
50          continue

      return

998   print *,'*** ERROR - End of file encountered ***'
      stop

      return

999   print *,'*** ERROR - Error reading file ***'
      stop

      return



      end


c-----------------------------------------------------
      subroutine findslope(npoin,nelem,zf,x,y,ikle,downslope)


      integer npoin,nelem,i,j,ielem
      integer downslope(npoin)
      integer ikle(3*nelem)
      integer ikle2(nelem,3),ielem
      double precision temp_slope,distance
      double precision x(npoin),y(npoin),zf(npoin)
      double precision slope_elem(nelem,3)
      double precision slope_node(npoin)

c work out which nodes are immediately downslope
c slope at each node of element are stored in slope_elem
c steepest slope at each node are stored in slope_node
c

c... convert ikles to ikle
      do 110 i = 1,3
       do 115 ielem = 1,nelem
          ikle2(ielem,i)=ikle(ielem+(i-1)*nelem)
115    continue
110   continue


      do i=1,npoin
         downslope(i)=0
         slope_node(i)=-999.9
      enddo

      do i=1,nelem

       do j=1,3

       slope_elem(i,j)=zf(ikle2(i,mod(j-1,3)+1))-zf(ikle2(i,mod(j,3)+1))
       distance=((x(ikle2(i,mod(j-1,3)+1))-x(ikle2(i,mod(j-1,3)+1))**2)+
     &      (y(ikle2(i,mod(j-1,3)+1))-y(ikle2(i,mod(j-1,3)+1)))**2)**0.5
        slope_elem(i,j)=slope_elem(i,j)/distance
        temp_slope=
     &    max(slope_elem(i,mod(j-1,3)+1),-slope_elem(i,mod(j+1,3)+1))
        if(temp_slope.gt.slope_node(ikle2(i,mod(j-1,3)+1))) then
            slope_node(ikle2(i,mod(j-1,3)+1))=temp_slope
            downslope(ikle2(i,mod(j-1,3)+1))=ikle2(i,mod(j,3)+1)
        else
            downslope(ikle2(i,mod(j-1,3)+1))=ikle2(i,mod(j+1,3)+1)
        endif

       enddo

      enddo

      end

c-----------------------------------------------------------------------
      subroutine geo(npoin,pkstress,downslope,threshold_weath,zf,
     &               x,y,stab_slope,zr_clay)

      integer npoin,i,j
      double precision zf(npoin),zr_clay(npoin)
      double precision threshold_weath
      double precision x(npoin),y(npoin)
      double precision stab_slope,distance,slope2
      double precision pkstress(npoin)
      integer downslope(npoin)


c implement collapse iteratively
c (note that method chosen deals with erosion only
c  and deosn't put slumped sediemnt at bottom of slope)
c slumping only occurs if downslope node is eroded

      do i=1,20
         do j=1,npoin
           if(pkstress(downslope(j)).gt.threshold_weath) then
             slope2=zf(j)-zf(downslope(j))
             distance=((x(j)-x(downslope(j)))**2+
     &                    (y(j)-y(downslope(j)))**2)**0.5
             slope2=slope2/distance
             if(slope2.gt.stab_slope) then
                zf(j)=zf(j)-(slope2-stab_slope)*distance/2
                zf(j)=max(zf(j),zr_clay(j))
             endif
           endif
         enddo
      enddo

      return


      end
```

Appendix 8 write_bathy routine

```
      program write_bathy

      implicit none
      integer nptfr,ndp,istat,nvar,k,ibid(2)
      integer nelmax,npoinmax,nelem,npoin,selafin,maxvars
      integer nstore,nstoremax,nt
      parameter (nelmax=160000,npoinmax=85000,nstoremax=75,maxvars=7)
      parameter(selafin=3)


      real w(npoinmax)
      double precision xbid(2),time,xtime,firsttime
      double precision f(npoinmax,maxvars)
      double precision x(npoinmax),y(npoinmax)
      double precision bottom(npoinmax)
      double precision time_new,morphtimestep
      integer ikle(nelmax*3),ikles(3,nelmax),i,ielem,itime
      integer ib(10), ivar,i, morph_number
      integer itrav(npoinmax)
      character*32 infile
      character*32 texte(maxvars),textmean,textmax,textmin,outfile
      character*16 descmean, descmax, descmin
      character*72 titre,line
      character*80 titsel
      logical eof
      external eof

8     read(5,998) line
998   format(a72)
      if(line(1:1).eq.'/') goto 8

9     read(5,999) line
999   format(a72)
      if(line(1:1).eq.'/') goto 9

      read(*,1001) infile
      open(12,file=infile,form='UNFORMATTED')

c10    read(5,999) line
c1000  format(a72)
c     if(line(1:1).eq.'/') goto 10
c
c     read(5,*) morphtimestep

11     read(5,1002) line
1002   format(a72)
      if(line(1:1).eq.'/') goto 11
      read(5,1001) outfile
1001  format(a32)

      rewind 12

c *********** read info from file ******************************

c... title

      call lit(x,w,itrav,titre,72,'CH',12,'STD',istat)
      write(*,*) 'Title of study:'
      write(*,*) titre
```

```
c... number of variables

      call lit(xbid,w,ib,titre,2,'I ',12,'STD',istat)
      nvar = ib(1)
      print *,ib(1)
c     write(*,*) ib(1),ib(2)

c... text for each variable
      do 100,k=1,nvar
        call lit(xbid,w,ibid,texte(k),32,'CH',12,'STD',istat)
c       print *,texte(k)
100   continue

c... 10 integer parameters

      call lit(xbid,w,ib,titre,10,'I',12,'STD',istat)


c... nelem etc

      call lit(xbid,w,ib,titre,4,'I',12,'STD',istat)
      nelem = ib(1)
      npoin = ib(2)
      ndp   = ib(3)
c     print *,nelem,npoin,ndp

c size checks
      if (nelem.gt.nelmax) then
        write(*,*) 'Too many elements'
        print *,nelem
        stop
      elseif (npoin.gt.npoinmax) then
        write(*,*) 'Too many nodes'
        stop
      endif
c     write(*,*) nelem,' elements, ',npoin,' nodes  ',ndp

c... ikle

      call lit(xbid,w,ikles,titre,nelem*ndp,'I',12,'STD',istat)

c... convert ikles to ikle
      do 110 i = 1,ndp
       do 115 ielem = 1,nelem
          ikle(ielem+(i-1)*nelem) = ikles(i,ielem)
115    continue
110   continue

c     write(*,*) ikle(1),ikle(1+nelem),ikle(1+2*nelem)

c... ipobo
c reads array itrav ( = 0 if interior node, = node no. if on boundary)

      call lit(xbid,w,itrav,titre,npoin,'I',12,'STD',istat)

c     write(*,*) itrav(1), itrav(200), itrav(2430)
c calculate number of boundary nodes:

      nptfr = 0
      if (npoin .ge.1) then
        do 120 i = 1,npoin
          if(itrav(i).ne.0) nptfr = nptfr + 1
120     continue
```

```
      endif

c     write(*,*) nptfr, ' nodes on boundary'


c... node coordinates

      call lit(x,w,ib,titre,npoin,'R4',12,'STD',istat)
      call lit(y,w,ib,titre,npoin,'R4',12,'STD',istat)

c     write(*,*) x(370),y(370)
c-------------------------------------------------------------
c Initialise output file
c write information to output results file
      open(13,file=outfile, form='UNFORMATTED')

c current test example, output file contains max of h only
      titsel = titre // '        '
      call ecrit(x,itrav,titsel,80,'CH',13,'STD',istat)
      ib(1) = nvar
      ib(2) = 0
      call ecrit(xbid,ib,titre,2,'I',13,'STD',istat)
      do 630 i = 1,nvar
      call ecrit(xbid,ibid,texte(i),32,'CH',13,'STD',istat)
630   continue
      ib(1) = 1
      do 651 i=2,10
651     ib(i) = 0
      call ecrit(xbid,ib,titre,10,'I',13,'STD',istat)
      ib(1) = nelem
      ib(2) = npoin
      ib(3) = ndp
      ib(4) = 0
      print *,'npoin ',npoin
      call ecrit(xbid,ib,titre,4,'I',13,'STD',istat)
      call ecrit(xbid,ikles,titre,nelem*ndp,'I',13,'STD',istat)
      call ecrit(xbid,itrav,titre,npoin,'I',13,'STD',istat)
      call ecrit(x,ib,titre,npoin,'R4',13,'STD',istat)
      call ecrit(y,ib,titre,npoin,'R4',13,'STD',istat)


c--------------------------------------------------------------
c   read times and results
       nstore = 0

500    if (eof(12)) goto 600
          call lit(time,w,itrav,titre,1,'R4',12,'STD',istat)
          write(*,*) 'Processing time ',time
           call lit(bottom,w,itrav,titre,npoin,'R4',12,'STD',istat)

        call ecrit(time,itrav,titre,1,'R4',13,'STD',istat)
          call ecrit(bottom,itrav,titre,npoin,'R4',13,'STD',istat)

       goto 500

600   continue
c*************************************************************
       open(20,status='old',file='newgeom.xyz')
       rewind(20)

       read(20,*) morph_number

c timestep is number of modelled tides mulitplied by scaling
```

```
c number and divided by 350 (spring tides in a year)

       morphtimestep=1.0*float(morph_number)/float(350)

       time_new = time + morphtimestep
       print *, time_new, time, morphtimestep


       do i=1,npoin
           read(20,*) bottom(i)
c          print *,i,bottom(i)

       enddo

       call ecrit(time_new,itrav,titre,1,'R4',13,'STD',istat)
       call ecrit(bottom,itrav,titre,npoin,'R4',13,'STD',istat)

       print *,'updated bathymetry file'


c****************************************************************


       stop

      close(12)
      close(13)
      close(20)

      end
```

Appendix 9 pick_morph routine

```
   program pick_morph

c             -------
c  reads in a Serafin results file (3-node triangular elements)
c  and outputs a single time level
c             -------

      implicit none
      integer nptfr,ndp,istat,nvar,k,ibid(2)
      integer nelmax,npoinmax,nelem,npoin,selafin,maxvars
      integer nstore,nstoremax,nt
      parameter (nelmax=60000,npoinmax=35000,nstoremax=90,maxvars=25)
      parameter(selafin=3)


      real w(npoinmax)
      double precision xbid(2),times(nstoremax),time,xtime
      double precision f(npoinmax,maxvars)
      double precision x(npoinmax),y(npoinmax)
      double precision max(npoinmax),min(npoinmax),mean(npoinmax)
      integer ikle(nelmax*3),ikles(3,nelmax),i,ielem
      integer ib(10), ivar
      integer itrav(npoinmax)
      character*32 infile
      character*32 texte(maxvars),textmean,textmax,textmin,outfile
      character*16 descmean, descmax, descmin
      character*72 titre
      character*80 titsel
      logical eof
      external eof


      write(*,*) 'Enter input file (must be Selafin format)'
      read(*,1001) infile
      open(12,file=infile,form='UNFORMATTED')
      write(*,*) 'Enter required time'
      read (*,*) xtime

      rewind 12

c *********** read info from file ******************************

c... title

      call lit(x,w,itrav,titre,72,'CH',12,'STD',istat)
      write(*,*) 'Title of study:'
      write(*,*) titre
c... number of variables

      call lit(xbid,w,ib,titre,2,'I ',12,'STD',istat)
      nvar = ib(1)
c     write(*,*) ib(1),ib(2)

c... text for each variable
      do 100,k=1,nvar
        call lit(xbid,w,ibid,texte(k),32,'CH',12,'STD',istat)
100   continue

c... 10 integer parameters
```

```
      call lit(xbid,w,ib,titre,10,'I',12,'STD',istat)


c... nelem etc

      call lit(xbid,w,ib,titre,4,'I',12,'STD',istat)
      nelem = ib(1)
      npoin = ib(2)
      ndp   = ib(3)
c     write(*,*) nelem,' elements, ',npoin,' nodes  ',ndp

c... ikle

      call lit(xbid,w,ikles,titre,nelem*ndp,'I',12,'STD',istat)

c... convert ikles to ikle
      do 110 i = 1,ndp
       do 115 ielem = 1,nelem
          ikle(ielem+(i-1)*nelem) = ikles(i,ielem)
115    continue
110   continue

c     write(*,*) ikle(1),ikle(1+nelem),ikle(1+2*nelem)

c... ipobo
c reads array itrav ( = 0 if interior node, = node no. if on boundary)

      call lit(xbid,w,itrav,titre,npoin,'I',12,'STD',istat)

c     write(*,*) itrav(1), itrav(200), itrav(2430)
c calculate number of boundary nodes:

      nptfr = 0
      if (npoin .ge.1) then
        do 120 i = 1,npoin
          if(itrav(i).ne.0) nptfr = nptfr + 1
120     continue
      endif

c     write(*,*) nptfr, ' nodes on boundary'


c... node coordinates

      call lit(x,w,ib,titre,npoin,'R4',12,'STD',istat)
      call lit(y,w,ib,titre,npoin,'R4',12,'STD',istat)

c     write(*,*) x(370),y(370)

c****************************************************************
c Print out variable descriptions

      write(*,*) 'The following variables have been read in:'
      do 610 i = 1,nvar
        write(*,*) i,'  ',texte(i)
610   continue

      write(*,*) 'Enter filename for results'
      read(*,1001) outfile
1000  format(a16)
1001  format(a32)
```

```
c****************************************************************
c write header information to output results file
      open(13,file=outfile, form='UNFORMATTED')

c current test example, output file contains max of h only
      titsel = titre // '        '
      call ecrit(x,itrav,titsel,80,'CH',13,'STD',istat)
      ib(1) = nvar
      ib(2) = 0
      call ecrit(xbid,ib,titre,2,'I',13,'STD',istat)
      do 630 i = 1,nvar
      call ecrit(xbid,ibid,texte(i),32,'CH',13,'STD',istat)
630   continue
      ib(1) = 1
      do 651 i=2,10
651     ib(i) = 0
      call ecrit(xbid,ib,titre,10,'I',13,'STD',istat)
      ib(1) = nelem
      ib(2) = npoin
      ib(3) = ndp
      ib(4) = 0
      call ecrit(xbid,ib,titre,4,'I',13,'STD',istat)
      call ecrit(xbid,ikles,titre,nelem*ndp,'I',13,'STD',istat)
      call ecrit(xbid,itrav,titre,npoin,'I',13,'STD',istat)
      call ecrit(x,ib,titre,npoin,'R4',13,'STD',istat)
      call ecrit(y,ib,titre,npoin,'R4',13,'STD',istat)
c---------------------------------------------------------------
c   read times and results

500    if (eof(12)) goto 600
          call lit(time,w,itrav,titre,1,'R4',12,'STD',istat)
          write(*,*) 'Reading time ',time
          do 510 i = 1,nvar
c           write(*,*) name
            call lit(f(1,i),w,itrav,titre,npoin,'R4',12,'STD',istat)
510       continue

c      write(*,*) time,u(1),v(1),h(1),surf(1),fond(1)
       if (time .ne.xtime) goto 500

600   continue
c-------------------------------------------------------------------
      call ecrit(time,itrav,titre,1,'R4',13,'STD',istat)

      do 2000 i = 1,nvar

      call ecrit(f(1,i),itrav,titre,npoin,'R4',13,'STD',istat)
2000  continue


      close(12)
      close(13)

      stop

      end
```