GEOLOGICAL SURVEY OF CANADA

OPEN FILE 2338

Classical analytical modelling of the offshore temperature profiles, Beaufort Shelf

P.P. Sidhu

1991

UNIVERSITY OF WATERLOO
Faculty of Engineering

CLASSICAL ANALYTICAL MODELLING OF THE OFFSHORE TEMPERATURE PROFILES, BEAUFORT SHELF

PREPARED BY:

P.P. Sidhu
CO-OP student,
Dept. of Civil Engineering,
University of Waterloo,
Waterloo, Ontario

May 1990

with the support of
Panel on Energy Research and Development,
EMR Canada

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES

LIST OF TABLES

SUMMARY

Two simple classical analytical models were undertaken to demonstrate the dynamic response of the thermal regime to the recent marine transgression at the Angasak, Amauligak, and at the four industrial sites Koakoak, Nerlerk, Kopanoar, and Ukalerk. An attempt was made to demonstrate the extent to which a simple model can be applied to explain the temperature profiles.

The classical models, the input parameters, and temperature profiles are described. The results of the modelling and comparison with the relative sea level curve are shown, conclusions drawn, and recommendations made.

It was concluded that geothermal modelling appears to be a useful tool in recreating past geological events but the time of marine transgression predicted by these models was a minimum because the effects of near-surface, latent heat, and salinity were not modelled. These models failed to explain the strong curvature in the industry temperature data but provided a reasonable fit for precise temperatures measured by the Geological Survey of Canada.

It is recommended that complete precise temperature profile would be useful to minimize the number of estimated input parameters that had to be made and that a detailed lithological analysis should be undertaken for the industrial wells to explain the strong curvature observed in their temperature profiles.

CONCLUSIONS

Geothermal modelling appears to be a useful tool in recreating past geological events, particularly the subsurface effects of marine transgression.

The time of marine transgression predicted by simple analytical models was a minimum, ie. the most recent probable date, because the effects of near-surface, latent heat, and salinity were not accounted for.

Fitting the isothermal profile observed at water depths greater than 20 m was very sensitive to the choice of seabed and permafrost base temperatures. In contrast, fitting the large, negative temperature-depth gradients observed at more shoreward sites was not particularly sensitive to these boundary conditions.

These classical analytical models failed to explain the strong curvature in the industry temperature data. This problem is of a special concern at Ukalerk, some 20 km to the east of Amauligak and at a similar water depth, but may represent lithological conditions of which there is little knowledge.

The value in the geothermal approach lies in its ability to estimate the nature of marine transgression from information obtained much deeper than geomorphic evidence. A valuable byproduct of these simple analytical models for more advanced numeric modelling is an appreciation of the sensitivity of seabed thermal behaviour to such parameters as sub-aerial temperatures prior to transgression, seabed temperatures after, the distance from present shoreline and ground thermal properties.

The major limitation is the degree of sophistication of the model and the need for simplifying assumptions; however, more complex models may require a knowledge of physical properties such as ice content, salinity, and submergence temperature histories that are often not known or are poorly constrained.

RECOMMENDATIONS

More complete geotechnical information and precision measured temperature profile would be useful to minimize the number of estimated input parameters that had to be made. A deeper temperature profile would likely improve the results at Angasak since the effect of latent heat would be reduced.

Detailed lithological analysis should be undertaken for the industrial wells to explain the strong curvature observed in the temperature profiles, since the latter are not adequately explained by the classical analysis.

In 1990, four 80 m temperature profiles (2 to 11 m water depth) were obtained along potential pipeline corridor between Mackenzie Delta and Amauligak field. The modelling programs developed herein should be undertaken on these profiles.

The more advanced finite difference models developed by Nixon [5] or Outcalt [6] that invoke salinities and latent heat can be applied to the offshore wells for comparison.

ACKNOWLEDGMENTS

The funding for this project was obtained from the Panel on Energy Research and Development, Energy Mines and Resources Canada. The author thanks Al Taylor for valuable comments on the manuscript and Jeff Weaver for the discussion on the industrial well logs.

1.0 INTRODUCTION

1.1 OFFSHORE PERMAFROST, CANADIAN BEAUFORT SHELF

Most of the continental shelf of the Beaufort Sea (Fig. 1) is underlain by ice-bonded permafrost (IBPF) which was first confirmed in 1970 when frozen icy sediments and ice were encountered in some borehole samples during industrial drilling [1]. Permafrost is defined as any soil, subsoil, rock, or surficial deposit occurring at a variable depth beneath the earth's surface in which a temperature below 0 °C has existed for at least two consecutive years. This definition is based exclusively on temperature disregarding texture, degree of compaction, water content, and lithologic characteristics of the material. Thus the problem of both the thickness and distribution of the permafrost is essentially a thermal one and may be determined from geophysical well logs of ground temperatures. The thickest permafrost is found when a combination of low surface temperatures , low terrestrial heat flow and high thermal conductivity exist [2].

Offshore permafrost has important implications because of the interest in oil and gas production from offshore reservoirs. To solve engineering problems such as the construction of Arctic ports, building of pipelines and of production well completions, geotechnical knowledge of soil properties and material behaviour throughout the permafrost zone are required. An understanding of the state of these materials can be acquired from the temperature profile. To design a well casing that can withstand the stresses and strains that occur when the surrounding permafrost thaws, the temperature profile and thermal properties are required to calculate the rate of thaw. Special problems are encountered with offshore exploration wells since they may be drilled close together in clusters (Fig.2) due to the expense of building a man-made island. This results in a greater heat source and thus more thaw may be encountered [3,4]. When these temperature measurements are not available the ground thermal regime will have to be predicted by modelling of nearby wells.

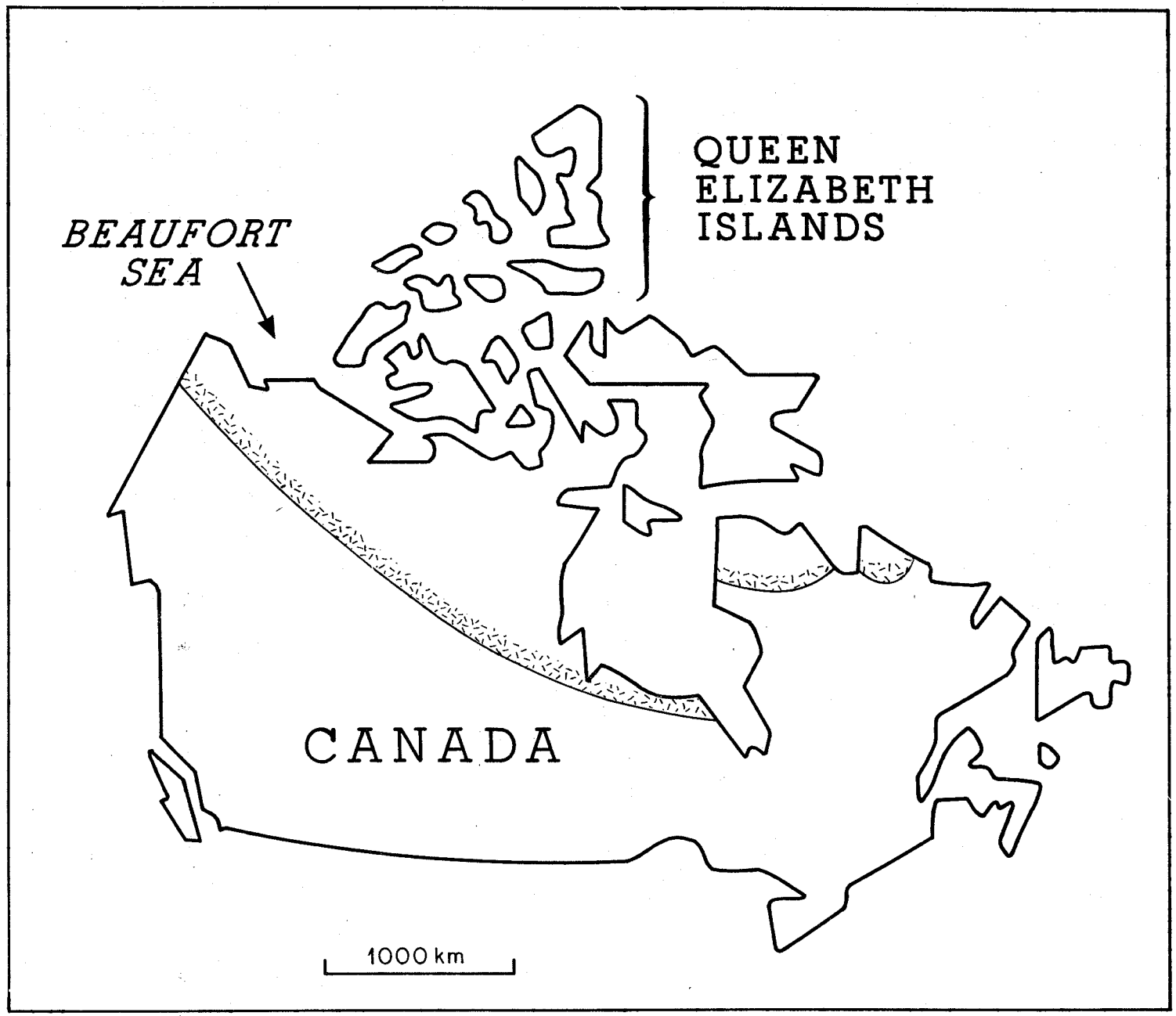


Figure 1.0 : Location Of Beaufort Sea Region

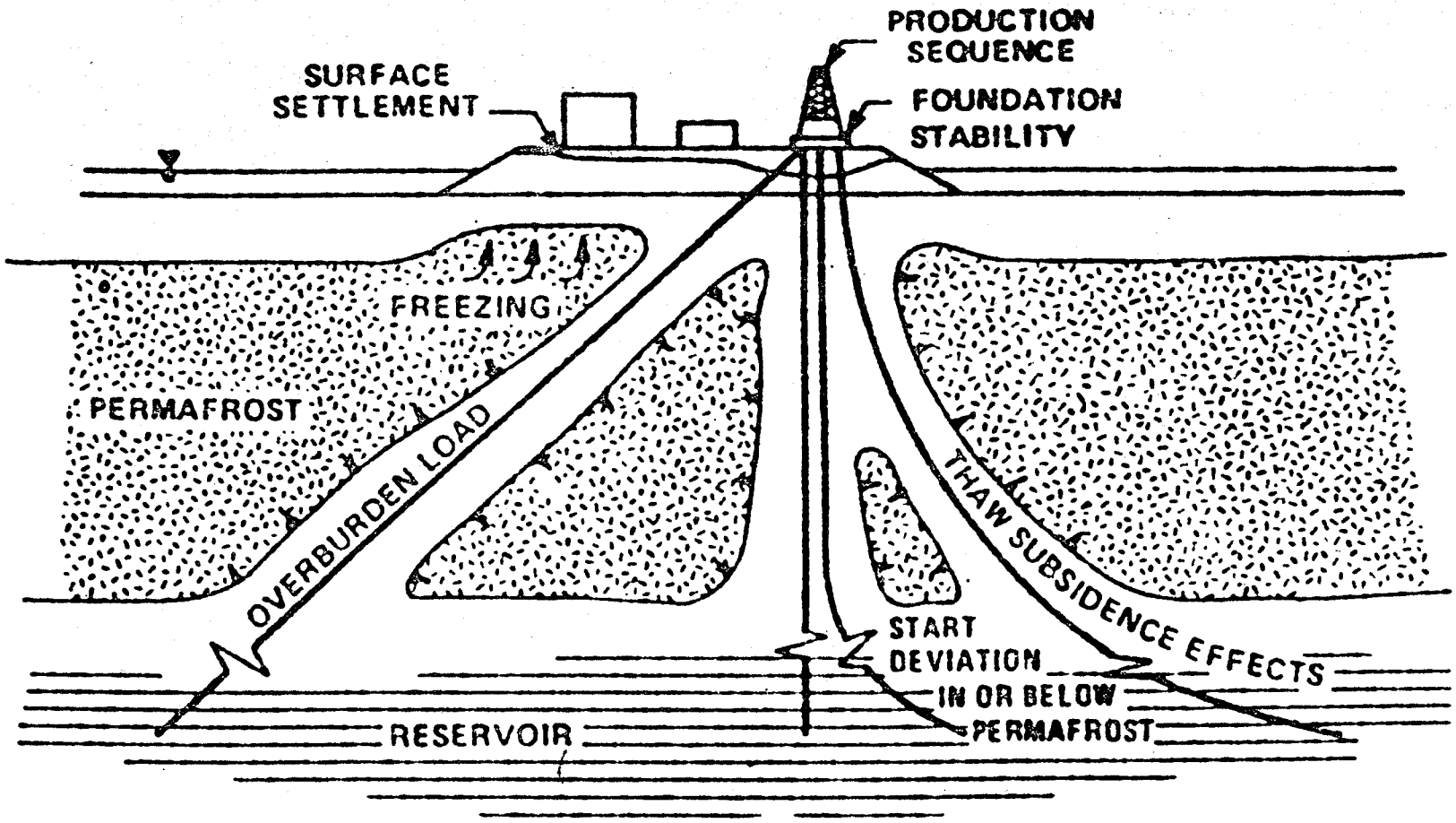


Figure 2 : Engineering Surface Settlement Diagram (Hayley [3])

1.2 SCOPE OF REPORT

In the Beaufort Shelf, the thickness of the permafrost and the temperature regime have been predicted on theoretical considerations [1,2,5,6] while thicknesses of ice-bonding have been estimated at 500-700 m from geophysical well logs [7], and some deep temperature measurements have been reported by industry [8]. Recently, precise documented temperature measurements have been reported for the Esso Angasak L-03 wellsite [9] and the Gulf Amauligak wellsite [10] by the Geological Survey of Canada.

A unique feature of the Beaufort Shelf that distinguishes it from most other major continental shelves is its thick permafrost. These frozen sediments are now slowly melting due to the marine transgression that occurred in the Holocene.

Simple classical analytical models have been undertaken to demonstrate the dynamic response of the thermal regime to the recent marine transgression at the Angasak, Amauligak, and at the four industrial sites Koakoak, Nerlerk, Kopanoar, and Ukalerk as reported by Weaver and Stewart (Fig. 3). An attempt is made to demonstrate the extent to which a simple model can be applied to explain the temperature profiles.

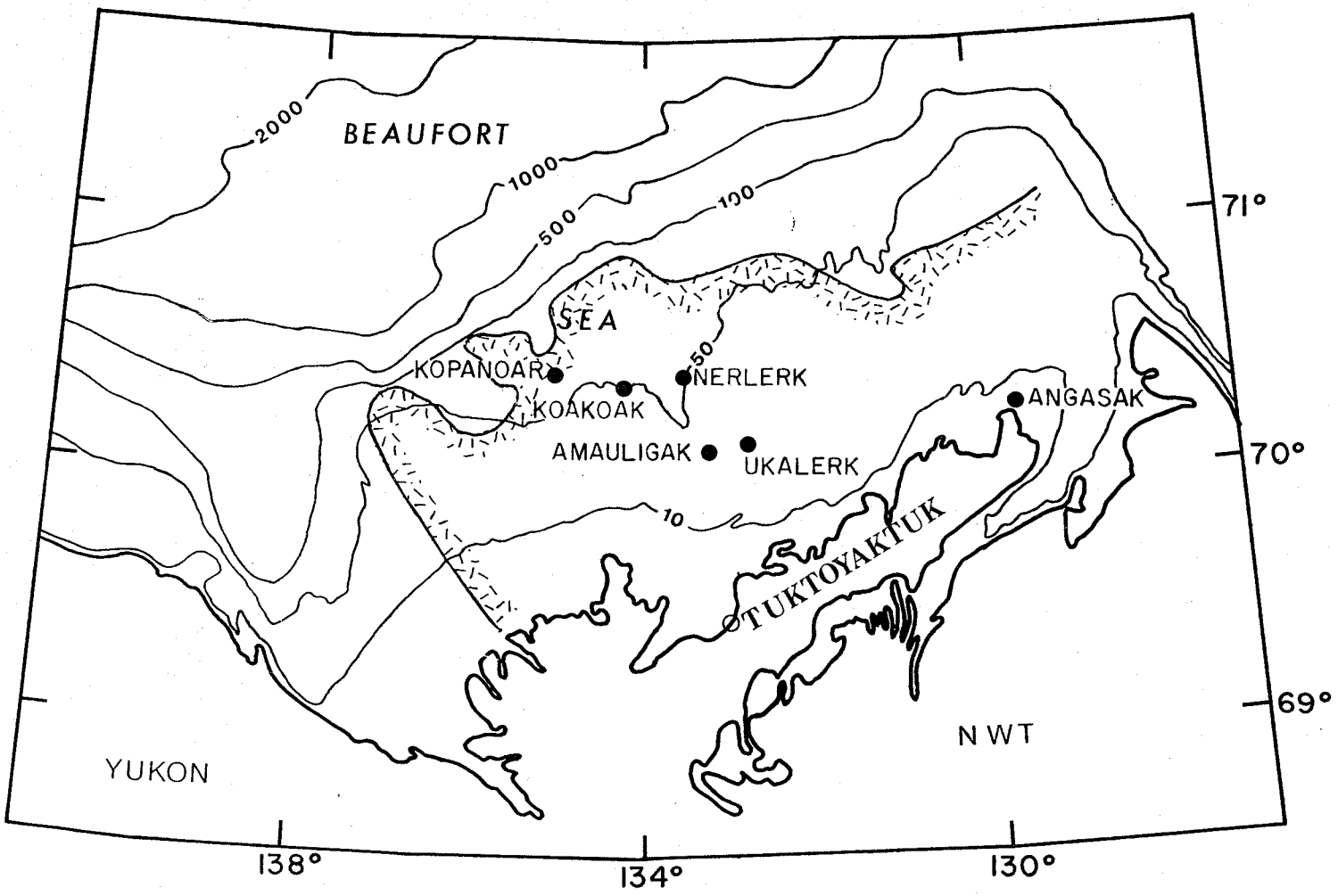


Figure 3 : Location Of Wellsites Within The Mackenzie Delta-Beaufort Sea Region

2.0 REGIONAL GEOLOGY

The Beaufort Shelf attained its principal physiographic elements and bathometric features in the Quaternary through the agents of deltaic formation, glaciation and deglaciation, sea-level changes and associated transgressions of the sea [11].

Most of the Beaufort Shelf was above sea level with mean surface temperatures estimated to be between -8 to -16 °C [12,13] during the mid to late Wisconsinan. These cold temperatures resulted in the growth of permafrost. An event of relatively recent geological history during the Holocene is marine transgression; this represents a major thermal event as the sea water near its freezing point inundates the sub-aerial permafrost landscape. As inundation proceeds, a warm thermal pulse propagates into the seabed, gradually raising the ground temperatures from a state of equilibrium with an Arctic sub-aerial environment towards one of equilibrium with a marine environment. Thousands to ten of thousand of years may be required to reach this new equilibrium and the permafrost is therefore considered to be relict. Recent sea-level changes are due mainly to the eustatic fluctuations in sea level, isostatic effects are minimal because most of the coast was unglaciated during the late Wisconsin. Therefore, offshore permafrost is presently degrading in response to marine transgression [1,14].

The presence of deep permafrost below the Beaufort Sea can only be explained by past exposure to very cold mean ground surface temperatures for a long time prior to submergence. Using radiocarbon-dated peat and peaty clay samples from geotechnical boreholes, Hill et al. [14] reconstructed a late Quaternary relative sea-level curve (Fig. 4). Peat and organic samples which were deposited in a freshwater marshy environment were identified in numerous boreholes in their study and are considered a fairly reliable indication of the time of submergence. Their work suggests that the Shelf must have been sub-aerial for at least 16,000 years before the recent marine transgression.

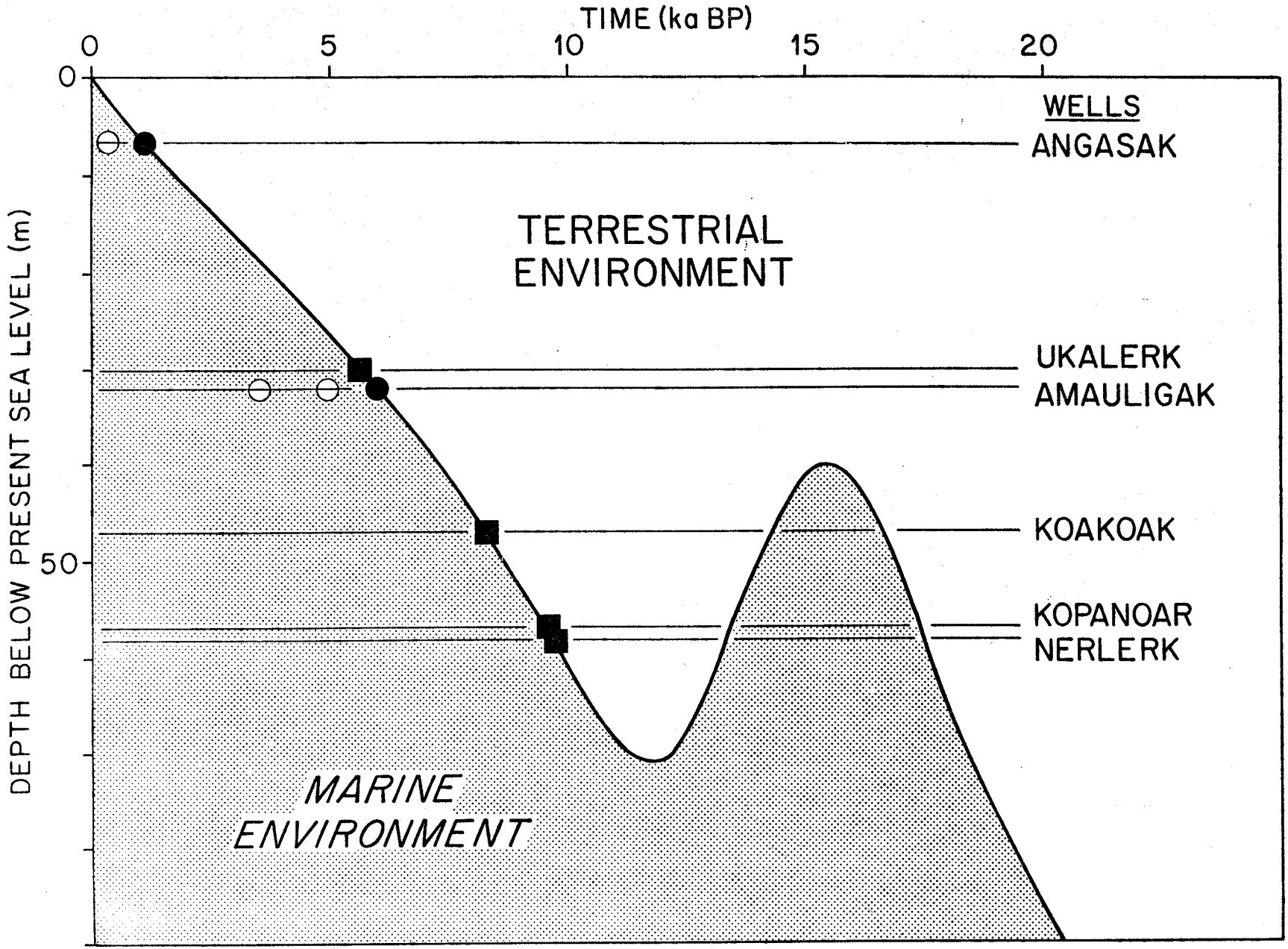


Figure 4 : Sea Level Curve For The Beaufort Sea From Palynological And Geomorphic Evidence (Hill et al [14])

3.0 TEMPERATURE PROFILES

3.1 ANGASAK

A multithermistor cable was installed in a 56 m geotechnical hole drilled from the sea ice at the Esso Angasak wellsite (Fig. 3). An automatic data logger was attached to the cable and a time series of precise temperature measurements were recovered. Details on installation of the thermistor cable and the recovery of the temperature data can be found in the paper by Taylor and Judge [9]. The temperature-depth profile for the Angasak wellsite is shown plotted in Figure 5. The Angasak site is only 2 km from the present shoreline in water depth of 6.7 m. The depth of permafrost in the area of Angasak ranges from 400 to 500 m [7], therefore temperatures only in the top 15 percent of the permafrost zone were recovered.

3.2 AMAULIGAK

The multithermistor cable at the Gulf Amauligak wellsite was installed in a geotechnical hole drilled from a caisson-retained island, the Molikpaq (Fig 6). This geotechnical hole is much deeper than at Angasak extending to a depth below the sea level of 336.5 m. An automatic data logger was attached at the wellhead and a time series of precise temperatures were recovered. Details on the installation of the thermistor cable, recovery of the temperature data, and the drilling and abandonment process can be found in a paper by Taylor et al [10]. The temperature-depth profile for the Amauligak wellsite is shown plotted in Figure 7. This wellsite is approximately 30 Km offshore with a water depth of 32 m. The depth of permafrost in the area of Amauligak ranges from 600 to 700 m [7], therefore temperatures in the top 50 percent of the permafrost zone were recovered.

3.3 INDUSTRIAL WELLS : Koakoak, Nerlerk, Kopanoar, & Ukalerk

Temperature profiles to more than 1000 m measured by industry as a part of geophysical well logging have been reported by Weaver and Stewart [8] for the Kopanoar, Koakoak, Nerlerk and

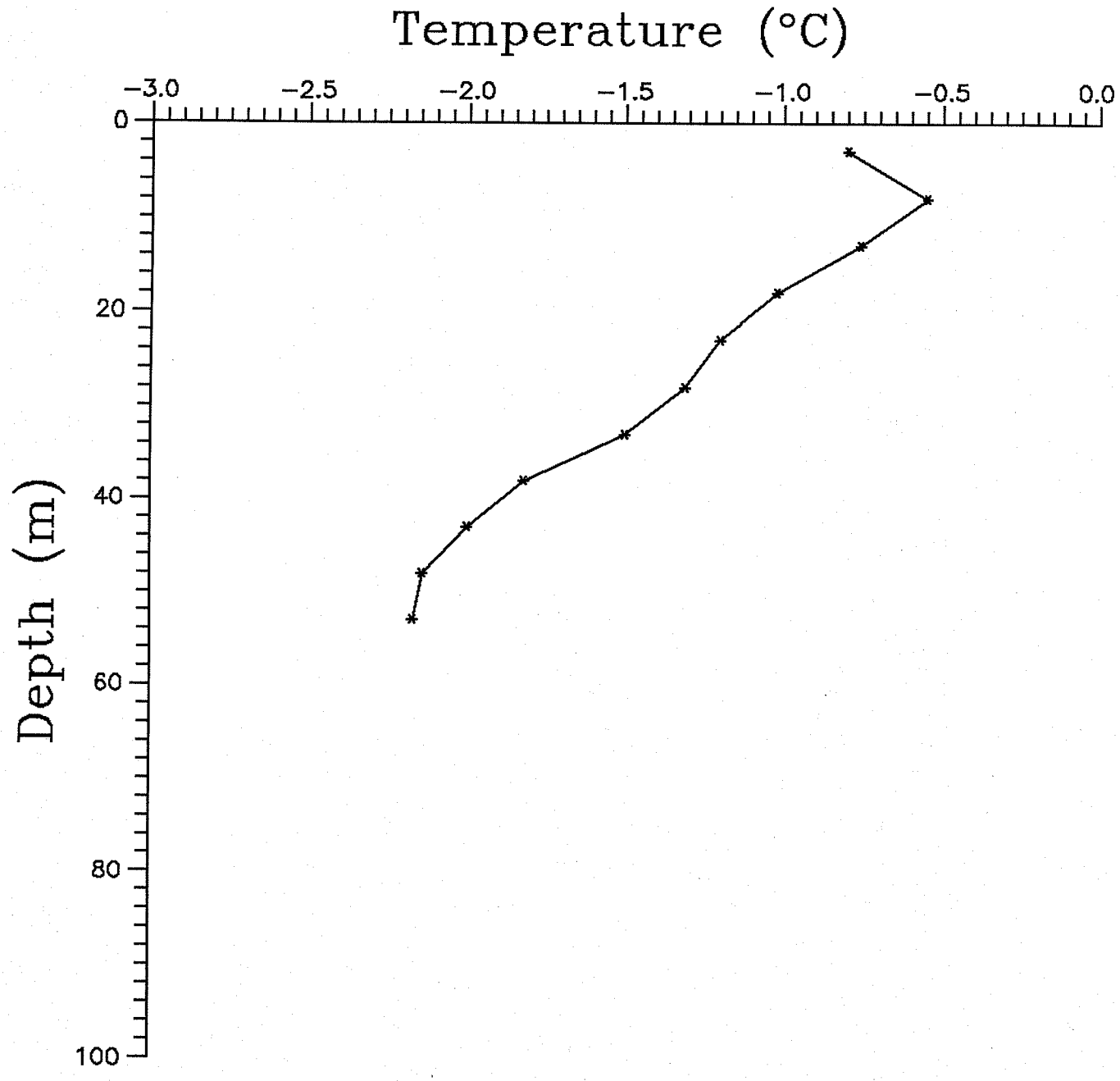


Figure 5 : Angasak Offshore Temperature Profile

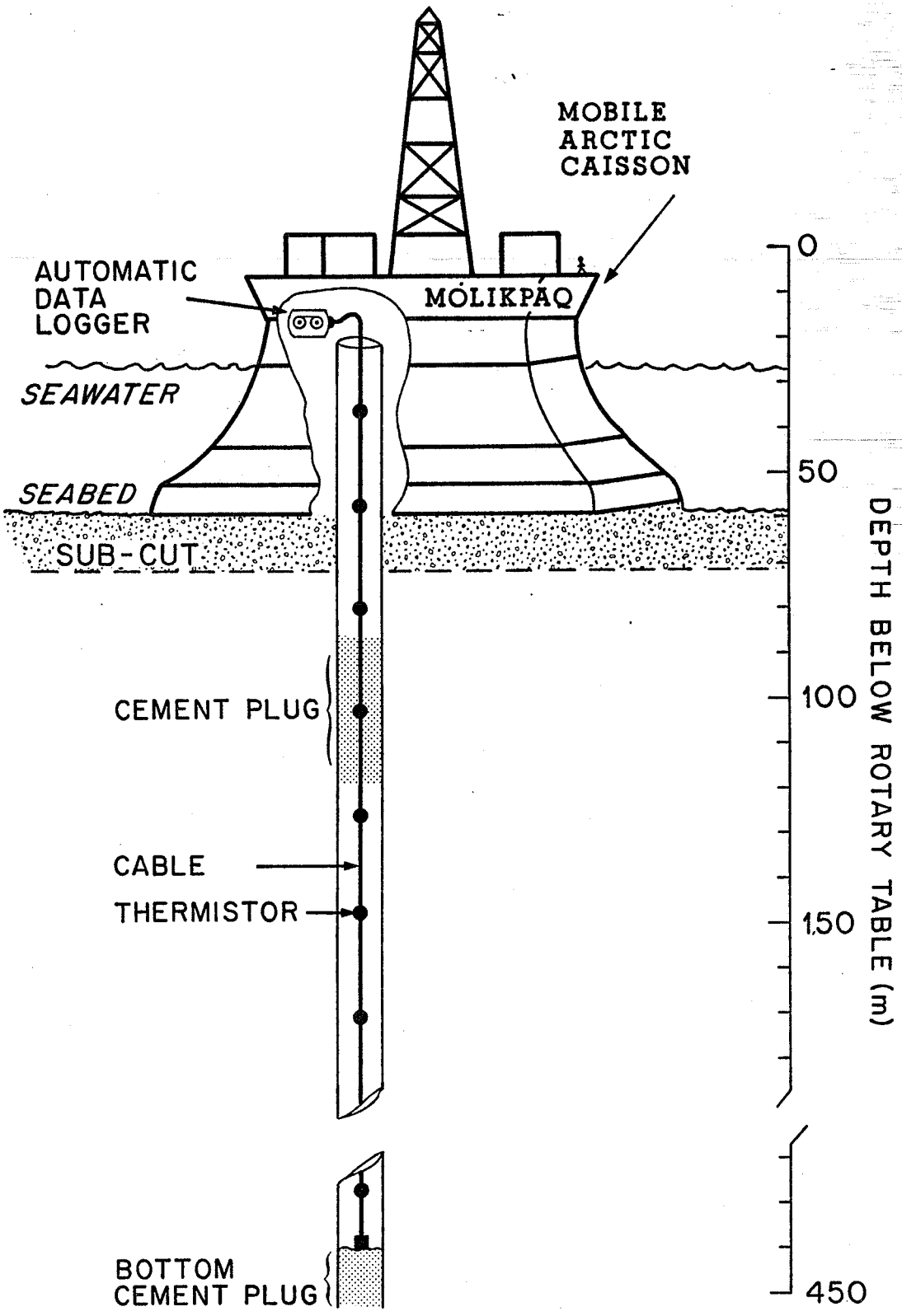


Figure 6 : Sketch Of The Molikaq And The Instrumentation In The Geotechnical Hole (Taylor et al. [10])

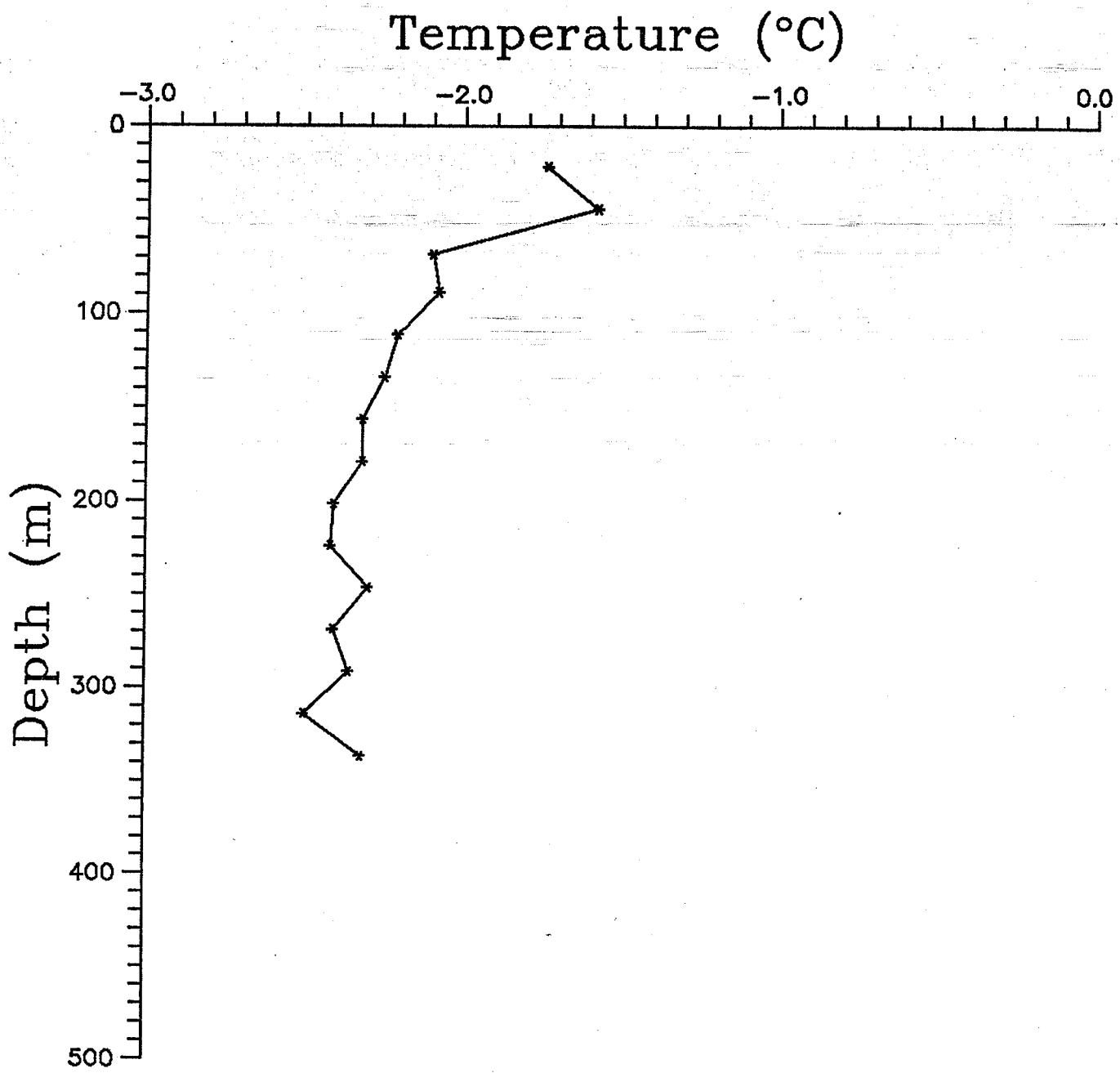


Figure 7 : Amauligak Offshore Temperature Profile

Ukalerk wells (Fig. 3). Weaver (personal communication 1990) estimates measurement error as much as 2 or 3 °K. These less precise temperature profiles have been plotted in Figure 8. The permafrost thickness at Ukalerk, Koakoak, Kopanoar, and Nerlerk are estimated at 780, 750, 480, and 680 meters respectively. The water depths for Ukalerk, Koakoak, Kopanoar, and Nerlerk are 30, 48, 57, and 58 meters respectively.

Comparing the temperature profiles for these industrial wells with the Angasak and Amauligak temperature profiles, we note that there is substantial curvature observed in the upper portion for the industrial graphs even though they are at a greater water depth then the Angasak well and approximately at the same water depth as the Amauligak well. The Amauligak temperature profile is almost isothermal with depth and since Ukalerk is only 20 km to the east of Amauligak and at similar water depth one might expect a comparable profile between these two wells. The Angasak profile reflects a relatively early stage of permafrost degradation while the Amauligak is much more advanced. This is consistent with the curve by Hill et al. (Fig. 4) since the Amauligak well is at a greater water depth.

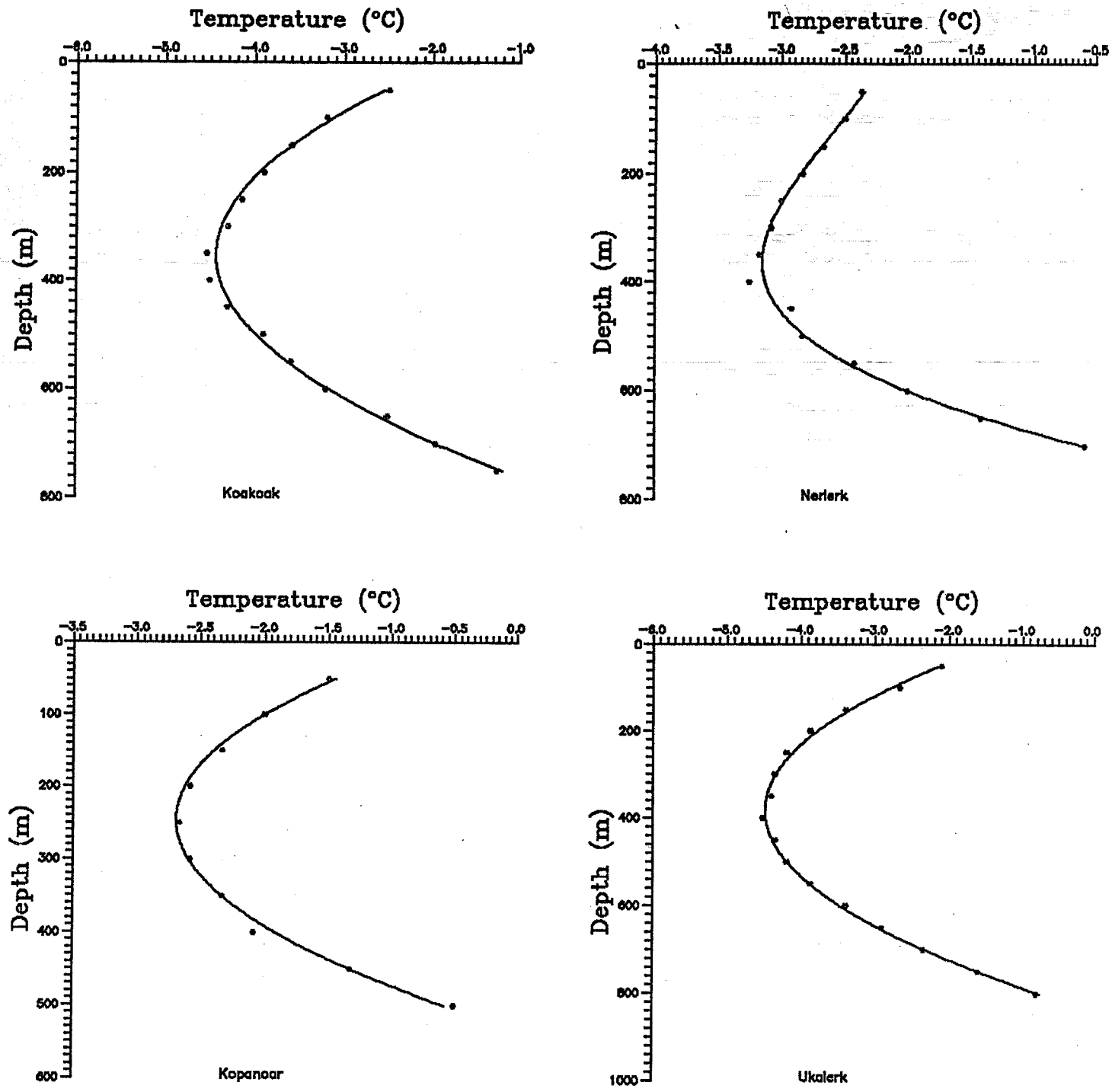


Figure 8 : Temperature Profile For The Industrial Wells Koakoak, Nerlerk, Kopanoar, & Ukalerk

4.0 CLASSICAL ANALYTICAL MODELS

Two simple, analytical models that employ a minimum of parameterization are used to explain the gross features of the preceding temperature profiles. The first is a two dimensional model that predicts subsurface temperature-depth gradient anomalies near shorelines in time following marine transgression or regression. This model was developed by Lachenbruch [15] and will be referred to as Lachenbruch '57 in this report. The second equation is a one dimensional model that predicts the transient variation in subsurface temperatures in offshore permafrost following marine transgression. This model was developed by Lachenbruch et al [16] and will be referred to as Lachenbruch et al. '82 in this report.

4.1 LACHENBRUCH '57

The problem of determining the contribution of the ocean to the changes in the subsurface thermal regime where the shoreline has shifted at some time in the past but has since remained stable may be described by [15, eq. 14]

$$\frac{dT(x,z,t)}{dz} = -A\left\{ \frac{x}{\pi(z^2+x^2)} \exp-\left(\frac{z^2+x^2}{4\alpha t}\right) + \frac{1}{2\sqrt{(\pi\alpha t)}} \exp-\left(\frac{z^2}{4\alpha t}\right)\left[1+\mathrm{erf}\frac{\alpha}{2\sqrt{(\alpha t)}}\right] \right\}$$

{eq. 1}

where

A = difference between the mean annual temperatures of the land surface and the seabed (K)
α = thermal diffusivity of the ground materials (m^2/s)
t = time since shoreline shifted (s)
z = depth below seabed (m)
x = distance from shoreline (m)
erf = error function

The thermal diffusivity of the materials can be calculated by using the equation

$$\alpha = \frac{K}{\rho c} \qquad \{eq.\ 2\}$$

where

K = thermal conductivity of the materials ($W \cdot m^{-1} \cdot K^{-1}$)
ρ = density (Kg/m^3)
c = specific heat ($J \cdot Kg^{-1} \cdot K^{-1}$)

4.2 LACHENBRUCH '82

The subsurface temperature field T(z,t) following inundation of a site due to marine transgression may be described by [16, eq. 27]

$$T(z,t) = (T_o - T_s)\frac{2}{\pi} \sum_{n=1}^{n=\infty} \left[\frac{1}{n} \exp\left(\frac{-n^2\pi^2 t}{4\lambda}\right) \sin\left(\frac{n\pi z}{Z}\right) \right] \quad \{eq.\ 3\}$$

where

T_o = mean annual surface temperature before transgression (°C)
T_s = seabed temperature after inundation (°C)
t = time since submergence (s)
z = depth below seabed (m)
Z = equilibrium permafrost thickness before submergence (m)
λ = thermal time constant (s)

The equilibrium permafrost thickness before submergence can be calculated from

$$Z = \frac{K_{avg}(T_o - T_s)}{Q} \quad \{eq.\ 4\}$$

where

K_{avg} = average thermal conductivity ($W \cdot m^{-1} \cdot K^{-1}$)
Q = terrestrial heat flow ($W \cdot m^{-2}$)

The thermal time constant can be calculated from the following equation

$$\lambda = \frac{Z^2}{4\alpha} \quad \{eq.\ 5\}$$

after obtaining the thermal diffusivity from equation 2 and the equilibrium permafrost thickness before submergence from

equation 4.

In the development of the above models the following assumptions have been made: (1) the transgression occurred in a short period at some time in the past and (2) the subsurface temperature and lithology beneath the seabed at the time of submergence were similar to those beneath the adjacent land today. These simple analytical models do not illustrate the influence of latent heat and salinity on the thermal regime.

4.3 COMPUTER SIMULATIONS

The program GRAD in Appendix I was written to numerically model the two dimensional equation 1 for the calculation of transient gradient anomaly due to marine transgression including the effect of a nearby shoreline. The program TRANS in Appendix II was written to numerically model the one dimensional equation 3 for the calculation of transient warming term for submarine permafrost due to marine transgression.

Both programs were written using modular design and then compiled with Microsoft FORTRAN 77 version 5.0. The SI system of units are used throughout the programs. The main features of the programs are the reading of depth-temperature measurement data from an input ASCII file, the plotting of the model fit on various output devices, and allowing the user to enter the fitting parameters (T_o, T_s, A, x, t) from the keyboard.

4.4 MODEL PARAMETERS

These models require an estimate of the subsurface temperatures prior to inundation.

Taylor and Judge [17] extrapolated temperatures in the upper 50 m at four petroleum exploratory wells in the Tuktoyaktuk Peninsula and reported mean surface temperatures today between -9 and -10 °C. Values between -11 and -13.5 °C were taken as estimates for this parameter. These are several degrees cooler than today but perhaps are typical of lower average temperatures during the late Wisconsin and early Holocene in the region [12,13].

Based on a knowledge of the geology of the region, Judge [2] estimated the terrestrial heat flow to be between 63 and 84 $mW \cdot m^{-2}$. A value of 70 $mW \cdot m^{-2}$ was taken for the ESSO Angasak wellsite. For the Amauligak and the four industrial wellsites a value of 49 $mW \cdot m^{-2}$ was used as a more recent analysis by Majorowicz and Dietrich [18] suggests low geothermal gradients and low heat flow in that area.

A specific heat of 1150 $J \cdot Kg^{-1} \cdot K^{-1}$ and a density of 2500 Kg/m^3 were assumed to be the average representation of the various lithologies. The average thermal conductivity of the Angasak wellsite was measured to be 3.2 $W \cdot m^{-1} \cdot K^{-1}$, estimated to be 3.0 $W \cdot m^{-1} \cdot K^{-1}$ (Taylor, personal communication 1990) for the Amauligak wellsite, and was calculated from temperature gradients to be 2.8, 3.0, 2.0, and 3.2 for Koakoak, Nerlerk, Kopanoar, and Ukalerk respectively. Table I summarizes the calculations for the thermal conductivities for the industrial wells. Detailed calculations for Nerlerk are placed in Appendix III.

Permafrost thicknesses prior to the recent transgression were calculated by the program from these parameters using equation 4.

INDUSTRIAL WELL	TEMPERATURE GRADIENT ($mK \cdot m^{-1}$)	CONDUCTIVITY UNFROZEN ($W \cdot m^{-1} \cdot K^{-1}$)	CONDUCTIVITY OF GRAINS ($W \cdot m^{-1} \cdot K^{-1}$)	CONDUCTIVITY FROZEN ($W \cdot m^{-1} \cdot K^{-1}$)
KOAKOAK	26.2	1.9	3.1	2.8
NERLERK	24.5	2.0	3.4	3.0
KOPANOAR	36.3	1.3	1.9	2.0
UKALERK	23.0	2.1	3.7	3.2

Table I : Summary Of The Thermal Conductivities For The Industrial Wells, Calculated From The Temperature Gradient

5.0 RESULTS OF COMPUTER MODELLING

The terrestrial heat flow, the thermal conductivity, and the thermal diffusivity were held constant and the ground temperature-depth profile in equilibrium with the earlier Arctic sub-aerial conditions is calculated from

$$T(z) = T_o + \left(\frac{Q}{K_{avg}}\right) z \qquad \text{(eq. 6)}$$

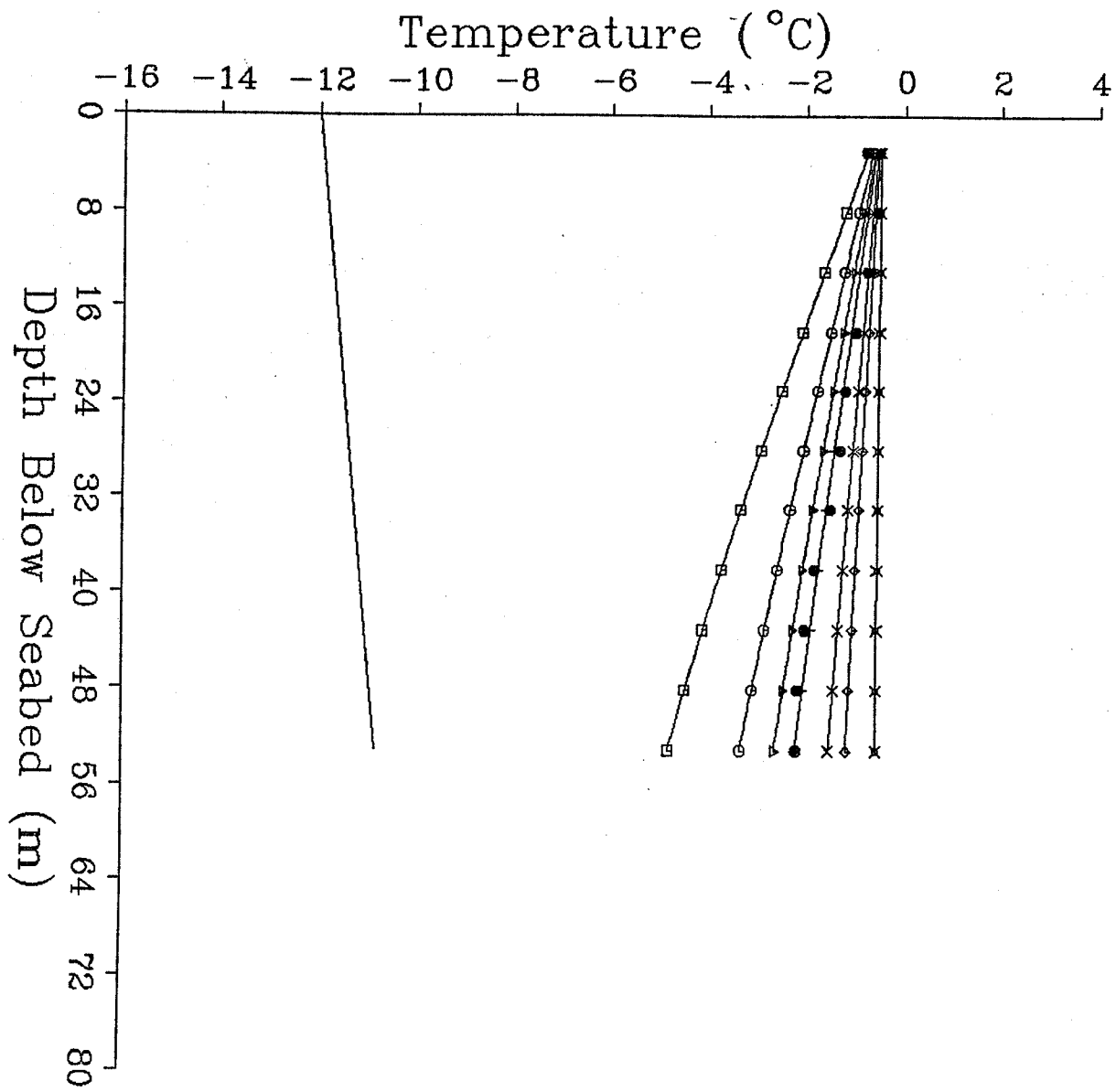
Using Lachenbruch et al. '82 model, subsurface temperatures were calculated for a range of pre-transgression surface temperatures, for several post-transgression seabed temperatures and for a range of times of inundation. The calculated temperatures were compared to the measured data, and the best model was selected in a least squares sense. Lachenbruch '57 model was applied to the Angasak wellsite to include the effect of the proximity (2 km) of the site to the present shoreline.

The results of the modelling for each site are presented in the following sections.

5.1 ANGASAK MODELLING RESULTS

Three computer runs were undertaken to match the temperature profile observed for Angasak. The measured temperature profile matches the run with a surface temperature (T_o) of -12 °C, and a seabed (T_s) and permafrost base (T_b) temperature of -0.5 °C (Fig. 9). From equation 4 these surface and permafrost base temperatures indicate an equilibrium permafrost thickness (Z) of 518 m. The measured temperatures (+/- 0.01 K precision) are shown by the solid dots and fit a time after inundation of approximately 400 years.

Lachenbruch '57 model was applied to predict the anomalous temperature-depth gradient expected at sites near the shoreline. Figure 10 shows the temperature gradient anomalies predicted by the model for A=+12 °C and x=2000 m. By averaging the times for the measured gradient anomalies shown as solid dots, a value of 360 years was obtained for marine inundation. The additional computer runs for Angasak and the times for the



ESSO ANGASAK

Fitting Parameters

THO = -12.00
THS = -0.50
THB = -0.50

Input File : ANGASAK.DT1

Output File : ANGASAK.OT2

Time Since Inundation [yrs]

□ 100.0 × 700.0
○ 200.0 ◇ 1000.0
△ 300.0 ⋈ 3000.0
+ 400.0 ● Measured Data

Figure 9 : Lachenbruch et al. '82 Model Fit Of Measured Temperatures At The Angasak Site

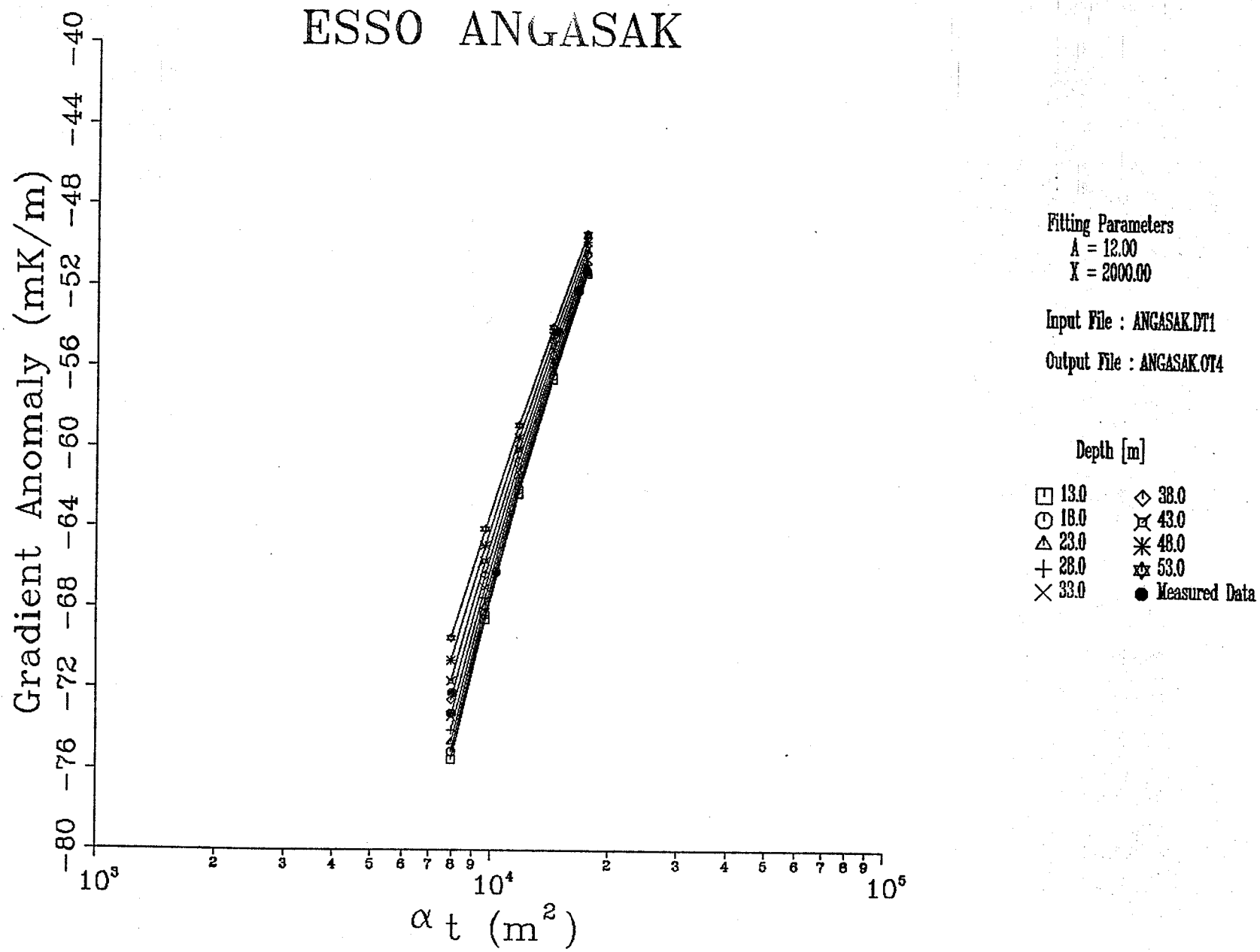


Figure 10 : Lachenbruch '57 Model Fit Of Measured Temperatures At The Angasak Site

calculated and measured gradient anomalies are placed in Appendix IV.

Figure 11 shows the anomalous gradient versus a reduced time since inundation. A lower limit of 0.1 years and a upper limit of 200 000 years were used and then divided into twenty logarithmic intervals to obtain the curve of the anomaly at a particular depth. This figure demonstrates three important features: (1) the effect of the inundation is greatest at the shallow depths, (2) it onsets at later times at greater depths and (3), the anomaly at all depths disappears at longer times after the inundation. Note that figure 10 is detail of one small part of this graph.

5.2 AMAULIGAK MODELLING RESULTS

Five computer runs were undertaken to match the temperature profile observed for the Amauligak wellsite. The temperatures predicted for a surface temperature (T_o) of -12 °C and a seabed (T_s) and permafrost base (T_b) temperature of -1.8 °C for various times since transgression were plotted in Figure 12. Using these surface and permafrost base temperatures in equation 4 a permafrost thickness (Z) of 625 m is calculated. The measured temperatures are shown by the solid dots and lie on the 3000 year profile, somewhat less then the 5800 years since inundation predicted by the relative sea level curve (Fig 4). Note that for nearly isothermal temperatures, the fit is quite sensitive to the choice of seabed and permafrost base temperatures. A choice of -1 °C would decrease the predicted time since inundation by 1000 years and a choice of -2 °C would increase the predicted time since inundation by almost a 1000 years. Runs containing these additional parameters are placed in Appendix IV.

5.3 INDUSTRIAL WELLS MODELLING RESULTS

Figures 13,14,15 and 16 show the position of the temperatures (+/- 2 °K precision) measured by industry [8] within the field of temperatures predicted at various times since inundation by classical modelling. Table II summaries the model best fitting the measured data for these wells. Transgression in this table

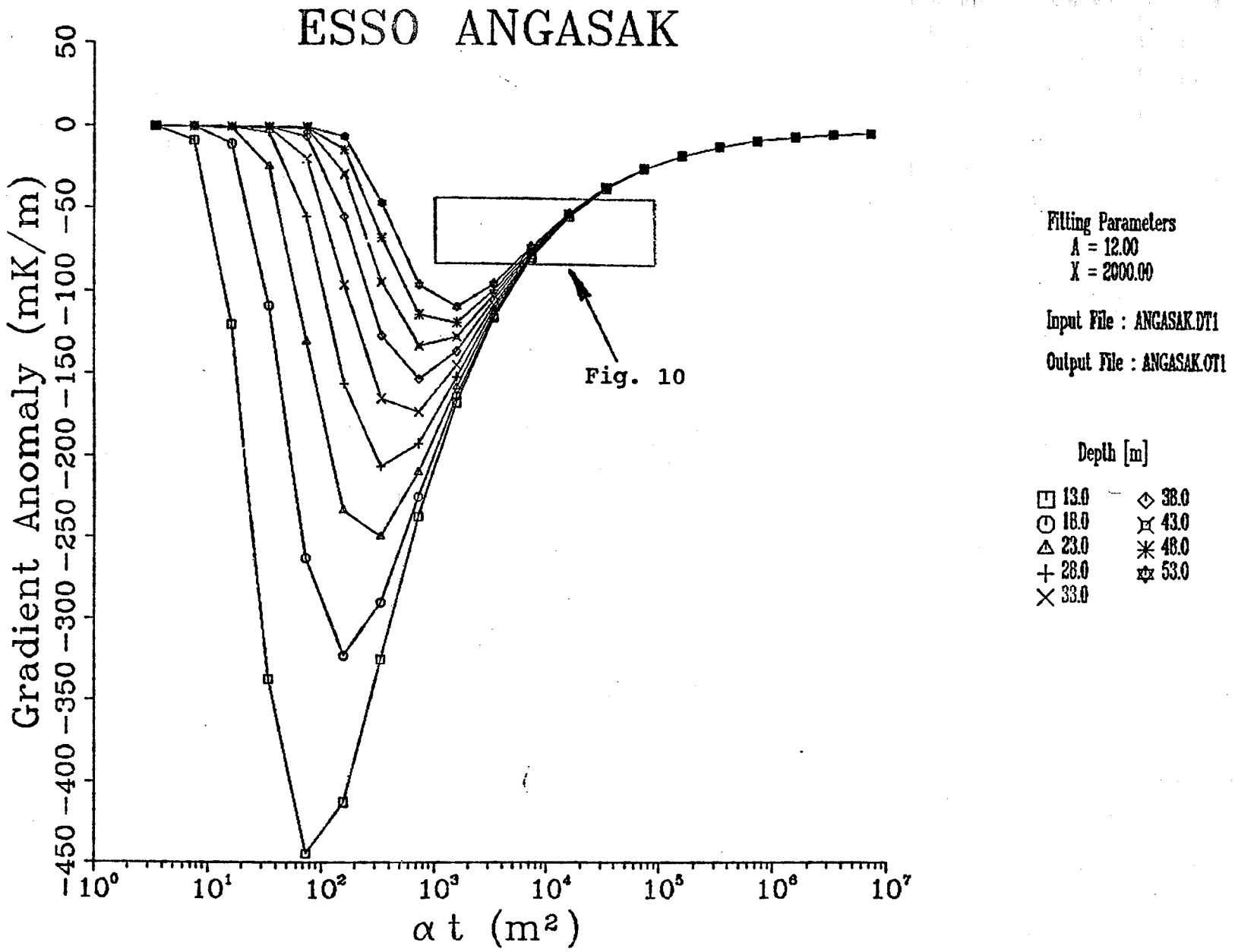


Figure 11 : Anomalous Gradient Versus A Reduced Time Since Inundation

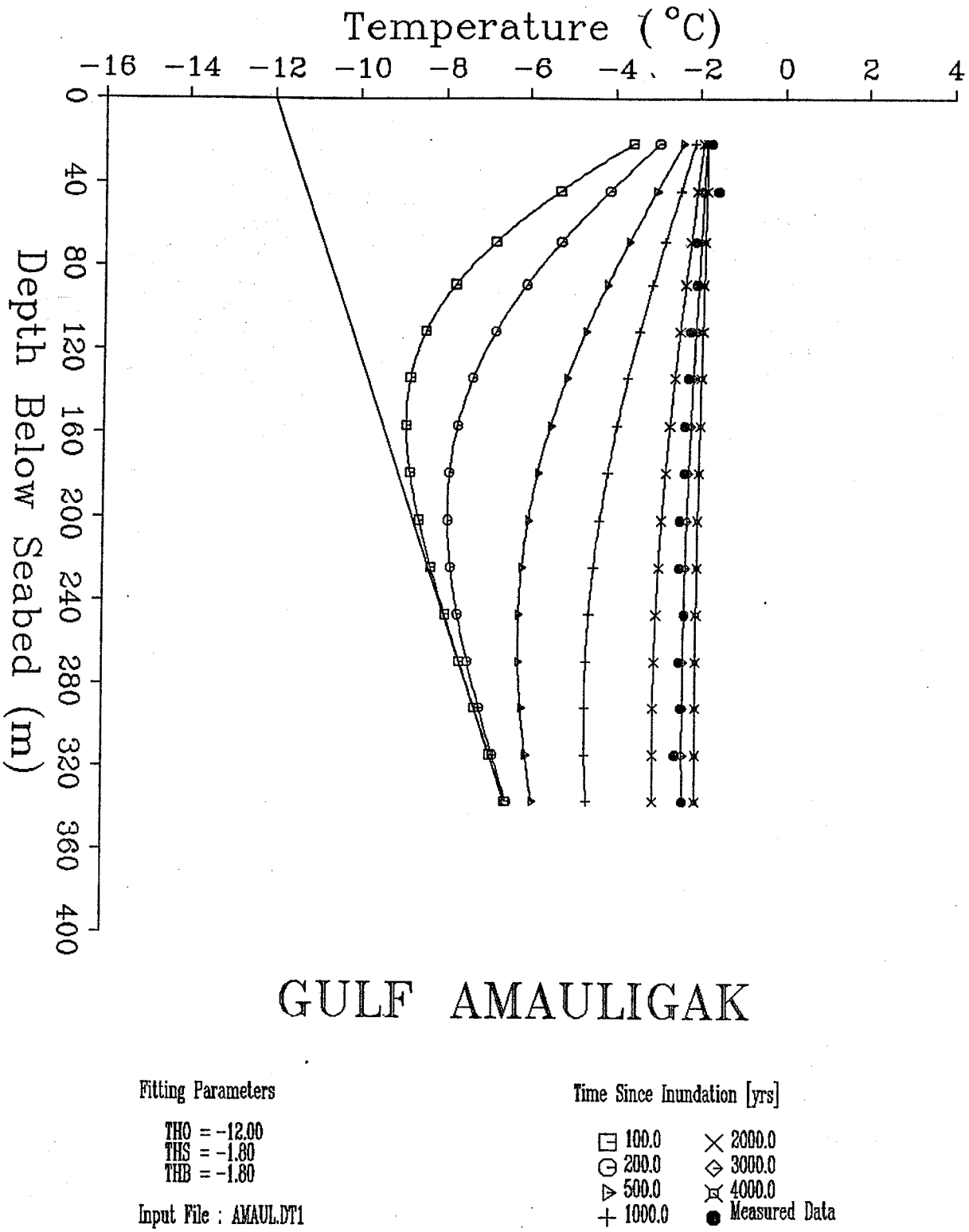


Figure 12 : Lachenbruch et al. '82 Model Fit Of Measured Temperatures At The Amauligak Site

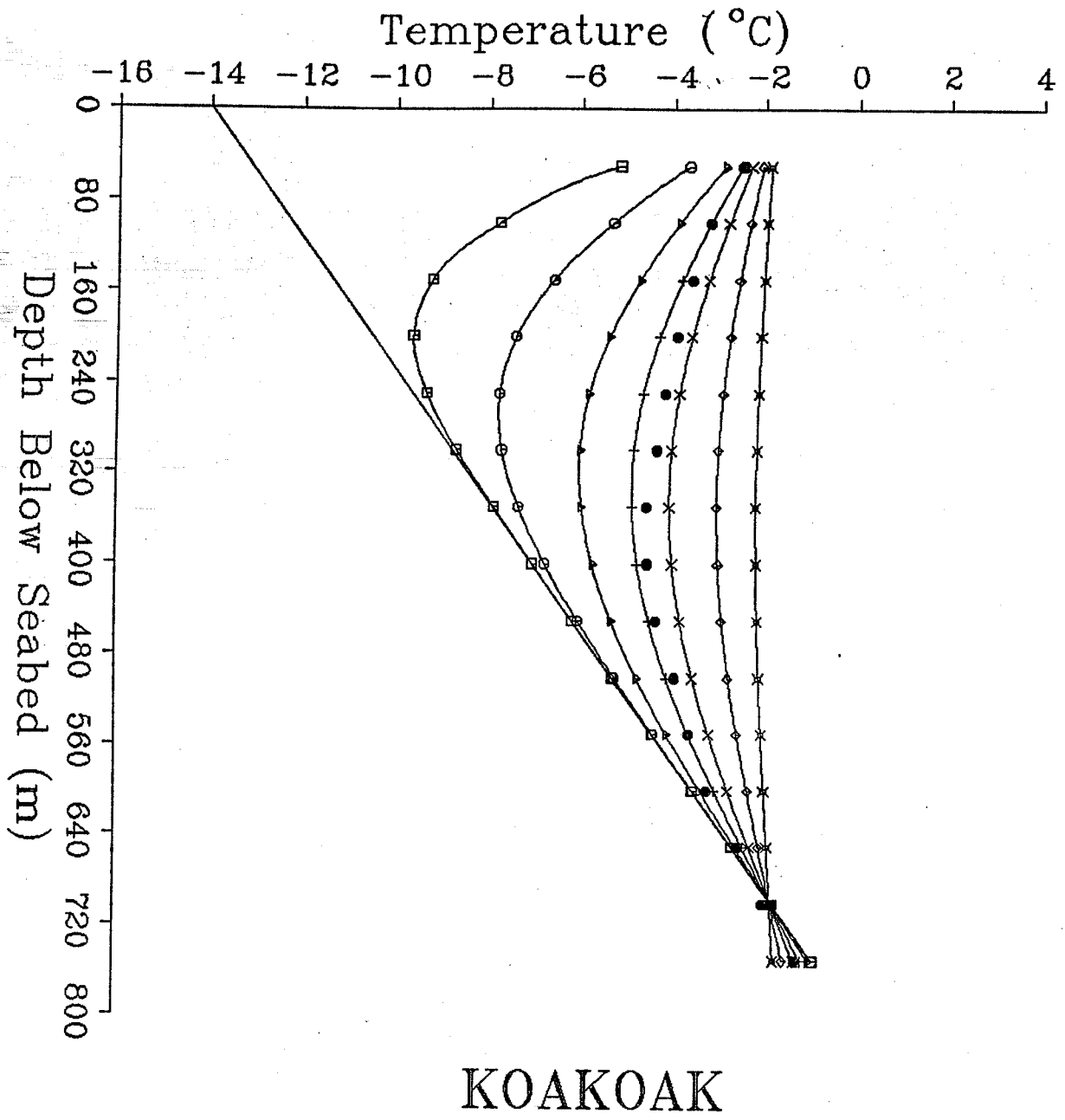


Figure 13 : Lachenbruch et al. '82 Model Fit Of Measured Temperatures At The Koakoak Site

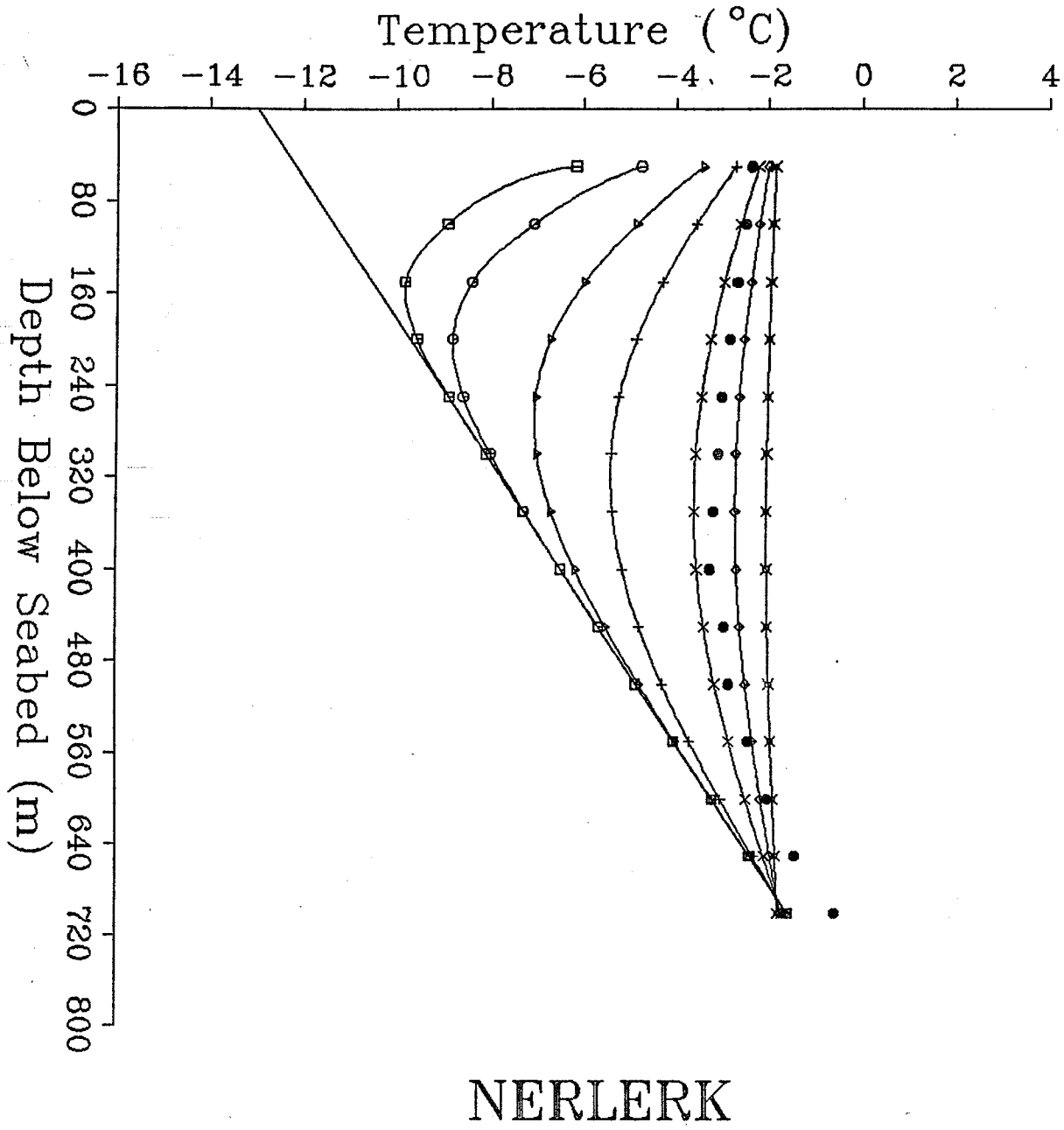


Figure 14 : Lachenbruch et al. '82 Model Fit Of Measured Temperatures At The Nerlerk Site

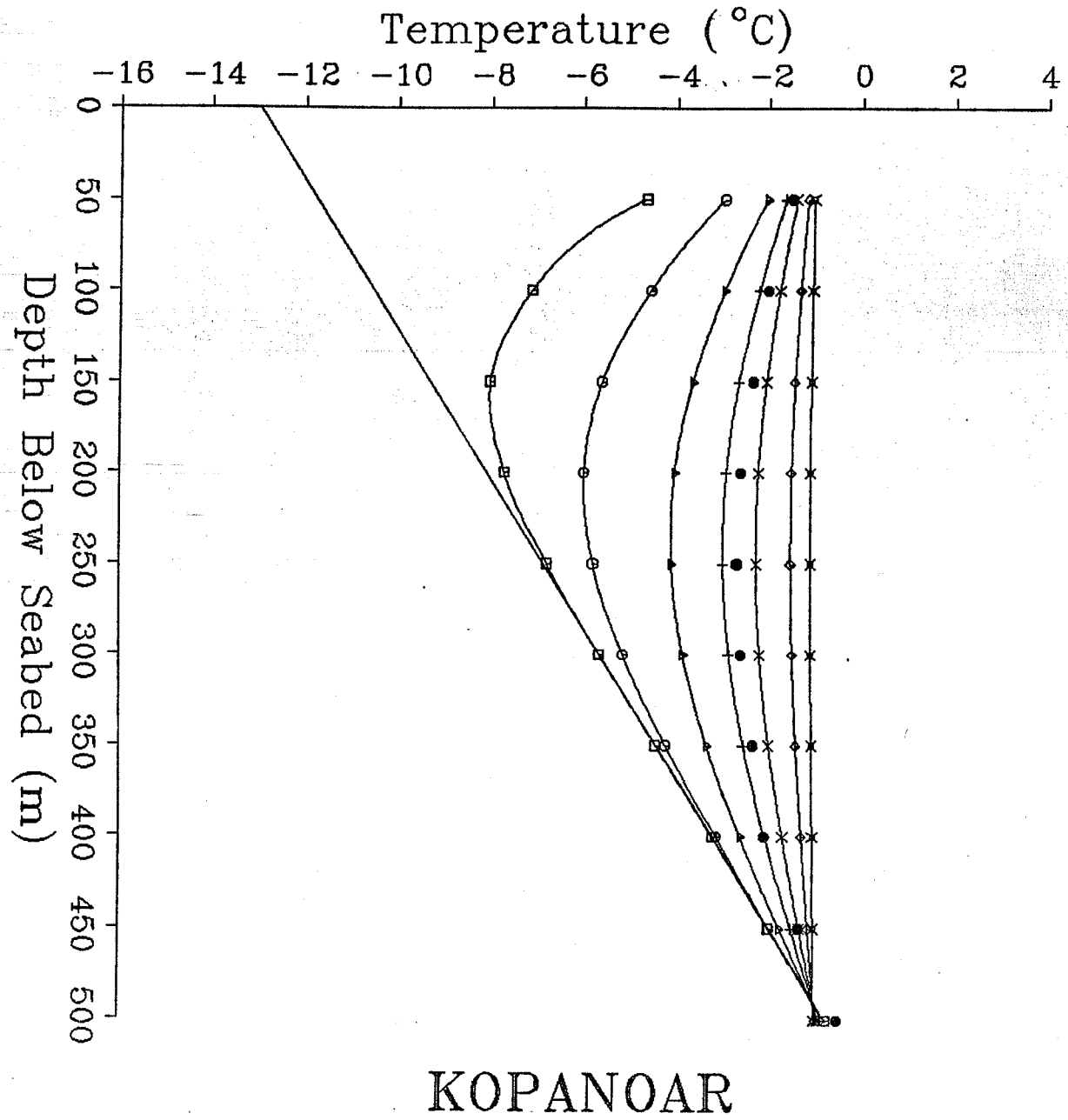


Figure 15 : Lachenbruch et al. '82 Model Fit Of Measured Temperatures At The Kopanoar Site

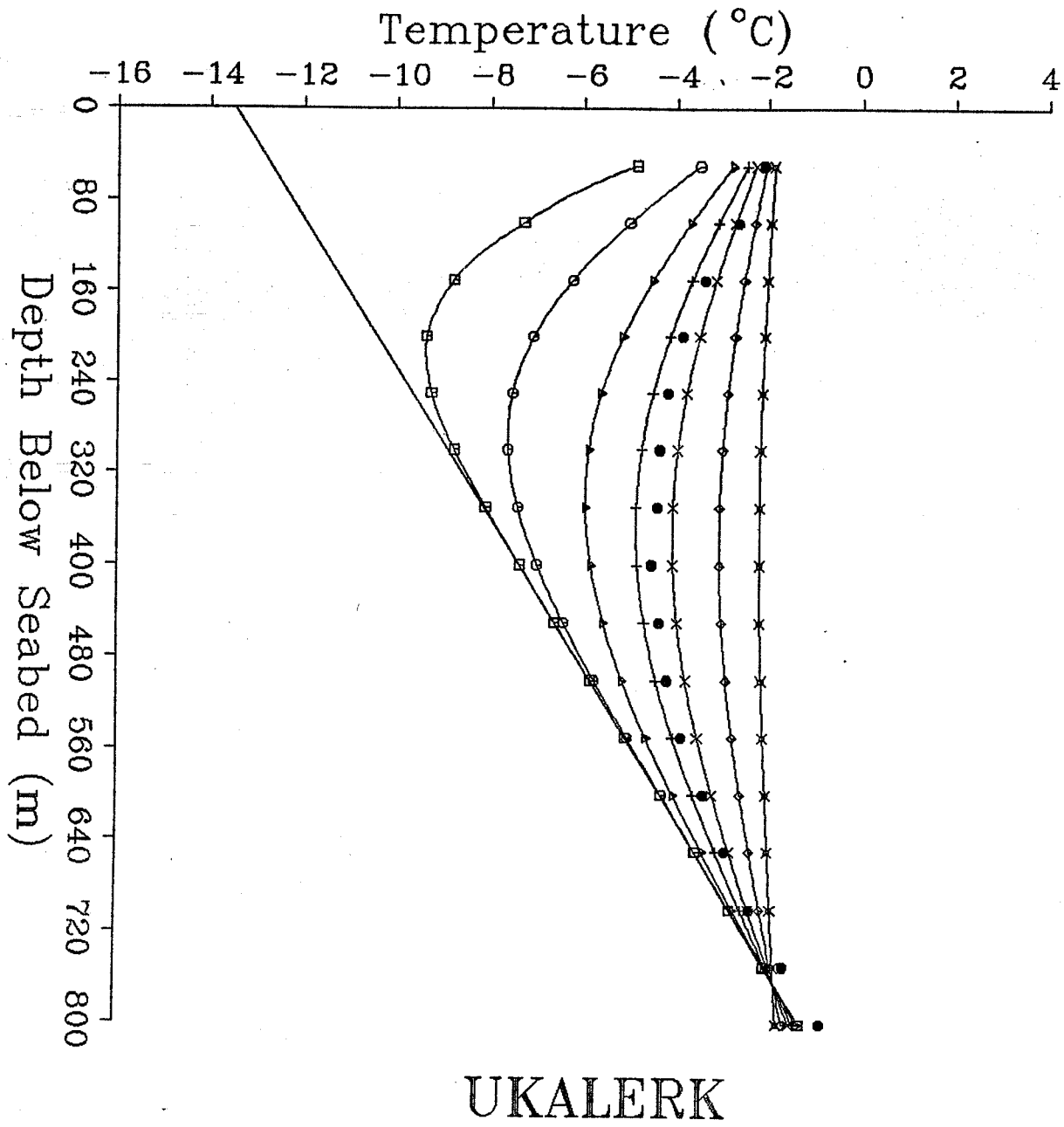


Figure 16 : Lachenbruch et al. '82 Model Fit Of Measured Temperatures At The Ukalerk Site

is given in years before the present (a BP). Additional model fits were run by varying the temperature parameters (T_o,T_s,T_b) and placed in Appendix IV.

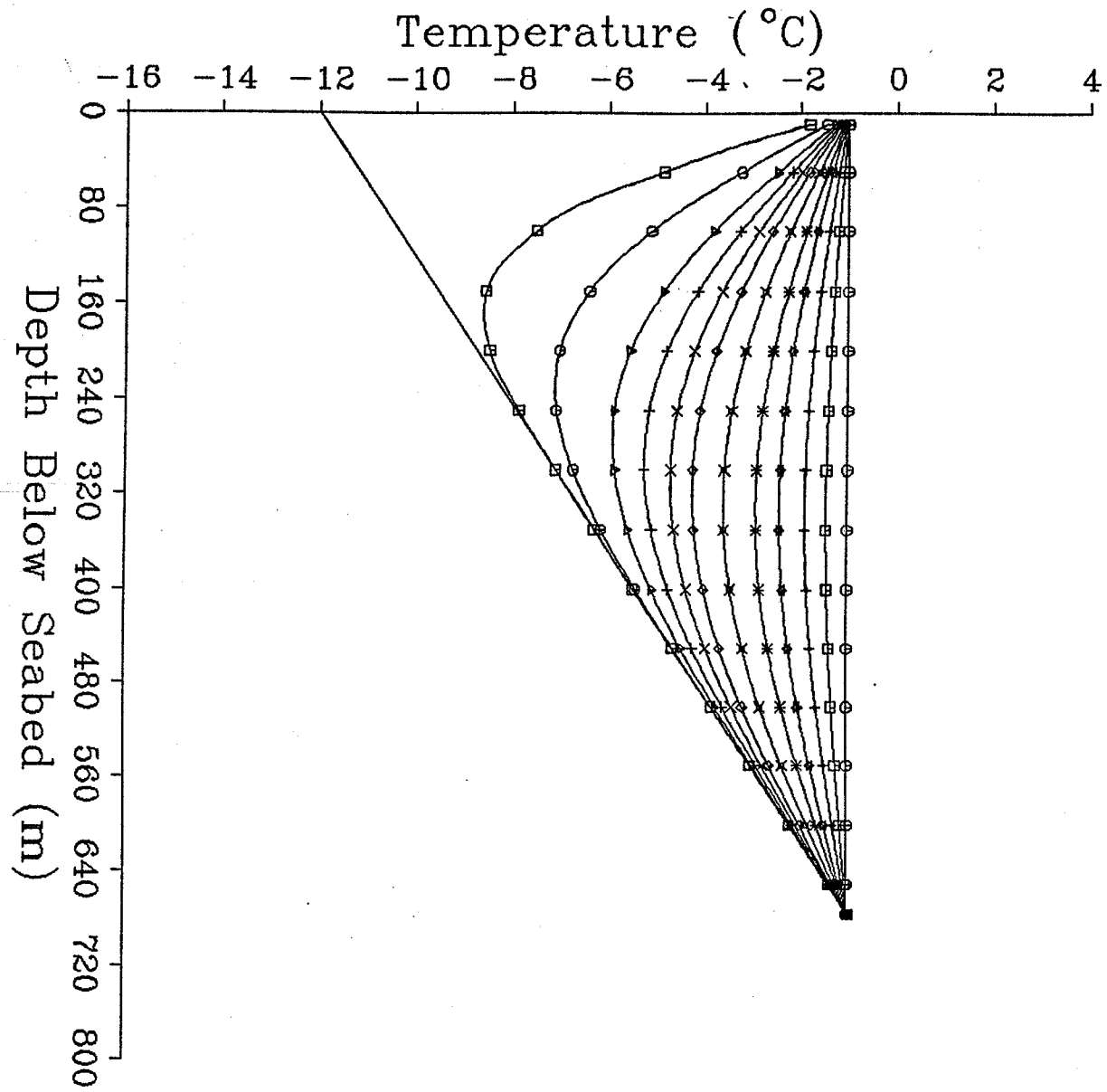
INDUSTRIAL WELL	T_o (°C)	T_s (°C)	T_b (°C)	Z (m)	TRANSGRESSION (a BP)
KOAKOAK	-14.0	-1.8	-1.8	697	1500-2000
NERLERK	-13.0	-1.8	-1.8	685	1500-2000
KOPANOAR	-13.0	-1.0	-1.0	490	1500-2000
UKALERK	-13.5	-1.8	-1.8	764	2000-3000

Table II : Summary Of The Transgression Times As Predicted By The Lachenbruch et al. '82 Model For The Industrial Wells

5.4 OFFSHORE TRANSECT

Figure 17 illustrates the temperature-depth profile predicted by Lachenbruch et al. '82 model for water depths from 1 m to 70 m corresponding to transgression times of 120 to 11020 years from the relative sea level curve (Fig. 4). The line through -12 °C is the temperature profile assumed in equilibrium with the pre-transgression, sub-aerial environment. The model assumes surface temperatures before transgression -12 °C, after transgression -1 °C, and the ice bonded permafrost base is at 670 m. Offshore transects with other surface temperatures before transgression are placed in Appendix IV.

This graph suggests the general nature of temperature-depth profiles that might be excepted at various present-day water depths offshore. Note that large negative temperature-depth gradients and temperature inversions should be observed in shallow water areas (confirmed at Angasak) while at water depths greater than 20 m, the model predicts nearly isothermal temperature profiles lying close to the freezing point (as observed at Amauligak).



OFFSHORE TRANSECT

Fitting Parameters

THO = -12.00
THS = -1.00
THB = -1.00

Input File : TRANSECT.DT2

Output File : TRANSECT.OT2

Water Depth [m]

1.0 8.0
2.0 10.0
3.0 12.0
4.0 15.0
5.0 20.0
6.0 70.0

Figure 17 : Offshore Transect : The Temperature-Depth Profile Predicted By Lachenbruch et al. Model For Water Depths From 1 to 70 m.

6.0 DISCUSSION

Times of transgression obtained from the relative sea level curve (solid circles Fig. 4) and predicted by the previous classical analytical modelling (open circles Fig. 4) are compared in Table III

WELL	WATER DEPTH (m BSL)	TRANSGRESSION Hill et al. (a BP)	TRANSGRESSION Classical Modelling (a BP)
ANGASAK	7	1200	400
UKALERK	30	5400	1500-2000
AMAULIGAK	32	5800	3000
KOAKOAK	47	8500	1500-2000
KOPANOAR	57	10400	1500-2000
NERLERK	58	10500	2000-3000

Table III : Comparison Of The Transgression Times As Predicted By The Classical Modelling And The Relative Sea Level Curve

From the above analysis the time to inundation appears to be very different between the Angasak and Amauligak wellsite. The shape of the Angasak temperature profile (Fig. 5) reflects a relatively early stage of permafrost degradation while the Amauligak temperature profile (Fig. 7) indicates an advanced stage of degradation.

The contrast between the geothermal predictions of inundation time and that from the relative sea level curve for the Angasak wellsite may be attributed to near-surface and latent heat effects since the upper boundary of the frozen section lies near 30 m. Melting the top 30 m of the ice-rich sediments would result in a slower evolution of the transient temperature profiles than predicted in Figure 9 due to the effects of latent heat and salinity. In the development of the analytical

models the assumption, that the transgression occurred in a short period at some time in the past, was made. Therefore, the time calculated by the model refers to the time of the main thermal event, the sudden increase in mean surface temperature, accompanying transgression. This time is probably later than the transgression event as observed from geomorphic evidence. For some time after the sea overran the site, the water was probably shallow and the sea ice could have been grounded for much of the winter. Thus lowering the mean surface temperatures to a value consistent with an sub-aerial environment. Therefore, only when water depths at the site exceed, perhaps 1 m would the inundated seabed experience higher mean annual surface temperatures. These effects are not considered in the classical models, therefore, the time of marine transgression predicted by these models is a minimum.

Better agreement is obtained at Amauligak since the temperature profile in the top 50 percent of the permafrost zone was recovered. This isothermal profile is very sensitive to the choice of seabed and permafrost base temperatures which are difficult to assess because they are assumed as a mean value for all of the submergence period. These parameters may not reflect the mean value if the sea was colder or warmer in the past.

Analysis of the industry data suggests the pronounced temperature inversions (Figs. 13-16) are more typical of shallower water depths (see Offshore Transect Fig. 17) than is the case. Therefore, unrealistically low times since transgression are required to explain these four temperature data sets. These profiles are less precise (+/- 2 K) and perhaps have residual thermal disturbance due to drilling of these petroleum explorations wells. Nixon [5] and Outcalt [6] have modelled them in terms of severe freezing point depression and unfrozen water content known to occur in fine grained soils such as marine clays. Very little physical property data (ie. thermal conductivity, salinity, moisture content, lithology) are available in the public domain which suggests that more sophisticated numerical modelling has to be undertaken with caution. But simple models can be applied to understand the regional conditions and to predict conditions where observations are not available.

Work was started on examining the detailed lithology of these wells, but is incomplete at this time. However, frequently sample recovery in the permafrost zone is incomplete; also geophysical well logs (ie. seismic, resistivity, gamma, neutron-density) are usually not run in the permafrost zone.

7.0 REFERENCES

1 Mackay, J.R.
1972: Offshore permafrost and ground ice, southern Beaufort Sea, Canada; Canadian Journal of Earth Sciences, v. 9, p. 1550-1561.

2 Judge, A.S.
1973: The prediction of permafrost thickness. Canadian Geotechnical Journal v. 10, p. 1-11.

3 Hayley, D.W.
1983: Geotechnical and engineering significance of subsea permafrost; in Permafrost Fourth International Conference Final Proceedings, p.93-95.

4 Jahns, H.O.
1983: Subsea permafrost and petroleum development; in Permafrost Fourth International Conference Final Proceedings, p.90-92.

5 Nixon, J.F.
1986: Thermal simulation of subsea saline permafrost; Canadian Journal of Earth Sciences, v. 23, p. 2039-2046.

6 Outcalt, S.
1985: A numerical model of subsea permafrost; in Freezing and thawing of soil-water systems, D.M. Anderson and P.J. Williams (ed); American Society of Civil Engineers, New York, p. 58-65.

7 Judge, A.S., Pelletier, B.R., and Norquay, I.
1987: Permafrost base and distribution of gas hydrates; in Marine Science Atlas of the Beaufort Sea: Geology and Geophysics, B.R. Pelletier (ed.). Geological Survey of Canada, Miscellaneous Report 40, p.39.

8 Weaver, J.S. and Stewart, J.M
1982: In-situ hydrates under the Beaufort Sea shelf; in Proceedings of the Fourth Canadian Permafrost Conference, H.M. French, (ed); National Research Council of Canada, p. 312-319.

9 Taylor, A.E. and Judge, A.S.
1988: Reconstruction of marine transgression history from an offshore ground temperature profile, Esso Angasak L-03 wellsite, Beaufort Sea; in Current Research, Part D, Geological Survey of Canada, Paper 88-1D, p.137-142.

10 Taylor, A.E. and Allen, V.S.
1989: Recovery of precise offshore permafrost temperatures from a deep geotechnical hole, Canadian Beaufort Sea; in Current Research, Part D, Geological Survey of Canada, Paper 88-1D, p.119-123.

11 Taylor, A.E. and Allen, V.S.
1986: Geothermal Program on the CCGS Nahidik cruises 1981-83, Canadian Beaufort Sea; internal report 86-05, Earth Physics Branch, EMR Canada.

12 Smith, P.A.
1986: The Late Pleistocene-Holocene stratigraphic records, Canning River Delta region, northern Alaska; Open file report 1237, Geological Survey of Canada, EMR Canada, p. 51-54

13 Brigham, J.K. and Miller, G.M.
1983: Paleotemperature estimates of the Alaskan Arctic coastal plain during the last 125,000 years; in Permafrost: Fourth International Conference, Proceeding, Fairbanks, Alaska, p. 80-85

14 Hill, P.R., Mudie, P.J., Moran, K. and Blasco, S.M.
1985: A sea-level curve for the Canadian Beaufort Shelf. Canadian Journal of Earth Sciences v. 22, p. 1383-1393.

15 Lachenbruch, A.H.
1957: Thermal effects of the ocean on permafrost. Bulletin fo the Geological Society of America v. 68, p. 1515-1530.

16 Lachenbruch, A.H., Sass, J.H., Marshall, B.V. and Moses, T.H., Jr.
1982: Permafrost, heat flow, and the geothermal regime at Prudhoe Bay, Alaska. Journal of Geophysical Research v. 87, p. 9301-9316.

17 Taylor, A.E, and Judge, A.S.
1977: Canadian geothermal data collection-northern wells 1976-1977; Geothermal Series of the Earth Physics Branch, EMR Canada, p.194

18 Majorowicz, J.A. and Dietrich, J.R.
1989: Comparison of the geothermal and organic maturation gradients of the central and southwestern Beaufort-MacKenzie Basin, Yukon and Northwest Territories; in Current Research, Part G, Geological Survey of Canada, Paper 89-1G, p.64

8.0 GLOSSARY

CAISSON-RETAINED ISLAND - a unique drilling platform that can be moved from wellsite to wellsite. It consists of a hollow steel ring or caisson unit, that is ballasted onto a beam on the seafloor and infilled with sand and a underwater sand platform or berm that extends outward from the ring.

MULTITHERMISTOR CABLE - a multiconductor electrical cable with thermistors as temperature sensors imbedded along its length. It is left in drill holes and temperatures can be acquired manually or with a data logger.

PERMAFROST - permanently frozen soil or subsoil, occurring at a variable depth beneath the earth's surface in which a temperature below freezing has existed for at least two consecutive years.

REGRESSION - retreat of the sea from land areas; also any change that converts offshore, deep-water conditions to nearshore, shallow-water conditions.

RELICT PERMAFROST - permafrost that was formed in the past and persists in places where it could not form today.

SUBMERGENCE - a rise of the water level in relation to the land, so that areas formerly dry land become inundated; it results either from a sinking of the land or from a net rise of the water level.

TRANSGRESSION - the spread of the sea over land areas; also any change that brings offshore, deep-water environments to areas formerly occupied by nearshore, shallow-water conditions.

APPENDIX I

SOURCE CODE FOR PROGRAM GRAD

```
$STORAGE:2
$DEBUG
************************************************************************
*
      PROGRAM GRAD
*
*  Calculation of transient gradient anomaly due to regression of the
*  ocean, as funtion of time for any depth.
*  After eqn. 14, Lachenbruch 1957.
*  Bull. GSA 68,1515-1530.
*
*  COMPILE SEQUENCE: "FL /c GRAD.FOR"
*          Microsoft (R) FORTRAN Optimizing Compiler Version 5.00
*          Copyright (c) Microsoft Corp 1982-1989. All rights reserved.
*          GRAD.FOR
*  LINK SEQUENCE: "LINK"
*          Microsoft (R) Segmented-Executable Linker  Version 5.03
*          Copyright (C) Microsoft Corp 1984-1989.  All rights reserved.
*
*          Object Modules [.OBJ]: GRAD.OBJ/SE:1024
*          Run File [GRAD.EXE]: GRAD.EXE
*          List File [NUL.MAP]: NUL.MAP
*          Libraries [.LIB]: PLOT88+LLIBFOR7+SUBFUN
*          Definitions File [NUL.DEF]: NUL.DEF
*
* PARAMETERS : NONE
C VARIABLES :
*   CHARACTER
*     - FOUT : Output file name.
*     - FILEIN : Input file name.
*     - DISKFL : Plot file name.
*     - TITLE : Title of graph to be plotted.
*     - TITBLK : Title of graph padded with leading blanks.
*     - CTEXT : Character string that contains the number of years
*               since inundation.
*     - AGAIN : If "yes" then run program again with different fitting
*               parameters.
*     - CHOICE : If "yes" then user wishes to plot measured gradient
*                anomaly.
*   INTEGER
*     - I, J : Array index.
*     - IOPORT : Input/Output port.
*     - MODEL : Output device identification.
*     - LENTIT : Length of character string "TITLE".
*     - LENBK : Number of blanks to pad.
*     - NUMZ : Number of depth measurements.
*     - NUMKFR : Number of frozen thermal conductivity measurements.
*     - NUMTIME : Number of inundation times.
*     - NUMPT : Number of data points.
*     - POINTS : Contains the first array index for each graph.
*     - NC : Number of characters returned in CTEXT.
*     - NUMEAS : Number of measured gradient anomalys and times.
*   REAL
```

```
*      - A : Temperature change upon shoreline movement.
*      - X : Distance from present shoreline.
*      - TIMES : Times of inundation.
*      - LAMDA : Thermal time constant [s].
*      - KFR(20) : Frozen thermal conductivity [W/(m*C)].
*      - KFRAVE : Average frozen thermal conductivity [W/(m*C)].
*      - Z(30) : Depth below seabed [m].
*      - T(20) : Temperature [C].
*      - ZKFR(20) : Conductivity depth [m].
*      - RHO(20) :  Density [Kg/m^3].
*      - Q : Heat flow [Wm^-2].
*      - ALPHA : Thermal diffusivity [m^2/s].
*      - FACT : Ratio of the desired drawing size to the normal drawing
*               size
*      - C : Specific Heat [J/(C*Kg)].
*      - XTIME : array of (alpha*time) used for plotting.
*      - YGRADT : array of anomaly gradient data used for plotting.
*      - XTEXT, YTEXT : Location of CTEXT
*      - GRADTM : measured gradient anomaly.
*      - GRADT : calculated gradient anomaly.
*      - GRADM : gradient of best fit line between three points.
*      - GRADEQM : equilibrium gradient.
*      - HEIGHT : height of symbol.
*      - MEASGR : measured gradient anomaly, used for plotting (REAl).
*      - MEASTM : calculated time for MEASGR (Real).
*    LOGICAL
*      - OVERFL : Flag, if true then overwrites an existing file.
*      - SKIP : Flag, if true then skips the plotting routine.
*      - MEAS : Flag , true if measured data is to be plotted.
*-----------------------------------------------------------------------
*
* these external functions and subroutines are found in library
* SUBFUN.LIB ....
      EXTERNAL GETFID,CRTFID,DEVICE,LENTRM
* Declare variables ...
      CHARACTER FOUT*12,TITLE*25,TITBLK*25,CHOICE*1,
     +          CTEXT*10,FILEIN*12,AGAIN*1,DISKFL*12
      INTEGER LENTIT,IOPORT,MODEL,NUMZ,NUMKFR,NUMTIME,NUMPT,LENBK,
     +        POINTS,NC,I,J,NUMEAS
      REAL A,KFR,Z,T,FACT,ZKFR,RHO,Q,C,ALPHA,KFRAVE,RHOAVE,X,
     +     TIMES,XTIME,YGRADT,XTEXT,YTEXT,GRADTM,GRADT,HEIGHT,GRADM,
     +     GRADEQM,MEASGR,MEASTM
      LOGICAL OVERFL,SKIP,MEAS
      DIMENSION T(30),KFR(20),ZKFR(20),RHO(20),TIMES(20),GRADTM(30),
     +          XTIME(500),YGRADT(500),POINTS(30),GRADT(20,30),Z(30),
     +          MEASGR(50),MEASTM(50)
* Initialize variables ...
      NUMZ = 30
      NUMKFR = 20
C Read measured data ...
      CALL ZTMEAS(FILEIN,Z,T,ZKFR,KFR,RHO,Q,C,NUMZ,NUMKFR)
      PRINT*
C Prompt user for output file name ...
      OVERFL = .FALSE.
```

```
      CALL CRTFID('Enter Output File Name -----> ',FOUT,OVERFL)
      IF (OVERFL) THEN
         OPEN(1,FILE=FOUT,STATUS='OLD')
         CLOSE(1,STATUS='DELETE')
      ENDIF
      OPEN(1,FILE=FOUT,STATUS='NEW')
      PRINT*
C Prompt user for device parameters for 'PLOTS' statement ...
      CALL DEVICE(IOPORT,MODEL,FACT,DISKFL)
* determine whether to skip the plotting routine ...
      IF (IOPORT .EQ. 0 .AND. MODEL .EQ. 0 .AND. FACT .EQ. 0.0) THEN
         SKIP = .TRUE.
      ELSE
         SKIP = .FALSE.
      ENDIF
      IF (.NOT. SKIP) THEN
* Prompt for title for graph ...
         PRINT*
         WRITE(*,'(A\)') ' Enter Graph Title (max. 25 characters)'
         WRITE(*,'(A\)') ' -----> '
         READ(*,'(BN,A25)')TITLE
         PRINT*
* Determine length of title ...
         LENTIT = LENTRM(TITLE)
* pad title with leading blanks for centering in subroutine GRAPH
         TITBLK = '                                        '
         IF (LENTIT .LT. LEN(TITLE)) THEN
            LENBK = (NINT(LEN(TITLE)-LENTIT)/2.)
            IF (LENBK .EQ. 0) THEN
                LENBK = 1
            ENDIF
            TITBLK = TITBLK(1:LENBK) // TITLE
            LENTIT = LENTIT + LENBK
         ELSE
            TITBLK = TITLE
         ENDIF
* determine if user wishes to plot measured gradient anomaly ...
 170     CHOICE = ' '
         WRITE(*,'(1X,A\)') 'Plot Measured Gradient Anomaly {Y/N} '
         WRITE(*,'(A\)') '-----> '
         READ(*,'(A)',ERR=170) CHOICE
         IF (CHOICE .EQ. 'Y' .OR. CHOICE .EQ. 'y') THEN
            MEAS = .TRUE.
         ELSE
            MEAS = .FALSE.
         ENDIF
         PRINT*
      ENDIF
C Calculate derived physical properties ...
      CALL DERPHY(KFR,RHO,C,NUMKFR,KFRAVE,RHOAVE,ALPHA)
C Calculate gradient anomaly from measured data ...
      CALL GRADNT(Z,T,Q,KFRAVE,NUMZ,GRADTM,GRADM,GRADEQM)
C Read fitting parameters ...
 500  CALL FITPAR(A,X,TIMES,NUMTIME)
```

```
C Calculate gradient anomaly ...
      PRINT*
      WRITE(*,'(A\)') ' Calculating Gradient Anomaly  '
      CALL EQU14(Z,A,X,ALPHA,TIMES,NUMZ,NUMTIME,GRADT)
C Find times for calculated gradient anomaly from measured ...
      IF (MEAS) THEN
         PRINT*
         WRITE(*,'(A\)') ' Calculating Times For Measured Gradient'
         WRITE(*,'(A\)') ' Anomaly  '
         CALL FNDTMS(GRADTM,GRADT,Z,TIMES,A,X,ALPHA,NUMTIME,NUMZ,
     +                MEASGR,MEASTM,NUMEAS)
         NUMPT = NUMTIME*NUMZ + NUMEAS
      ELSE
         NUMPT = NUMTIME*NUMZ
      ENDIF
C Write to output file the above data that was read in ...
      PRINT*
      WRITE(*,'(A\)') ' Generating A Report File  '
      CALL REPORT(FOUT,Z,T,ZKFR,KFR,RHO,Q,C,KFRAVE,RHOAVE,ALPHA,A,X,
     +            GRADTM,TIMES,GRADT,NUMZ,NUMKFR,NUMTIME,GRADM,GRADEQM,
     +            MEAS,MEASGR,MEASTM,NUMEAS)
      CLOSE(1)
* If user wishes to plot ...
      IF (.NOT. SKIP) THEN
C Form arrays X, Y for plotting ...
         PRINT*
         WRITE(*,'(A\)') ' Forming Arrays For Plotting  '
         CALL FARRAY(TIMES,ALPHA,GRADT,XTIME,YGRADT,NUMTIME,NUMZ,
     +               NUMPT,POINTS,MEAS,MEASGR,MEASTM,NUMEAS)
C Plot graph ...
         PRINT*
         WRITE(*,'(A\)') ' Generating Parameters For PLOT88  '
         CALL GRAPH(XTIME,YGRADT,POINTS,IOPORT,MODEL,FACT,TITBLK,LENTIT,
     +              NUMPT,DISKFL,MEAS)
* add labels to the graph ...
* first set the width to height ratio of characters (condensed) ...
         CALL ASPECT(0.50)
* define the COMPLX symbol font ...
         CALL COMPLX
* add fitting parameters ...
         CALL SYMBOL(12.,7.8,.2,'Fitting Parameters',0.,18)
         CALL SYMBOL(12.3,7.5,.2,'A = ',0.,4)
         CALL SYMBOL(12.3,7.2,.2,'X = ',0.,4)
         WRITE(*,'(A\)') '.'
         CALL NUMBER(12.7,7.5,.2,A,0.,2)
         CALL NUMBER(12.7,7.2,.2,X,0.,2)
* add input and output file names ....
         CALL SYMBOL(12.,6.6,.2,'Input File :',0.,12)
         CALL SYMBOL(13.1,6.6,.2,FILEIN,0.,12)
         CALL SYMBOL(12.,6.1,.2,'Output File :',0.,13)
         CALL SYMBOL(13.2,6.1,.2,FOUT,0.,12)
C add legend ...
         CALL SYMBOL(12.4,5.,.2,'Depth [m]',0.,9)
         WRITE(*,'(A\)') '.'
```

```
            XTEXT = 12.
            YTEXT = 4.5
            J = 0
            DO 200 I=1, NUMZ
               CALL SYMBOL(XTEXT,YTEXT,.2,CHAR(J),0.0,-1)
               CALL GETNUM(Z(I),1,CTEXT,NC)
               CALL SYMBOL(XTEXT+.2,YTEXT-.05,.2,CTEXT,0.0,NC)
               YTEXT = YTEXT -.30
               IF (I .EQ. ((NUMZ+1)/2)) THEN
                  YTEXT = 4.5
                  XTEXT = XTEXT + 1.25
               ENDIF
               J = J + 1
               IF (J .EQ. 6) J = 10
               IF (J .EQ. 12) J = 14
               IF (J .EQ. 16) J = 0
 200        CONTINUE
            IF (MEAS) THEN
               CALL SYMBOL(XTEXT+.2,YTEXT-.05,.2,'Measured Data',0.,13)
               DO 220 HEIGHT=.15, .01, -.02
                  CALL SYMBOL(XTEXT,YTEXT,HEIGHT,CHAR(1),0.0,-1)
 220           CONTINUE
            ENDIF
            WRITE(*,'(A\)') '.'
            PRINT*
            PRINT*, 'Drawing Plot ...'
            CALL PLOT(0.,0.,999)
         ENDIF
         PRINT*
C Determine whether to run program again
 120     AGAIN = ' '
         WRITE(*,'(1X,A\)') 'Another Run With Different Fitting Parameter'
         WRITE(*,'(A\)') ' {Y/N} -----> '
         READ(*,'(A)',ERR=120) AGAIN
         IF (AGAIN .EQ. 'Y' .OR. AGAIN .EQ. 'y') THEN
* Prompt user for output file name ...
            OVERFL = .FALSE.
            CALL CRTFID('Enter Output File Name -----> ',FOUT,OVERFL)
            IF (OVERFL) THEN
               OPEN(1,FILE=FOUT,STATUS='OLD')
               CLOSE(1,STATUS='DELETE')
            ENDIF
            OPEN(1,FILE=FOUT,STATUS='NEW')
            PRINT*
            CALL DEVICE(IOPORT,MODEL,FACT,DISKFL)
            PRINT*
            GOTO 500
         ENDIF
         STOP
 1000 END

*
**********************************************************************
*
```

```
C SUBROUTINE DERPHY  1989-06-30-PPS
* Module to calculate derived physical properties : average thermal
* conductivity, average density, and thermal diffusivity.
*
* Parameters :
*    KFR - frozen conductivity [W/(m*C)] (Real).
*    RHO - density [Kg/m^3] (Real).
*    C - Specific Heat (Real).
*    NUMKFR - number of frozen conductivity measurements (Integer).
*    KFRAVE - average frozen thermal conductivity [W/(m*C)(Real).
*    RHOAVE - average density [Kg/m^3] (Real).
*    ALPHA -  thermal diffusivity [m^2/s] (Real).
*
* Variables
*    I - index (Integer).
*
*---------------------------------------------------------------------
*
      SUBROUTINE DERPHY(KFR,RHO,C,NUMKFR,KFRAVE,RHOAVE,ALPHA)
*
* declare variables ...
      REAL KFR,RHO,C,KFRAVE,RHOAVE,ALPHA
      INTEGER I,NUMKFR
      DIMENSION KFR(NUMKFR),RHO(NUMKFR)
* calculate the average frozen thermal conductivity and the average
* density...
      DO 100 I=1, NUMKFR
         KFRAVE = KFRAVE + KFR(I)
         RHOAVE = RHOAVE + RHO(I)
 100  CONTINUE
      KFRAVE = KFRAVE/NUMKFR
      RHOAVE = RHOAVE/NUMKFR
* calculate the thermal diffusivity ...
      ALPHA = KFRAVE / (C*RHOAVE)
      RETURN
      END
*
********************** {END OF SUBROUTINE: DERPHY} **********************
*
C FUNCTION ERFS  1989-08-03-PPS
* Module to calculate the error function.
*
* Parameters :
*    X - value at which the error function is evaluated (Real).
* Variables :
*    A1..A6 - Coefficients of the error function (Real).
*    S - sum of terms (Real).
*---------------------------------------------------------------------
*
      FUNCTION ERFS(X)
*
* declare variables ...
      REAL X,A1,A2,A3,A4,A5,A6,S
      A1=0.0705230784
```

```
      A2=0.0422820123
      A3=0.0092705272
      A4=0.0001520143
      A5=0.0002765672
      A6=0.0000430638
      S=1.0+A1*X
      S=S+A2*X*X
      S=S+A3*X*X*X
      S=S+A4*X*X*X*X
      S=S+A5*X*X*X*X*X
      S=S+A6*X*X*X*X*X*X
      ERFS=1.0-1.0/(S**16)
      RETURN
      END
*
************************* {END OF FUNCTION : ERFS} ************************
*
C SUBROUTINE EQU14  1989-08-03-PPS
* Module to approximate the gradient anomaly (After equation 14,
* Lachenbruch, 1957)
*
* Parameters :
*    Z - depth  (Real).
*    A - Temperature change upon shoreline movement (Real).
*    X - Distance from present shoreline (Real).
*    ALPHA - thermal diffusivity (Real).
*    TIMES - times of inundation (Real).
*    NUMZ - number of depths (Integer).
*    NUMTIME - number of times (Integer).
*    GRADT - gradient anomaly due to the ocean, given by Equ. 14 (Real).
* Variables
*    STIME - time in seconds instead of years (Real).
*    PI - 3.14159 (Real).
*    I,J - array indexes (Integer).
*    TERM1,TERM2,TERM3,TERM4,TERM5 -  terms in equation 14 (Real).
*-----------------------------------------------------------------------
*
      SUBROUTINE EQU14(Z,A,X,ALPHA,TIMES,NUMZ,NUMTIME,GRADT)
*
* declare variables ...
      REAL PI,Z,A,X,ALPHA,TIMES,GRADT,ERFS,STIME,TERM1,TERM2,TERM3,
     +     TERM4,TERM5
      INTEGER I,J,NUMZ,NUMTIME
      DIMENSION GRADT(NUMTIME,NUMZ),Z(NUMZ),TIMES(NUMTIME)
      PI = 4*ATAN(1.0)

* Range through all times ...
      DO 100 I=1, NUMTIME
         WRITE(*,'(A\)') '.'
         STIME = TIMES(I)*31536000.0*ALPHA
* Range through all depths ...
         DO 200 J=1, NUMZ
* calculate the terms in equation 14 ...
            TERM1 = X*EXP(-(Z(J)**2 + X**2)/(4.*STIME))
```

```
            TERM2 = (PI*(Z(J)**2 + X**2))
            TERM3 = EXP(-(Z(J)**2/(4.*STIME)))
            TERM4 = 2*(PI*STIME)**.5
            TERM5 = 1.+ERFS(.5*X/(STIME**.5))
            GRADT(I,J) = -A*(TERM1/TERM2+TERM3/TERM4*TERM5)*1000
 200      CONTINUE
 100   CONTINUE
       PRINT*
       RETURN
       END
*
********************** {END OF SUBROUTINE: EQU14} **********************
*
C SUBROUTINE FARRAY  1989-08-14-PPS
* Module to form arrays for plotting
*
* Parameters :
*     TIMES - indunation times (Real).
*     ALPHA - thermal diffusivity (Real).
*     GRADT - gradient anomaly (Real).
*     X - (alpha*time) (Real).
*     Y - gradient anomaly data (Real).
*     NUMTIME - number of times (Integer).
*     NUMZ - number of depths (Integer).
*     NUMPT - number of data points (Integer).
*     POINTS - contains the first array index of each graph (Integer).
*     MEAS - flag , true if measured data is to be plotted (Logical).
*     MEASGR - measured gradient anomaly, used for plotting (REAl).
*     MEASTM - calculated time for MEASGR (Real).
*     NUMEAS - number of measured gradients and times (Integer).
* Variables
*     POS - position in the array POINTS (Real).
*     I,J,K - array indexes (Integer).
*     NUMDM - number of measure data (Integer).
*-----------------------------------------------------------------------
*
       SUBROUTINE FARRAY(TIMES,ALPHA,GRADT,X,Y,NUMTIME,NUMZ,NUMPT,
      +                  POINTS,MEAS,MEASGR,MEASTM,NUMEAS)
* declare variables ...
       REAL TIMES,GRADT,X,Y,ALPHA,POS,MEASGR,MEASTM
       INTEGER I,J,K,NUMTIME,NUMPT,POINTS,NUMZ,NUMDM,NUMEAS
       LOGICAL MEAS
       DIMENSION GRADT(NUMTIME,NUMZ),X(NUMPT),Y(NUMPT),MEASTM(NUMEAS),
      +          POINTS(30),TIMES(NUMTIME),MEASGR(NUMEAS)

       POS = 1
       NUMDM = 0
       IF (MEAS) THEN
* Add the measured data to arrays X, Y ...
          DO 200 K=1, NUMEAS
             X(K) = ALPHA*MEASTM(K)*31536000.0
             Y(K) = MEASGR(K)
 200      CONTINUE
          POINTS(POS) = 1
```

```
            POINTS(POS+1) = POINTS(POS) + NUMEAS
            POS = POS + 1
            NUMDM = NUMEAS
         ENDIF
*  Form arrays X,Y = TIMES,GRADT(Z,T) for plotting ...
         DO 100 I=1,NUMZ
            DO 101 J=1,NUMTIME
               K = I*NUMTIME-NUMTIME+J + NUMDM
               X(K) = (ALPHA*TIMES(J)*31536000.0)
               Y(K) = GRADT(J,I)
101         CONTINUE
            POINTS(POS) = I*NUMTIME - NUMTIME + 1 + NUMDM
            POS = POS + 1
            WRITE(*,'(A\)') '.'
100      CONTINUE
* add the ending value ...
         POINTS(POS) = POINTS(POS-1) + NUMTIME
         PRINT*
         RETURN
         END
*
********************** {END OF SUBROUTINE: FARRAY} **********************
*
C SUBROUTINE FITPAR  1989-07-20-PPS
* Module to read the fitting paramaters from the keyboard.
* Parameters :
*     A - temperature change upon shoreline movement (Real).
*     X - distance from present shoreline (Real).
*     TIMES - times of inundation (Real).
*     NUMTIME - number of inundation times (Integer).
* Variables :
*     RATIO - ratio of final and initial time used as increment (Real)
*     I - array index (Integer).
*-------------------------------------------------------------------------
*
         SUBROUTINE FITPAR(A,X,TIMES,NUMTIME)
*
* Declare variables ...
         REAL A,X,TIMES,RATIO
         INTEGER I,NUMTIME
         DIMENSION TIMES(20)
* Prompt user to enter the fitting parameters ...
         WRITE(*,*) 'ENTER FITTING PARAMATERS'
 100     WRITE(*,'(A\)') '    Temperature Change Upon Shoreline Movement '
         WRITE(*,'(A\)') '[K] -----> '
         READ(*,*,ERR=100) A
 110     WRITE(*,'(A\)') '    Distance From Present Shoreline [m] -------'
         WRITE(*,'(A\)') '----------> '
         READ(*,*,ERR=110) X
 112     WRITE(*,'(A\)') '    Number of Time Intervals ------------ -----'
         WRITE(*,'(A\)') '----------> '
         READ(*,*,ERR=112) NUMTIME
 120     WRITE(*,'(A\)') '    Inundation Time (Lower Range) [years] -----'
         WRITE(*,'(A\)') '----------> '
```

```
      READ(*,*,ERR=120) TIMES(1)
 130  WRITE(*,'(A\)') '    Inundation Time (Upper Range) [years]------'
      WRITE(*,'(A\)') '---------> '
      READ(*,*,ERR=120) TIMES(NUMTIME)
* Form time array ...
      RATIO = (TIMES(NUMTIME)/TIMES(1))**(1./(NUMTIME-1))
      DO 200 I=2, NUMTIME-1
         TIMES(I) = TIMES(1)*RATIO**(I-1)
 200  CONTINUE
      RETURN
      END
*
********************** {END OF SUBROUTINE: FITPAR} ***********************
*
C SUBROUTINE FNDTMS  1989-08-21-PPS
* Module to read the fitting paramaters from the keyboard.
* Parameters :
*    GRADTM - measured gradient anomaly (Real).
*    GRADT - calculated gradient anomaly (Real).
*    Z - depth [m] (Real).
*    TIMES - desired search times [years] (Real).
*    A - Temperature change upon shoreline movement (Real).
*    X - Distance from present shoreline (Real).
*    ALPHA - thermal diffusivity (Real).
*    NUMTIME - Number of desired search times (Integer).
*    NUMZ - Number of depth measurements (Integer).
*    MEASGR - measured gradient anomaly, used for plotting (REAl).
*    MEASTM - calculated time for MEASGR (Real).
*    NUMEAS - number of measured gradients and times (Integer).
* Variables :
*    GRDNT1,GRDNT2 - gradient interval (Real).
*    TIME1, TIME2 - time interval (Real).
*    TARGET - point inside gradient interval (Real).
*    TRTIME - time corresponding to target (Real).
*    STIME - time in seconds instead of years (Real).
*    PI - 3.14159 (Real).
*    I,J - array indexes (Integer).
*    TERM1,TERM2,TERM3,TERM4,TERM5 -  terms in equation 14 (Real).
*    GRDNT - gradient anomaly (Real).
*    FOUND - flag , if true then an interval was found (Logical).
*    I,J,K - array indexes (Integer).
*-----------------------------------------------------------------------
*
      SUBROUTINE FNDTMS(GRADTM,GRADT,Z,TIMES,A,X,ALPHA,NUMTIME,NUMZ,
     +                  MEASGR,MEASTM,NUMEAS)
*
* Declare variables ...
      REAL GRADTM,GRADT,Z,TIMES,MEASGR,TIME1,TIME2,TARGET,PI,A,X,
     +     ALPHA,ERFS,STIME,TERM1,TERM2,TERM3,TERM4,TERM5,GRDNT,TRTIME,
     +     GRDNT1,GRDNT2,MEASTM
      INTEGER I,J,K,NUMZ,NUMTIME,NUMEAS
      LOGICAL FOUND
      DIMENSION TIMES(NUMTIME),GRADTM(NUMZ-2),GRADT(NUMTIME,NUMZ),
     +          Z(NUMZ),MEASGR(50),MEASTM(50)
```

```
      PI = 4*ATAN(1.0)
* search for time interval ...
      K = 1
      DO 100 I=2, NUMZ-1
         FOUND = .FALSE.
         TARGET = ABS(GRADTM(I-1))
         DO 200 J=1, NUMTIME-1
            GRDNT1=ABS(GRADT(J,I))
            GRDNT2=ABS(GRADT(J+1,I))
* find the time interval ...
            IF ((GRDNT1 .GE. TARGET .AND. TARGET .GE. GRDNT2) .OR.
     +          (GRDNT1 .LE. TARGET .AND. TARGET .LE. GRDNT2)) THEN
               FOUND = .TRUE.
               TIME1 = TIMES(J)
               TIME2 = TIMES(J+1)
               DO WHILE (ABS(GRDNT1-GRDNT2) .GE. 0.0001)
                  TRTIME = (TIME1+TIME2)/2.
                  STIME = TRTIME*31536000.0*ALPHA
* get precise time in the interval by calculating the terms in equ.14...
                  TERM1 = X*EXP(-(Z(I)**2 + X**2)/(4.*STIME))
                  TERM2 = (PI*(Z(I)**2 + X**2))
                  TERM3 = EXP(-(Z(I)**2/(4.*STIME)))
                  TERM4 = 2*(PI*STIME)**.5
                  TERM5 = 1.+ERFS(.5*X/(STIME**.5))
                  GRDNT=ABS(-A*(TERM1/TERM2+TERM3/TERM4*TERM5)*1000.)
                  IF (GRDNT1 .GE. TARGET .AND. TARGET .GE. GRDNT2) THEN
                     IF (TARGET .GE. GRDNT) THEN
                        TIME2 = TRTIME
                        GRDNT2 = GRDNT
                     ELSE
                        TIME1 = TRTIME
                        GRDNT1 = GRDNT
                     ENDIF
                  ELSE
                     IF (TARGET .LE. GRDNT) THEN
                        TIME2 = TRTIME
                        GRDNT2 = GRDNT
                     ELSE
                        TIME1 = TRTIME
                        GRDNT1 = GRDNT
                     ENDIF
                  ENDIF
               ENDDO
* If an interval was found then store the gradient anomaly and time ...
               IF (FOUND) THEN
                  MEASGR(K) = GRADTM(I-1)
                  MEASTM(K) = TRTIME
                  K = K + 1
                  WRITE(*,'(A\)') '.'
               ENDIF
            ENDIF
 200     CONTINUE
 100  CONTINUE
      NUMEAS = K - 1
```

```
      PRINT*
      RETURN
      END
*
********************* {END OF SUBROUTINE: FNDTMS} ***********************
*
C SUBROUTINE GRADNT  1989-08-03-PPS
* Routine to calculate the gradient from measured data taking three
* points at a time.
*
* Parameters :
*    Z - depth [m] (Real).
*    T - temperature [C] (Real).
*    Q - heat flow [Wm^-2] (Real).
*    KFRAVE - average thermal conductivity [W/(m*C]) (Real).
*    NUMZ - number of depths (Integer).
*    GRADTM - measured gradient anomaly (Real).
*    GRADM - gradient of best fit line between three points (Real).
*    GRADEQM - equilibrium gradient (Real).
* Variables :
*    I - array indexes (Integer).
*    X, Y - set of points to run a best fit line through (Real).
*    INTERCT - dummy variable used to call LTSQ (Real).
*-----------------------------------------------------------------------
*
      SUBROUTINE GRADNT(Z,T,Q,KFRAVE,NUMZ,GRADTM,GRADM,GRADEQM)
*
* declare variables ...
      REAL Z,T,Q,KFRAVE,GRADM,GRADEQM,GRADTM,X,Y,INTERCT
      INTEGER NUMZ,I
      DIMENSION X(3),Y(3),Z(NUMZ),T(NUMZ),GRADTM(NUMZ-2)
* calculate equilibrium gradient ...
      GRADEQM = Q/KFRAVE*1000.
* gradient of best fit line between three points ...
      DO 10 I=2, NUMZ-1
         X(1) = Z(I-1)
         X(2) = Z(I)
         X(3) = Z(I+1)
         Y(1) = T(I-1)
         Y(2) = T(I)
         Y(3) = T(I+1)
         CALL LTSQ(X,Y,3,GRADM,INTERCT)
         GRADM = GRADM*1000.
* calculate measured gradient anomaly ...
         GRADTM(I-1) = GRADM - GRADEQM
 10   CONTINUE
      RETURN
      END
*
********************* {END OF SUBROUTINE: GRADNT} ***********************
*
C SUBROUTINE GRAPH  1989-07-13-PPS
* Generic plotting routine for X, Y data arrays. If measured data is to
* be plotted it should stored at the beginning of the arrays.
```

```
*
* Parameters :
*    X - array of X elements (Real).
*    Y - array of Y elements (Real).
*    POINTS - contains the first array index of each graph that is to
*             be plotted (Integer).
*    PORT - Input/Output port (Integer).
*    MODEL - Output device identification (Integer).
*    FACT - Ratio of the desired drawing size to the normal drawing
*           size (Real).
*    TITLE - Title of graph to be plotted (Char).
*    LENTIT - Length of character string "TITLE" (Integer).
*    NUMPT - dimension of the data arrays (Integer).
*    DISFL - name of file to store the graph on disk (Char).
*    MEAS - flag , true if measured data is to be plotted (Logical).
* Variables :
*    XLABEL - Title label for the x-axis (Char).
*    MEASD - Flag, if true measured data is plotted (Logical).
*    YLABEL - Title label for the y-axis (Char).
*    XO - temporary array of X data (Real).
*    YO - temporary array of Y data (Real).
*    I, J, K - array indexes (Integer).
*    INTEQ - the index of the centered symbol to be drawn at each pt (Int)
*    HEIGHT - height of centered symbol (Real).
*-----------------------------------------------------------------------
*
      SUBROUTINE GRAPH(X,Y,POINTS,PORT,MODEL,FACT,TITLE,LENTIT,NUMPT,
     +                 DISKFL,MEAS)

* declare variables ...
      REAL X,Y,FACT,XO,YO,HEIGHT
      INTEGER PORT,MODEL,LENTIT,NUMPT,POINTS,I,J,K,INTEQ
      CHARACTER*(*) TITLE,DISKFL
      CHARACTER*30 XLABEL,YLABEL
      LOGICAL MEAS,MEASD
      DIMENSION X(NUMPT+2),Y(NUMPT+2),XO(50),YO(50),POINTS(30)
* initialize the PLOT88 Software ...
      CALL PLOTS(0,PORT,MODEL)
      MEASD = MEAS
* determine whether to write to disk ...
      IF (DISKFL .NE. '~~~~~~~~.~~~') THEN
         OPEN(90,FILE=DISKFL,STATUS='NEW',FORM='BINARY')
      ENDIF
* enlarge or reduce the size of the entire drawing ...
      CALL FACTOR(FACT)
* define a new origin ...
      CALL PLOT(2.0,3.0,-3)
      WRITE(*,'(A\)') '.'
* define the TRIPLX symbol font to be used by SYMBOL ...
      CALL TRIPLX
* add title ...
      CALL SYMBOL(0.0,10.,.4,TITLE,0.,LENTIT)
* scale data ...
      CALL SCALG(X,10.,NUMPT,1)
```

```
      CALL SCALE(Y,10.,NUMPT,1)
      WRITE(*,'(A\)') '.'
* define the COMPLEX symbol font to be used by SYMBOL ...
      CALL COMPLX
* set the characteristics of the X-axis annotation ...
      CALL STAXIS(.222,.3,.111,.111,1)
* drawn the X-axis ...
      XLABEL = '  t (m )'
      CALL LGAXS(0.,0.,XLABEL,-8,10.,0.,X(NUMPT+1),X(NUMPT+2))
      CALL SYMBOL(5.7,-.83,.18,'2',0.,1)
      CALL GRKPLX
      CALL SYMBOL(4.0,-.95,.3,CHAR(97),0.,1)
      CALL COMPLX
      WRITE(*,'(A\)') '.'
* set Y-axis annotation ...
      CALL STAXIS(.222,.3,.111,.111,-1)
* draw Y-axis ...
      YLABEL = 'Gradient Anomaly (mK/m)'
      CALL AXIS(0.,0.,YLABEL,23,10.,90.,Y(NUMPT+1),
     +          Y(NUMPT+2))
      WRITE(*,'(A\)') '.'
* set the line type and centered symbol characteristics ...
      CALL STLINE(1,.1,0.0)
* form arrays XO, YO to plot each graph with a different symbol ...
      J = 1
      INTEQ = 0
 200  K = 1
      DO 100 I=POINTS(J), POINTS(J+1)-1
         XO(K) = X(I)
         YO(K) = Y(I)
         K = K + 1
 100  CONTINUE
      XO(K) = X(NUMPT+1)
      XO(K+1) = X(NUMPT+2)
      YO(K) = Y(NUMPT+1)
      YO(K+1) = Y(NUMPT+2)
* draw the centered symbols ...
      IF (MEASD) THEN
         DO 111 HEIGHT=.1, .01, -.02
            CALL STLINE(1,HEIGHT,0.)
            CALL LGLIN(XO,YO,K-1,1,-1,1,-1)
 111     CONTINUE
         CALL STLINE(1,.1,0.0)
         MEASD = .FALSE.
      ELSE
         CALL LGLIN(XO,YO,K-1,1,1,INTEQ,-1)
         INTEQ = INTEQ +  1
         IF (INTEQ .EQ. 6) INTEQ = 10
         IF (INTEQ .EQ. 12) INTEQ = 14
         IF (INTEQ .EQ. 16) INTEQ = 0
      ENDIF
      J = J + 1
      IF (POINTS(J+1) .NE. 0) THEN
         GOTO 200
```

```
      ENDIF
      WRITE(*,'(A\)') '.'
      RETURN
      END
*
********************** {END OF SUBROUTINE: GRAPH} ***********************
*
C SUBROUTINE LTSQ   1989-07-21-PPS
* Module to calculate the least squares fit of a straight line. Returns
* the slope A and intercept B of Y = AX+B
*
* Parameters :
*    X - x coordinate (Real).
*    Y - y coordinate (Real).
*    NUMXY - number of X, Y pairs (Integer).
*    A - slope of staight line (Real).
*    B - intercept of straight line (Real).
* Variables :
*    SMX - sum of x ' s (Real).
*    SMY - sum of y ' s (Real).
*    SMXX - sum of x*x ' s (Real).
*    SMXY - sum of x*y ' s (Real).
*    SSX -  sum of squares of x' s (Real).
*    I - array index (Integer).
*-------------------------------------------------------------------------
*
      SUBROUTINE LTSQ(X,Y,NUMXY,A,B)
*
* declare and initialize variables ...
      REAL X,Y,A,B,SMX,SMY,SMXX,SMXY,SSX
      INTEGER NUMXY,I
      DIMENSION  X(200),Y(200)
      SMX = 0.0
      SMY = 0.0
      SMXX = 0.0
      SMXY = 0.0
      SSX = 0.0
* Find the sums ...
      DO 200 I=1, NUMXY
         SMX = SMX + X(I)
         SMY = SMY + Y(I)
         SMXX = SMXX + (X(I)*X(I))
         SMXY = SMXY + (X(I)*Y(I))
 200  CONTINUE
* Calculate the sum of the squares of X, slope, and the intercept ...
      SSX = (NUMXY*SMXX) - (SMX*SMX)
      A = ((NUMXY*SMXY) - (SMX*SMY)) /SSX
      B= (SMY - (A*SMX)) /NUMXY
      RETURN
      END
*
********************** {END OF SUBROUTINE: LTSQ} ************************
*
C SUBROUTINE REPORT  1989-07-06-PPS
```

```
* Module to print measured data : depth, temperature, conductivity depth,
* frozen conductivity, density, heat flow and Specific Heat.
*
* Parameters :
*     FOUT - name of the output file (Char).
*     Z - depth [m] (Real).
*     T - temperature [C] (Real).
*     ZKFR - conductivity depth [m] (Real).
*     KFR - frozen conductivity [W/(m*C]) (Real).
*     RHO - density [Kg/m^3] (Real).
*     Q - heat flow [Wm^-2] (Real).
*     C - Specific Heat (Real).
*     KFRAVE - average thermal conductivity [W/(m*C]) (Real).
*     RHOAVE - average density [Kg/m^3] (Real).
*     ALPHA - thermal diffusivity [m^2/s] (Real).
*     A - temperature change upon shoreline movement (Real).
*     X - distance from present shoreline (Real).
*     GRADTM - measured gradient anomaly (Real).
*     TIMES - desired search times [years] (Real).
*     GRADT - calculated gradient anomaly (Real).
*     NUMZ - Number of depth measurements (Integer).
*     NUMKFR - number of frozen conductivity measurements (Integer).
*     NUMTIME - Number of desired search times (Integer).
*     GRADM - gradient of best fit line between three points (Real).
*     GRADEQM - equilibrium gradient (Real).
*     MEAS - Flag , true if measured data is to be plotted (Logical).
*     MEASGR - measured gradient anomaly, used for plotting (REAl).
*     MEASTM - calculated time for MEASGR (Real).
*     NUMEAS - number of measured gradients and times (Integer).
* Variables
*     I,J,K - array index (Integer).
*     YEAR -  current year (Integer).
*     MONTH - current month (Integer).
*     DAY - todays date (Integer).
*     HOUR - current hour (Integer).
*     MIN - current minute (Integer).
*     SEC - current second (Integer).
*     S100TH - hundredths of a second (Integer).
*     INT - initial value of loop variable (Integer).
*     FINAL - final value of loop variable (Integer).
*     MORE - flag, if true more temperatures need to be printed (Logical).
*-----------------------------------------------------------------------
*
      SUBROUTINE REPORT(FOUT,Z,T,ZKFR,KFR,RHO,Q,C,KFRAVE,RHOAVE,ALPHA,
     +                  A,X,GRADTM,TIMES,GRADT,NUMZ,NUMKFR,NUMTIME,
     +                  GRADM,GRADEQM,MEAS,MEASGR,MEASTM,NUMEAS)
*
* declare variables ...
      CHARACTER FOUT*12
      REAL Z,T,ZKFR,KFR,RHO,Q,C,KFRAVE,RHOAVE,ALPHA,A,X,GRADTM,TIMES,
     +     GRADT,GRADM,GRADEQM,MEASGR,MEASTM
      INTEGER I,J,K,NUMZ,NUMKFR,YEAR,MONTH,DAY,HOUR,MIN,SEC,NUMTIME,
     +        S100TH,INT,FINAL,NUMEAS
      LOGICAL MORE,MEAS
```

```
      DIMENSION Z(NUMZ),T(NUMZ),ZKFR(NUMKFR),KFR(NUMKFR),RHO(NUMKFR),
     +          TIMES(NUMTIME),GRADT(NUMTIME,NUMZ),GRADTM(NUMZ-2),
     +          MEASGR(NUMEAS),MEASTM(NUMEAS)
* get the system date and time ...
      CALL GETDAT(YEAR,MONTH,DAY)
      CALL GETTIM(HOUR,MIN,SEC,S100TH)
* write headings for the data file ....
      WRITE(1,10) 'File : ', FOUT, 'Date :', YEAR, MONTH, DAY
      WRITE(1,15) 'Time : ', HOUR, MIN, SEC
      WRITE(1,*)
      WRITE(1,20) 'Depth Measured (Z)', 'Temperature Measured (T)',
     +            'Gradient Anomaly'
      WRITE(1,20) '      in [m]      ', '         in [C]         ',
     +            '   in [mK/m]    '
      WRITE(1,20) '------------------', '------------------------',
     +            '----------------'
      WRITE(1,*)
      WRITE(1,5) Z(1), T(1), 'N/A'
      WRITE(1,25) (Z(I), T(I), GRADTM(I-1), I=2,NUMZ-1)
      WRITE(1,5) Z(NUMZ), T(NUMZ), 'N/A'
      WRITE(*,'(A\)') '.'
      WRITE(1,*)
      WRITE(1,30) 'Frozen Conductivity Depth (ZKFR)', 'Frozen',
     +            ' Conductivity (KFR)', 'Density (RHO)'
      WRITE(1,30) '            in [m]              ', '     ',
     +            'in [W/(m*K)]        ', 'in [Kg/m^3] '
      WRITE(1,30) '--------------------------------', '------',
     +            '------------------', '-------------'
      WRITE(1,*)
      WRITE(1,35) (ZKFR(I), KFR(I), RHO(I), I=1, NUMKFR)
      WRITE(1,*)
      WRITE(1,40) 'Gradient (GRADM) ------------------> ', GRADM,
     +            ' [mK/m]'
      WRITE(1,40) 'Equilibrium Gradient (GRADEQM) -----> ', GRADEQM,
     +            ' [mK/m]'
      WRITE(1,40) 'Heat Flow (Q) ---------------------> ', Q,
     +            ' [W/m^2]'
      WRITE(1,40) 'Specific Heat (C) -----------------> ', C,
     +            ' [J/(Kg*K)]'
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      WRITE(1,45) 'DERIVED PHYSICAL PROPERTIES'
      WRITE(1,50) 'Average Thermal Conductivity (KFRAVE) -----> ',
     +            KFRAVE, ' [W/(m*K)]'
      WRITE(1,50) 'Average Density (RHOAVE) -----------------> ',
     +            RHOAVE, ' [Kg/m^3]'
      WRITE(1,52) 'Thermal Diffusivity (ALPHA) --------------> ',
     +            ALPHA,' [m^2/s]'
      WRITE(1,*)
      WRITE(1,45) 'FITTING PARAMETERS'
      WRITE(1,55) 'Temperature Change Upon Shoreline Movement (A) --',
     +            '---> ', A, ' [K]'
      WRITE(1,55) 'Distance From Present Shoreline (X) ------------',
     +            '---> ', X, ' [m]'
```

```
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      WRITE(1,75) '-------------------------------------------',
     +            '-------------------------------------------'
* print the depth, search times, and gradient anomaly ...
      INT = 1
      IF (NUMTIME .GT. 5) THEN
         MORE = .TRUE.
         FINAL = 5
      ELSE
         FINAL = NUMTIME
         MORE = .FALSE.
      ENDIF
100   WRITE(1,60) 'Depth (Z)','Gradient Anomaly [mK/m]'
      WRITE(1,65) 'Time [yrs] ----->',(TIMES(K), K=INT, FINAL)
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      DO 101 J=1,NUMZ
         WRITE(1,70) Z(J),(GRADT(K,J), K=INT, FINAL)
101   CONTINUE
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      IF (MORE) THEN
         INT = FINAL + 1
         IF (NUMTIME - FINAL .GT. 5) THEN
            FINAL = FINAL + 5
         ELSE
            FINAL = NUMTIME
            MORE = .FALSE.
         ENDIF
         GOTO 100
      ENDIF
      WRITE(*,'(A\)') '.'
      IF (MEAS) THEN
         WRITE(1,*)
         WRITE(1,80) 'Measured Gradient Anomaly', '  Time  '
         WRITE(1,80) '        in [mK/m]         ', 'in [yrs]'
         WRITE(1,80) '--------------------------', '--------'
         WRITE(1,80)
         WRITE(1,85) (MEASGR(I),MEASTM(I), I=1,NUMEAS)
      ENDIF
      PRINT*

* Format Statements ...
  5   FORMAT(9X,F7.2,25X,F7.3,27X,A)
 10   FORMAT(5X,A7,A,45X,A7,1X,I4,'-',I2,'-'I2)
 15   FORMAT(70X,A7,I2.2,1H:,I2.2,1H:,I2.2,)
 20   FORMAT(5X,A,10X,A,10X,A)
 25   FORMAT(9X,F7.2,25X,F7.3,23X,E10.4)
 30   FORMAT(5X,A,5X,A,A,5X,A)
 35   FORMAT(19X,F7.2,25X,F7.2,18X,F7.2)
 40   FORMAT(10X,A,F7.2,A)
 45   FORMAT(5X,A)
 50   FORMAT(10X,A,F7.2,A)
```

```
 52    FORMAT(10X,A,E10.4,A)
 55    FORMAT(10X,A,A,F7.2,A)
 60    FORMAT(5X,A,32X,A)
 65    FORMAT(7X,A,F7.1,5(7X,F7.1))
 70    FORMAT(6X,F7.2,6X,6(4X,E10.4))
 75    FORMAT(2X,A,A)
 80    FORMAT(5X,A,25X,A)
 85    FORMAT(12X,E10.4,33X,F7.1)
       RETURN
       END
*
********************** {END OF SUBROUTINE: REPORT} **********************
*
C SUBROUTINE ZTMEAS  1989-06-28-PPS
* Module to read measured data : depth, temperature, conductivity depth,
* frozen conductivity, density, heat flow and Specific Heat.
*
* Parameters :
*     FILEIN - name of file that contains the measured data (Char).
*     Z - depth [m] (Real).
*     T - temperature [C] (Real).
*     ZK - conductivity depth [m] (Real).
*     KFR - frozen conductivity [W/(m*C) (Real).
*     RHO - density [Kg/m^3] (Real).
*     Q - heat flow [Wm^-2] (Real).
*     C - Specific Heat (Real).
*     DIM1, DIM2 - array dimensions (Integer).
* Variables :
*     I - index (Integer).
*-----------------------------------------------------------------------
*
      SUBROUTINE ZTMEAS(FILEIN,Z,T,ZK,KFR,RHO,Q,C,DIM1,DIM2)
*
* declare variables ...
      REAL Z,T,ZK,KFR,RHO,Q,C
      INTEGER I,DIM1,DIM2
      CHARACTER FILEIN*12
      DIMENSION Z(DIM1),T(DIM1),ZK(DIM2),KFR(DIM2),RHO(DIM2)
* get the measured data ...
      CALL GETFID('Enter Measured Data File Name -----> ', FILEIN)
      OPEN (2,FILE=FILEIN,STATUS='OLD')
      REWIND(2)
* store the depth and temperature data into arrays ...
      I = 1
      READ(2,*)
      READ(2,*) Z(I), T(I)
      DO WHILE ((Z(I) .NE. 999.) .AND. (T(I) .NE. 999.))
         I = I + 1
         READ(2,*) Z(I), T(I)
      ENDDO
      DIM1 = I - 1
* store the conductivity depth,frozen conductivity  and density data
* into arrays ...
      I = 1
```

```
      READ(2,*)
      READ(2,*) ZK(I), KFR(I), RHO(I)
      DO WHILE ((ZK(I) .NE. 999.) .AND. (KFR(I) .NE. 999.) .AND.
     +         (RHO(I) .NE. 999.))
         I = I + 1
         READ(2,*) ZK(I), KFR(I), RHO(I)
      ENDDO
      DIM2 = I - 1
* get the heat flow and the Specific Heat ...
      READ(2,*)
      READ(2,*) Q, C
      CLOSE(2)
      RETURN
      END
*
********************** {END OF SUBROUTINE: ZTMEAS} **********************
```

APPENDIX II

SOURCE CODE FOR PROGRAM TRANS

```
$DEBUG
$STORAGE:2
*************************************************************************
*
      PROGRAM TRANS
*  Calculation of transient warming term for submarine permafrost.
*  See eqn. 27, Lachenbruch et al. 1982.
*  Mode 1 calculates curves for measured depth and temperature, mode 2
*  calculates curves for deserved depths and calculated temperatures,and
*  mode 3 calculates the transect for deserved water detphs.
*
*  COMPILE SEQUENCE: "FL /c TRANS.FOR"
*          Microsoft (R) FORTRAN Optimizing Compiler Version 5.00
*          Copyright (c) Microsoft Corp 1982-1989. All rights reserved.
*          TRANS.FOR
*  LINK SEQUENCE: "LINK"
*          Microsoft (R) Segmented-Executable Linker  Version 5.03
*          Copyright (C) Microsoft Corp 1984-1989.  All rights reserved.
*
*          Object Modules [.OBJ]: TRANS.OBJ/SE:1024
*          Run File [TRANS.EXE]: TRANS.EXE
*          List File [NUL.MAP]: TRANS.MAP
*          Libraries [.LIB]: PLOT88+LLIBFOR7+SUBFUN
*          Definitions File [NUL.DEF]: NUL.DEF
*
* PARAMETERS : NONE
C VARIABLES :
*   CHARACTER
*     - FOUT : Output file name.
*     - FILEIN : Input file name.
*     - DISKFL : Plot file name.
*     - TITLE : Title of graph to be plotted.
*     - TITBLK : Title of graph padded with leading blanks.
*     - MODE : mode of operation.
*     - XLABEL : Title label for the x-axis.
*     - YLABEL : Title label for the y-axis.
*     - CTEXT : Character string that contains the number of years
*               since inundation.
*     - AGAIN : If "yes" then run program again with different fitting
*               parameters.
*   INTEGER
*     - I, J : Array index.
*     - IOPORT : Input/Output port.
*     - MODEL : Output device identification.
*     - LENTIT : Length of character string "TITLE".
*     - LENBK : Number of blanks to pad.
*     - NUMZ : Number of depth measurements.
*     - NUMTM : Number of temperature measurements.
*     - NUMKFR : Number of frozen thermal conductivity measurements.
*     - NUMTIME : Number of desired search times.
*     - NUMPT : Number of data points.
*     - POINTS : Contains the first array index for each graph.
*     - NC : Number of characters returned in CTEXT.
```

```
*    REAL
*      - LAMDA : Thermal time constant [s].
*      - KFR(20) : Frozen thermal conductivity [W/(m*K)].
*      - KFRAVE : Average frozen thermal conductivity [W/(m*K)].
*      - THO : Initial land surface temperature before inundation [C].
*      - Z(30) : Depth below seabed [m].
*      - T(20) : Temperature [C].
*      - ZKFR(20) : Conductivity depth [m].
*      - RHO(20) :  Density [Kg/m^3].
*      - TIMES(20) : Desired search times [years].
*      - W_DEPTH(20) : Desired water depth for transect [m].
*      - THS : Seabed temperature after inundation [C].
*      - THB : Permafrost base temperature [C].
*      - Q : Heat flow [Wm^-2].
*      - ALPHA : Thermal diffusivity [m^2/s].
*      - ZPF : Equilibrium frozen thickness before submergence [m].
*      - FACT : Ratio of the desired drawing size to the normal drawing
*               size
*      - C : Specific Heat [J/(K*Kg)].
*      - TP : 2-D array of calculated temperature profiles.
*      - X : array of depth data used for plotting.
*      - Y : array of temperature data used for plotting.
*      - XTEXT, YTEXT : Location of CTEXT
*      - HEIGHT : height of symbol.
*    LOGICAL
*      - OVERFL : Flag, if true then overwrites an existing file.
*      - SKIP : Flag, if true then skips the plotting routine.
*-----------------------------------------------------------------------
*
* these external functions and subroutines are found in library
* SUBFUN.LIB ....
      EXTERNAL GETFID,CRTFID,DEVICE,LENTRM
C Declare variables ...
      CHARACTER FOUT*12,TITLE*25,MODE*1,XLABEL*25,YLABEL*25,TITBLK*25,
     +          CTEXT*10,FILEIN*12,AGAIN*1,DISKFL*12
      INTEGER LENTIT,IOPORT,MODEL,NUMZ,NUMKFR,NUMTIME,NUMPT,LENBK,
     +        POINTS,NC,I,NUMTM,J
      REAL LAMBDA,KFR,Z,T,FACT,ZKFR,RHO,Q,C,THO,ZPF,ALPHA,KFRAVE,RHOAVE,
     +     THS,THB,TIMES,TP,X,Y,XTEXT,YTEXT,HEIGHT,W_DEPTH
      LOGICAL OVERFL,SKIP
      DIMENSION Z(30),T(30),KFR(20),ZKFR(20),RHO(20),TIMES(20),
     +          TP(20,30),X(500),Y(500),POINTS(20),W_DEPTH(20)

* Initialize variables ...
      NUMZ = 30
      NUMTM = 30
      NUMTIME = 20
      NUMKFR = 20
      XLABEL = 'Depth Below Seabed (m)'
      YLABEL = 'Temperature ( C)'
C Read mode of operation ...
      WRITE(*,'(5X,A)') '1. Read Measured Z, T Data                    '
      WRITE(*,'(5X,A)') '2. Predict Z, T Profiles From Given Parameters'
      WRITE(*,'(5X,A)') '3. Predict Transect For Water Depths'
```

```
 110  MODE = ' '
      WRITE(*,'(A\)') ' Choose Mode Of Operation {1,2,3} -----> '
      READ(*,'(A)',ERR=110) MODE
      PRINT*
* Process according to the modes of operation ...
      SELECT CASE(MODE)
         CASE('1')
C Read measured data ...
            CALL ZTMEAS(FILEIN,Z,T,ZKFR,KFR,RHO,Q,C,TIMES,NUMTM,NUMZ,
     +                  NUMKFR,NUMTIME)
         CASE('2')
* Read depths required for calculated temperatures ...
            CALL ZTPRED(FILEIN,Z,ZKFR,KFR,RHO,Q,C,TIMES,NUMZ,NUMKFR,
     +                  NUMTIME)
         CASE('3')
* Read times and water depths required for traansect ...
            CALL ZTTRAN(FILEIN,Z,ZKFR,KFR,RHO,Q,C,TIMES,W_DEPTH,NUMZ,
     +                  NUMKFR,NUMTIME)
         CASE DEFAULT
            GOTO 110
      END SELECT
      PRINT*
C Prompt user for output file name ...
      OVERFL = .FALSE.
      CALL CRTFID('Enter Output File Name -----> ',FOUT,OVERFL)
      IF (OVERFL) THEN
         OPEN(1,FILE=FOUT,STATUS='OLD')
         CLOSE(1,STATUS='DELETE')
      ENDIF
      OPEN(1,FILE=FOUT,STATUS='NEW')
      PRINT*
C Prompt user for device parameters for 'PLOTS' statement ...
      CALL DEVICE(IOPORT,MODEL,FACT,DISKFL)
* determine whether to skip the plotting routine ...
      IF (IOPORT .EQ. 0 .AND. MODEL .EQ. 0 .AND. FACT .EQ. 0.0) THEN
         SKIP = .TRUE.
      ELSE
         SKIP = .FALSE.
      ENDIF
      IF (.NOT. SKIP) THEN
* Prompt for title for graph ...
         PRINT*
         WRITE(*,'(A\)') ' Enter Graph Title (max. 25 characters)'
         WRITE(*,'(A\)') ' -----> '
         READ(*,'(BN,A25)')TITLE
         PRINT*
* Determine length of title ...
         LENTIT = LENTRM(TITLE)
* pad title with leading blanks for centering in subroutine GRAPH
         TITBLK = '                         '
         IF (LENTIT .LT. LEN(TITLE)) THEN
            LENBK = (NINT(LEN(TITLE)-LENTIT)/2.)
            IF (LENBK .EQ. 0) THEN
                LENBK = 1
```

```
            ENDIF
            TITBLK = TITBLK(1:LENBK) // TITLE
            LENTIT = LENTIT + LENBK
         ELSE
            TITBLK = TITLE
         ENDIF
      ENDIF
C Calculate derived physical properties ...
      CALL DERPHY(KFR,RHO,C,NUMKFR,KFRAVE,RHOAVE,ALPHA)
C Read fitting parameters ...
 500  CALL FITPAR(THO,THS,THB)
C Calculate variable physical properties ...
      CALL VARPHY(KFRAVE,THO,THB,Q,ALPHA,ZPF,LAMBDA)
C Calculate temperature profile ...
      PRINT*
      WRITE(*,'(A\)') ' Calculating Temperature Profiles  '
      CALL TEMPRO(MODE,Z,TIMES,ZPF,LAMBDA,THO,THS,THB,NUMZ,NUMTIME,
     +            TP,NUMPT)
C Write to output file the above data that was read in ...
      PRINT*
      WRITE(*,'(A\)') ' Generating A Report File  '
      CALL REPORT(FOUT,Z,T,ZKFR,KFR,RHO,Q,C,KFRAVE,RHOAVE,ALPHA,THO,THS,
     +           THB,ZPF,LAMBDA,TIMES,TP,NUMTM,NUMZ,NUMKFR,NUMTIME,MODE)
      CLOSE(1)
* If user wishes to plott ...
      IF (.NOT. SKIP) THEN
C Form arrays X, Y for plotting ...
         PRINT*
         WRITE(*,'(A\)') ' Forming Arrays For Plotting  '
         CALL FARRAY(MODE,Z,TP,T,THO,THB,ZPF,X,Y,NUMTM,NUMZ,NUMTIME,
     +               NUMPT,POINTS)
C Plot graph ...
         CALL PLOTS(0,IOPORT,MODEL)
         WRITE(*,'(A\)') ' Generating Parameters For PLOT88  '
         CALL GRAPH(X,Y,POINTS,FACT,TITBLK,LENTIT,XLABEL,
     +              YLABEL,NUMPT,DISKFL)

* add labels to the graph ...
* first set the width to height ratio of characters (condensed) ...
         CALL ASPECT(0.50)
* define the COMPLX symbol font ...
         CALL COMPLX
C add fitting parameters ...
         CALL SYMBOL(12.,.5,.2,'Fitting Parameters',90.,18)
         CALL SYMBOL(12.5,.8,.2,'THO = ',90.,6)
         CALL SYMBOL(12.75,.8,.2,'THS = ',90.,6)
         CALL SYMBOL(13.0,.8,.2,'THB = ',90.,6)
         WRITE(*,'(A\)') '.'
         CALL NUMBER(12.5,1.4,.2,THO,90.,2)
         CALL NUMBER(12.75,1.4,.2,THS,90.,2)
         CALL NUMBER(13.,1.4,.2,THB,90.,2)
* add input and output file names ....
         CALL SYMBOL(13.5,.5,.2,'Input File :',90.,12)
         CALL SYMBOL(13.5,1.6,.2,FILEIN,90.,12)
```

```
         CALL SYMBOL(14.,.5,.2,'Output File :',90.,13)
         CALL SYMBOL(14.,1.7,.2,FOUT,90.,12)
C add legend ...
         IF (MODE .EQ. '3') THEN
            CALL SYMBOL(12.,6.0,.2,'Water Depth [m]',90.,15)
         ELSE
            CALL SYMBOL(12.,5.6,.2,'Time Since Inundation [yrs]',90.,27)
         ENDIF
         WRITE(*,'(A\)') '.'
         XTEXT = 12.5
         YTEXT = 6.
         J = 0
         DO 200 I=1, NUMTIME
            CALL SYMBOL(XTEXT,YTEXT,.2,CHAR(J),0.0,-1)
            IF (MODE .EQ. '3') THEN
               CALL GETNUM(W_DEPTH(I),1,CTEXT,NC)
            ELSE
               CALL GETNUM(TIMES(I),1,CTEXT,NC)
            ENDIF
            CALL SYMBOL(XTEXT+.05,YTEXT+.2,.2,CTEXT,90.0,NC)
            XTEXT = XTEXT + .30
            IF (I .EQ. ((NUMTIME+1)/2)) THEN
               YTEXT = YTEXT + 1.25
               XTEXT = 12.5
            ENDIF
            J = J + 1
            IF (J .EQ. 6) J = 10
            IF (J .EQ. 12) J = 14
            IF (J .EQ. 16) J = 0
 200     CONTINUE
         IF (MODE .EQ. '1') THEN
            CALL SYMBOL(XTEXT,YTEXT+.2,.2,'Measured Data',90.,13)
            DO 220 HEIGHT=.15, .01, -.02
               CALL SYMBOL(XTEXT,YTEXT,HEIGHT,CHAR(1),0.0,-1)
 220        CONTINUE
         ENDIF
         WRITE(*,'(A\)') '.'
         PRINT*
         PRINT*, 'Drawing Plot ...'
         CALL PLOT(0.,0.,999)
      ENDIF
      PRINT*
C Determine whether to run program again
 120  AGAIN = ' '
      WRITE(*,'(1X,A\)') 'Another Run With Different Fitting Parameter'
      WRITE(*,'(A\)') ' {Y/N} -----> '
      READ(*,'(A)',ERR=120) AGAIN
      IF (AGAIN .EQ. 'Y' .OR. AGAIN .EQ. 'y') THEN
* Prompt user for output file name ...
         OVERFL = .FALSE.
         CALL CRTFID('Enter Output File Name -----> ',FOUT,OVERFL)
         IF (OVERFL) THEN
            OPEN(1,FILE=FOUT,STATUS='OLD')
            CLOSE(1,STATUS='DELETE')
```

```
            ENDIF
            OPEN(1,FILE=FOUT,STATUS='NEW')
            PRINT*
            CALL DEVICE(IOPORT,MODEL,FACT,DISKFL)
            PRINT*
            GOTO 500
         ENDIF
         STOP
 1000 END
*
*****************************************************************************
*
C SUBROUTINE DERPHY  1989-06-30-PPS
* Module to calculate derived physical properties : average thermal
* conductivity, average density, and thermal diffusivity.
*
* Parameters :
*     KFR - frozen conductivity [W/(m*K)] (Real).
*     RHO - density [Kg/m^3] (Real).
*     C - Specific Heat (Real).
*     NUMKFR - number of frozen conductivity measurements (Integer).
*     KFRAVE - average frozen thermal conductivity [W/(m*K)(Real).
*     RHOAVE - average density [Kg/m^3] (Real).
*     ALPHA -  thermal diffusivity [m^2/s] (Real).
*
* Variables
*     I - index (Integer).
*
*-----------------------------------------------------------------------------
*
      SUBROUTINE DERPHY(KFR,RHO,C,NUMKFR,KFRAVE,RHOAVE,ALPHA)
*
* declare variables ...
      REAL  KFR,RHO,C,KFRAVE,RHOAVE,ALPHA
      INTEGER I,NUMKFR
      DIMENSION KFR(NUMKFR),RHO(NUMKFR)
* calculate the average frozen thermal conductivity and the average
* density...
      DO 100 I=1, NUMKFR
         KFRAVE = KFRAVE + KFR(I)
         RHOAVE = RHOAVE + RHO(I)
 100  CONTINUE
      KFRAVE = KFRAVE/NUMKFR
      RHOAVE = RHOAVE/NUMKFR
* calculate the thermal diffusivity ...
      ALPHA = KFRAVE / (C*RHOAVE)
      RETURN
      END
*
********************** {END OF SUBROUTINE: DERPHY} **********************
*
C SUBROUTINE FARRAY  1989-07-12-PPS
* Module to form arrays for plotting
*
```

```
* Parameters :
*     MODE - mode of operation (Char).
*     Z - depth  (Real).
*     TP - 2-D array of calculated temperature profiles (Real).
*     TM - measured temperatures (Real).
*     THO - initial land surface temperature (Real).
*     THB - permafrost base temperature [C] (Real).
*     ZPF - equilibrium frozen thickness before submergence [m] (Real).
*     X - depths of calculated data (Real).
*     Y - temperatures of calculated data (Real).
*     NUMTM - number of temperatures measured (Integer).
*     NUMZ - number of depths (Integer).
*     NUMTIME - number of times (Integer).
*     NUMPT - number of data points (Integer).
*     POINTS - contains the first array index of each graph (Integer).
* Variables
*     I,J,K - array indexes (Integer).
*     POS - position in the pointer array (Integer).
*-----------------------------------------------------------------------
*
      SUBROUTINE FARRAY(MODE,Z,TP,TM,THO,THB,ZPF,X,Y,NUMTM,NUMZ,NUMTIME,
     +                  NUMPT,POINTS)
*
* declare variables ...
      CHARACTER MODE*1
      REAL Z,TP,X,Y,TM,THO,THB,ZPF
      INTEGER I,J,K,NUMZ,NUMTM,NUMTIME,NUMPT,POINTS,POS
      DIMENSION Z(NUMZ),TP(NUMTIME,NUMZ),X(NUMPT),Y(NUMPT),POINTS(20),
     +          TM(NUMTM)
*  FORM ARRAYS X,Y = Z,TH(Z,T) FOR PLOTTING
      DO 100 I=1,NUMTIME
         DO 101 J=1,NUMZ
            K=I*NUMZ-NUMZ+J
            X(K)=Z(J)
            Y(K)=TP(I,J)
101      CONTINUE
         POINTS(I) = I*NUMZ - NUMZ + 1
         WRITE(*,'(A\)') '.'
100   CONTINUE
* add the ending value ...
      POS = NUMTIME + 1
      POINTS(POS) = POINTS(POS-1) + NUMZ
      POS = POS + 1
* add equilibrium line to X, Y arrays ...
      Y(K+1) = THO
      X(K+1) = 0
      IF (ZPF .GT. Z(NUMZ)) THEN
         X(K+2) = Z(NUMZ)
         Y(K+2) = (THB-THO)/ZPF*Z(NUMZ)+THO
      ELSE
         X(K+2) = ZPF
         Y(K+2) = THB
      ENDIF
      POINTS(POS) = POINTS(POS-1) + 2
```

```
      IF (MODE .EQ. '1') THEN
*  Add the measured data to arrays X, Y
         K = K + 3
         DO 200 J=1, NUMTM
            X(K) = Z(J)
            Y(K) = TM(J)
            K = K + 1
 200     CONTINUE
         POINTS(POS+1) = POINTS(POS) + NUMTM
      ENDIF
      PRINT*
      RETURN
      END
*
********************** {END OF SUBROUTINE: FARRAY} **********************
*
C SUBROUTINE FITPAR  1989-06-30-PPS
* Module to read the fitting paramaters from the keyboard.
* Parameters :
*    THO - initial land surface temperature (Real).
*    TMPSEA - seabed temperature (Real).
*    TMPBAS - permafrost base temperature (Real).
* Variables :
*-----------------------------------------------------------------------
*
      SUBROUTINE FITPAR(THO,TMPSEA,TMPBAS)
*
* Declare variables ...
      REAL THO,TMPSEA,TMPBAS
* Prompt user to enter the fitting parameters ...
      WRITE(*,*) 'ENTER FITTING PARAMATERS'
 100  WRITE(*,'(A\)') '    Initial Land Surface Temperature [C] -----> '
      READ(*,*,ERR=100) THO
 110  WRITE(*,'(A\)') '    Seabed Temperature [C] -------------------> '
      READ(*,*,ERR=110) TMPSEA
 120  WRITE(*,'(A\)') '    Permafrost Base Temperature [C] ----------> '
      READ(*,*,ERR=120) TMPBAS
      RETURN
      END
*
********************** {END OF SUBROUTINE: FITPAR} **********************
*
C SUBROUTINE GRAPH  1989-07-13-PPS
* Generic plotting routine for X, Y data arrays.
*
* Parameters :
*    X - array of X elements (Real).
*    Y - array of Y elements (Real).
*    POINTS - contains the first array index of each graph that is to
*             be plotted (Integer).
*    FACT - Ratio of the desired drawing size to the normal drawing
*           size (Real).
*    TITLE - Title of graph to be plotted (Char).
*    LENTIT - Length of character string "TITLE" (Integer).
```

```
*     XLABEL - Title label for the x-axis (Char).
*     YLABEL - Title label for the y-axis (Char).
*     NUMPT - dimension of the data arrays (Integer).
* Variables :
*     XO - temporary array of X data (Real).
*     YO - temporary array of Y data (Real).
*     MEAS - flag , true if measured data is to be plotted (Logical).
*     I, J, K - array indexes (Integer).
*     INTEQ - the index of the centered symbol to be drawn at each pt (Int)
*     HEIGHT - height of centered symbol (Real).
*-----------------------------------------------------------------------
*
      SUBROUTINE GRAPH(X,Y,POINTS,FACT,TITLE,LENTIT,XLABEL,
     +                 YLABEL,NUMPT,DISKFL)

* declare variables ...
      REAL X,Y,FACT,XO,YO,HEIGHT
      INTEGER LENTIT,NUMPT,POINTS,I,J,K,INTEQ
      CHARACTER*(*) XLABEL,YLABEL,TITLE,DISKFL
      LOGICAL MEAS
      DIMENSION X(NUMPT+2),Y(NUMPT+2),XO(50),YO(50),POINTS(20)
* determine whether to write to disk ...
      IF (DISKFL .NE. '~~~~~~~~.~~~') THEN
         OPEN(90,FILE=DISKFL,STATUS='NEW',FORM='BINARY')
      ENDIF
* enlarge or reduce the size of the entire drawing ...
      CALL FACTOR(FACT)
* define a new origin ...
      CALL PLOT(1.2,2.7,-3)
      WRITE(*,'(A\)') '.'
* define the TRIPLX symbol font to be used by SYMBOL ...
      CALL TRIPLX
* add title ...
      CALL SYMBOL(11.0,0.0,.4,TITLE,90.,LENTIT)
* scale data ...
      CALL SCALE(X,10.,NUMPT,1)
      CALL SCALE(Y,10.,NUMPT,1)
      WRITE(*,'(A\)') '.'
* define the COMPLEX symbol font to be used by SYMBOL ...
      CALL COMPLX
* set the characteristics of the X-axis annotation ...
      CALL STAXIS(.222,.3,.111,.111,-1)
* drawn the X-axis ...
      CALL AXIS(0.,0.,XLABEL,-22,10.,0.,X(NUMPT+1),X(NUMPT+2))
      WRITE(*,'(A\)') '.'
* set Y-axis annotation ...
      CALL STAXIS(.222,.3,.111,.111,-1)
* draw Y-axis ...
      CALL AXIS(0.,0.,YLABEL,16,10.,90.,Y(NUMPT+1),
     +          Y(NUMPT+2))
      CALL SYMBOL(-.97,6.55,.2,'o',90.,1)
      WRITE(*,'(A\)') '.'
* set the line type and centered symbol characteristics ...
      CALL STLINE(1,.1,0.0)
```

```
* form arrays XO, YO to plot each graph with a different symbol ...
      MEAS = .FALSE.
      J = 1
      INTEQ = 0
 200  K = 1
      DO 100 I=POINTS(J), POINTS(J+1)-1
         XO(K) = X(I)
         YO(K) = Y(I)
         K = K + 1
 100  CONTINUE
      XO(K) = X(NUMPT+1)
      XO(K+1) = X(NUMPT+2)
      YO(K) = Y(NUMPT+1)
      YO(K+1) = Y(NUMPT+2)
* draw equilibrium line ...
      IF (POINTS(J+1) - POINTS(J) .EQ. 2) THEN
         CALL LINE(XO,YO,K-1,1,0,0)
         MEAS = .TRUE.
      ELSE
* draw the centered symbols ...
         IF (MEAS) THEN
            DO 111 HEIGHT=.1, .01, -.02
               CALL STLINE(1,HEIGHT,0.)
               CALL LINE(XO,YO,K-1,1,-1,1)
 111        CONTINUE
         ELSE
            CALL LINE(XO,YO,K-1,1,-1,INTEQ)
            INTEQ = INTEQ +  1
            IF (INTEQ .EQ. 6) INTEQ = 10
            IF (INTEQ .EQ. 12) INTEQ = 14
            IF (INTEQ .EQ. 16) INTEQ = 0
* draw a curve through the centered symbols ...
            CALL CURVE(XO,YO,K-1,.05)
         ENDIF
      ENDIF
      J = J + 1
      IF (POINTS(J+1) .NE. 0) THEN
         GOTO 200
      ENDIF
      WRITE(*,'(A\)') '.'
      RETURN
      END
*
********************** {END OF SUBROUTINE: GRAPH} ***********************
*
C SUBROUTINE REPORT  1989-07-06-PPS
* Module to print measured data : depth, temperature, conductivity depth,
* frozen conductivity, density, heat flow and Specific Heat.
*
* Parameters :
*    FOUT - name of the output file (Char).
*    Z - depth [m] (Real).
*    T - temperature [C] (Real).
*    ZKFR - conductivity depth [m] (Real).
```

```
*     KFR - frozen conductivity [W/(m*C]) (Real).
*     RHO - density [Kg/m^3] (Real).
*     Q - heat flow [Wm^-2] (Real).
*     C - Specific Heat (Real).
*     KFRAVE - average thermal conductivity [W/(m*C]) (Real).
*     RHOAVE - average density [Kg/m^3] (Real).
*     ALPHA - thermal diffusivity [m^2/s] (Real).
*     THO - initial land surface temperature before inundation [C] (Real).
*     THS - seabed temperature after inundation [C] (Real).
*     THB - permafrost base temperature [C] (Real).
*     ZPF - equilibrium frozen thickness before submergence [m] (Real).
*     LAMBDA - thermal time constant [s] (Real).
*     TIMES - desired search times [years] (Real).
*     TP - 2-D array of calculated temperature profiles (Real).
*     NUMTM - number of measured temperatures (Integer).
*     DIM1, DIM2, DIM3 - array dimensions (Integer).
*     MODE : mode of operation (Char).
* Variables :
*     LAMBYR - thermal time constant in years (Real).
*     I,J,K - array index (Integer).
*     YEAR -  current year (Integer).
*     MONTH - current month (Integer).
*     DAY - todays date (Integer).
*     HOUR - current hour (Integer).
*     MIN - current minute (Integer).
*     SEC - current second (Integer).
*     S100TH - hundredths of a second (Integer).
*     INT - initial value of loop variable (Integer).
*     FINAL - final value of loop variable (Integer).
*     MORE - flag, if true more temperatures need to be printed (Logical).
*-----------------------------------------------------------------------
*
      SUBROUTINE REPORT(FOUT,Z,T,ZKFR,KFR,RHO,Q,C,KFRAVE,RHOAVE,ALPHA,
     +           THO,THS,THB,ZPF,LAMBDA,TIMES,TP,NUMTM,DIM1,DIM2,DIM3,
     +           MODE)
*
* declare variables ...
      CHARACTER FOUT*12,MODE*1
      REAL Z,T,ZKFR,KFR,RHO,Q,C,KFRAVE,RHOAVE,ALPHA,THO,
     +     THS,THB,ZPF,LAMBDA,TIMES,TP,LAMBYR
      INTEGER I,J,K,DIM1,DIM2,YEAR,MONTH,DAY,HOUR,MIN,SEC,
     +        DIM3,S100TH,INT,FINAL,NUMTM
      LOGICAL MORE
      DIMENSION Z(DIM1),T(NUMTM),ZKFR(DIM2),KFR(DIM2),RHO(DIM2)
      DIMENSION TIMES(DIM3),TP(DIM3,DIM1)
* convert the thermal time constant from seconds to years ...
      LAMBYR = LAMBDA/31536000.0
* get the system date and time ...
      CALL GETDAT(YEAR,MONTH,DAY)
      CALL GETTIM(HOUR,MIN,SEC,S100TH)
* write headings for the data file ....
      WRITE(1,10) 'File : ', FOUT, 'Date :', YEAR, MONTH, DAY
      WRITE(1,15) 'Time : ', HOUR, MIN, SEC
      WRITE(1,*)
```

```
      IF (MODE .EQ. '1') THEN
         WRITE(1,20) 'Depth Measured (Z)', 'Temperature Measured (T)'
         WRITE(1,20) '      in [m]      ', '          in [C]          '
         WRITE(1,20) '------------------', '------------------------'
         WRITE(1,*)
         WRITE(1,25) (Z(I), T(I), I=1, NUMTM)
      ENDIF
      WRITE(*,'(A\)') '.'
      WRITE(1,*)
      WRITE(1,30) 'Frozen Conductivity Depth (ZKFR)', 'Frozen',
     +            ' Conductivity (KFR)', 'Density (RHO)'
      WRITE(1,30) '            in [m]              ', '      ',
     +            'in [W/(m*K)]        ', 'in [Kg/m^3] '
      WRITE(1,30) '--------------------------------', '------',
     +            '------------------', '-------------'
      WRITE(1,*)
      WRITE(1,35) (ZKFR(I), KFR(I), RHO(I), I=1, DIM2)
      WRITE(1,*)
      WRITE(1,40) 'Heat Flow (Q) ------------> ',Q, ' [W/m^2]'
      WRITE(1,40) 'Specific Heat (C) --------> ',C, ' [J/(Kg*K)]'
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      WRITE(1,45) 'DERIVED PHYSICAL PROPERTIES'
      WRITE(1,50) 'Average Thermal Conductivity (KFRAVE) -----> ',
     +            KFRAVE, ' [W/(m*K)]'
      WRITE(1,50) 'Average Density (RHOAVE) ------------------> ',
     +            RHOAVE, ' [Kg/m^3]'
      WRITE(1,52) 'Thermal Diffusivity (ALPHA) ---------------> ',
     +            ALPHA,' [m^2/s]'
      WRITE(1,*)
      WRITE(1,45) 'FITTING PARAMETERS'
      WRITE(1,55) 'Initial Land Surface Temperature Before Inundation',
     +            ' (THO) -----> ',THO,' [C]'
      WRITE(1,55) 'Seabed Temperature After Inundation (THS) --------',
     +            '------------> ',THS,' [C]'
      WRITE(1,55) 'Permafrost Base Temperature (THB) ----------------',
     +            '------------> ',THB,' [C]'
      WRITE(1,*)
      WRITE(1,45) 'VARIABLE PHYSICAL PROPERTIES'
      WRITE(1,55) 'Equilibrium Frozen Thickness Before Submergence',
     +            ' (ZPF) -----> ', ZPF,' [m]'
      WRITE(1,55) 'Thermal Time Constant (LAMBDA)  ---------------',
     +            '------------> ', LAMBYR, ' [yrs]'
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      WRITE(1,75) '-------------------------------------------',
     +            '-------------------------------------------'
*  print the depth, search times, and temperature ...
      INT = 1
      IF (DIM3 .GT. 6) THEN
         MORE = .TRUE.
         FINAL = 6
      ELSE
         FINAL = DIM3
```

```
         MORE = .FALSE.
      ENDIF
100   WRITE(1,60) 'Depth (Z)','Temperature (T)'
      WRITE(1,65) 'Time [yrs] ----->',(TIMES(K), K=INT, FINAL)
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      DO 101 J=1,DIM1
         WRITE(1,70) Z(J),(TP(K,J), K=INT, FINAL)
101   CONTINUE
      WRITE(1,*)
      WRITE(*,'(A\)') '.'
      IF (MORE) THEN
         INT = FINAL + 1
         IF (DIM3 - FINAL .GT. 6) THEN
            FINAL = FINAL + 6
         ELSE
            FINAL = DIM3
            MORE = .FALSE.
         ENDIF
         GOTO 100
      ENDIF
      WRITE(*,'(A\)') '.'
      PRINT*
* Format Statements ...
 10   FORMAT(5X,A7,A,45X,A7,1X,I4,'-',I2,'-'I2)
 15   FORMAT(70X,A7,I2.2,1H:,I2.2,1H:,I2.2,)
 20   FORMAT(10X,A,15X,A)
 25   FORMAT(15X,F7.2,30X,F7.3)
 30   FORMAT(5X,A,5X,A,A,5X,A)
 35   FORMAT(19X,F7.2,25X,F7.2,18X,F7.2)
 40   FORMAT(10X,A,F7.2,A)
 45   FORMAT(5X,A)
 50   FORMAT(10X,A,F7.2,A)
 52   FORMAT(10X,A,E10.4,A)
 55   FORMAT(10X,A,A,F7.2,A)
 60   FORMAT(5X,A,32X,A)
 65   FORMAT(7X,A,2X,F7.1,5(4X,F7.1))
 70   FORMAT(6X,F7.2,9X,6(4X,F7.3))
 75   FORMAT(2X,A,A)
      RETURN
      END
*
********************** {END OF SUBROUTINE: REPORT} **********************
*
C SUBROUTINE SUMS  1989-06-30-PPS
* Module to calculate the sum in Eqn. 27, Lachenbruch et al. 1982
*
* Parameters :
*     Z - depth (Real).
*     TIME - time in seconds (Real).
*     ZPF - equilibrium frozen thickness before submergence (Real).
*     LAMBDA - thermal time constant (Real).
*     SUM - sum of the terms in equ. 27 of Lachenbruch.
* Variables :
```

```
*     I - loop counter (Integer).
*     S - Temporary variable to store indetemediate calculation (Real).
* Constant :
*     PI - 3.14159 (Real).
*-----------------------------------------------------------------------
      SUBROUTINE SUMS(Z,TIME,ZPF,LAMBDA,SUM)
*
* declare variables ...
      REAL SUM,Z,TIME,ZPF,LAMBDA,PI,S
      INTEGER I
      PI = 4*ATAN(1.0)
* Calculate the sum ...
      SUM = 0.
      DO 10 I=1, 30
         S = EXP(-I**2*PI**2*TIME/(4.*LAMBDA))*SIN(I*PI*Z/ZPF)/I
         SUM = SUM + S
 10   CONTINUE
      RETURN
      END
*
************************ {END OF SUBROUTINE: SUMS} ************************
*
C SUBROUTINE TEMPRO  1989-07-11-PPS
* Module to calculate temperature profiles at each depth Z(I) for varying
* times TIMES(I).
* Parameters :
*     MODE - mode of operation (Char).
*     Z - depth (Real).
*     TIMES - desired search times (Real).
*     ZPF -  equilibrium frozen thickness before submergence (Real).
*     LAMBDA - thermal time constant (Real).
*     THO - initial land surface temperature (Real).
*     THS - seabed temperature after inundation [C] (Real).
*     THB - permafrost base temperature (Real).
*     NUMZ - Number of depth measurements (Integer).
*     NUMTIME - Number of desired search times (Integer).
*     TP - 2-D array of calculated temperature profiles (Real).
*     NUMPT - number of data points (Integer).
* Variables :
*     SUM - sum of the terms in equ. 27 of Lachenbruch (Real).
*     I,J - array indexes (Integer).
*     TIMESEC - times converted from years into seconds (Real).
* Constant :
*     PI - 3.14159 (Real).
*     YRSEC - 31536000.0 number of seconds in one year (Real).
*-----------------------------------------------------------------------
*
      SUBROUTINE TEMPRO(MODE,Z,TIMES,ZPF,LAMBDA,THO,THS,THB,NUMZ,
     +                  NUMTIME,TP,NUMPT)

*
* Declare variables ...
      REAL Z,ZPF,LAMBDA,THO,THS,THB,TP,TIMESEC,SUM,PI,YRSEC,TIMES
      CHARACTER MODE*1
```

```
      INTEGER I,J,NUMTIME,NUMZ,NUMPT
      DIMENSION Z(NUMZ),TP(NUMTIME,NUMZ),TIMES(NUMTIME)
      PI = 4*ATAN(1.0)
* Number of seconds in a year ...
      YRSEC = 31536000.0
* Range through all times ...
      DO 201 I = 1, NUMTIME
         TIMESEC = TIMES(I)*YRSEC
* Range through all depths ...
         DO 301 J = 1, NUMZ
            CALL SUMS(Z(J),TIMESEC,ZPF,LAMBDA,SUM)
            TP(I,J)=THS+(THB-THS)*Z(J)/ZPF+(THO-THS)*2./PI*SUM
 301     CONTINUE
         WRITE(*,'(A\)') '.'
 201  CONTINUE
      PRINT*
* the extra two pts are for the equilibrium line ...
      IF (MODE .EQ. '1') THEN
         NUMPT = NUMZ*NUMTIME + NUMZ + 2
      ELSE
         NUMPT = NUMZ*NUMTIME + 2
      ENDIF
      RETURN
      END
*
********************** {END OF SUBROUTINE: TEMPRO} **********************
*
C SUBROUTINE VARPHY  1989-06-30-PPS
* Module to calculate varying physical properties : equilibrium permafrost
* thickness before submergence and thermal time constant.
*
* Parameters :
*    KFRAVE - average frozen conductivity [W/(m*C]) (Real).
*    THO - initial land surface temperature [C] (Real).
*    THB - permafrost base temperature (Real).
*    Q - heat flow [Wm^-2] (Real).
*    ALPHA -  thermal diffusivity [m^2/s] (Real).
*    ZPF - equilibrium frozen thickness before submergence [m] (Real).
*    LAMBDA - thermal time constant [s] (Real).
* Variables :
*-----------------------------------------------------------------------
*
      SUBROUTINE VARPHY(KFRAVE,THO,THB,Q,ALPHA,ZPF,LAMBDA)
*
* declare variables ...
      REAL  KFRAVE,THO,Q,ALPHA,ZPF,LAMBDA
* calculate the equilibrium permafrost thickness before submergence ...
      ZPF = -KFRAVE*(THO-THB)/Q
* calculate the  thermal time constant ...
      LAMBDA = ZPF**2 / (4.*ALPHA)
      RETURN
      END
*
********************** {END OF SUBROUTINE: VARPHY} **********************
```

```
*
C SUBROUTINE ZTMEAS  1989-06-28-PPS
* Module to read measured data : depth, temperature, conductivity depth,
* frozen conductivity, density, heat flow and Specific Heat.
*
* Parameters :
*    FILEIN - name of file that contains the measured data (Char).
*    Z - depth [m] (Real).
*    T - temperature [C] (Real).
*    ZK - conductivity depth [m] (Real).
*    KFR - frozen conductivity [W/(m*K) (Real).
*    RHO - density [Kg/m^3] (Real).
*    Q - heat flow [Wm^-2] (Real).
*    C - Specific Heat (Real).
*    TIMES - desired search times (Real).
*    NUMTM - number of measured temperatures (Integer).
*    DIM1, DIM2, DIM3 - array dimensions (Integer).
* Variables :
*    I - index (Integer).
*-----------------------------------------------------------------------
*
      SUBROUTINE ZTMEAS(FILEIN,Z,T,ZK,KFR,RHO,Q,C,TIMES,NUMTM,DIM1,DIM2,
     +                  DIM3)
*
* declare variables ...
      REAL Z,T,ZK,KFR,RHO,Q,C,TIMES
      INTEGER I,DIM1,DIM2,DIM3
      CHARACTER FILEIN*12
      DIMENSION Z(DIM1),T(DIM1),ZK(DIM2),KFR(DIM2),RHO(DIM2),
     +          TIMES(DIM3)
* get the measured data ...
      CALL GETFID('Enter Measured Data File Name -----> ',FILEIN)
      OPEN (2, FILE=FILEIN, STATUS='OLD')
      REWIND(2)
* store the depth and temperature data into arrays ...
      I = 1
      READ(2,*)
      READ(2,*) Z(I), T(I)
      DO WHILE (T(I) .NE. 999.)
         I = I + 1
         READ(2,*) Z(I), T(I)
      ENDDO
      NUMTM = I - 1
* if there are any more depths, read them ...
      DO WHILE (Z(I) .NE. 999.)
         I = I + 1
         READ(2,*) Z(I)
      ENDDO
      DIM1 = I - 1
* store the conductivity depth, frozen conductivity  and density data
* into arrays ...
      I = 1
      READ(2,*)
      READ(2,*) ZK(I), KFR(I), RHO(I)
```

```
      DO WHILE ((ZK(I) .NE. 999.) .AND. (KFR(I) .NE. 999.) .AND.
     +         (RHO(I) .NE. 999.))
         I = I + 1
         READ(2,*) ZK(I), KFR(I), RHO(I)
      ENDDO
      DIM2 = I - 1
* get the heat flow and the Specific Heat ...
      READ(2,*)
      READ(2,*) Q, C
* get the desired search times ...
      I = 1
      READ(2,*)
      READ(2,*) TIMES(I)
      DO WHILE (TIMES(I) .NE. 999.)
         I = I + 1
         READ(2,*) TIMES(I)
      ENDDO
      DIM3 = I - 1
      CLOSE(2)
      RETURN
      END
*
********************** {END OF SUBROUTINE: ZTMEAS} **********************
*
C SUBROUTINE ZTPRED  1989-06-29-PPS
* Module to read depths desired for calculated temperatures. Also reads
* frozen conductivity depth, frozen conductivity, density, heat flow and
* Specific Heat.
*
* Parameters :
*    FILEIN - name of file that contains the measured data (Char).
*    ZP - predicted depth for calculated temperature [m] (Real).
*    ZKFR - frozen conductivity depth [m] (Real).
*    KFR - frozen conductivity [W/(m*K) (Real).
*    RHO - density [Kg/m^3] (Real).
*    Q - heat flow [Wm^-2] (Real).
*    C - Specific Heat (Real).
*    TIMES - desired search times (Real).
*    DIM1, DIM2, DIM3 - array dimensions (Integer).
* Variables :
*    I - index (Integer).
*-----------------------------------------------------------------------
*
      SUBROUTINE ZTPRED(FILEIN,ZP,ZKFR,KFR,RHO,Q,C,TIMES,DIM1,DIM2,DIM3)
*
* declare variables ...
      REAL ZP,ZKFR,KFR,RHO,Q,C,TIMES
      INTEGER I,DIM1,DIM2,DIM3
      CHARACTER FILEIN*12
      DIMENSION ZP(DIM1),ZKFR(DIM2),KFR(DIM2),RHO(DIM2),TIMES(DIM3)
* get the measured data ...
      CALL GETFID('Enter Data File Name -----> ',FILEIN)
      OPEN (2, FILE=FILEIN, STATUS='OLD')
      REWIND(2)
```

```
* store the depth data into an array ...
      I = 1
      READ(2,*)
      READ(2,*) ZP(I)
      DO WHILE (ZP(I) .NE. 999.)
         I = I + 1
         READ(2,*) ZP(I)
      ENDDO
      DIM1 = I - 1
* store the frozen conductivity depth, frozen conductivity, and density
* data into arrays ...
      I = 1
      READ(2,*)
      READ(2,*) ZKFR(I), KFR(I), RHO(I)
      DO WHILE ((ZKFR(I) .NE. 999.) .AND. (KFR(I) .NE. 999.) .AND.
     +         (RHO(I) .NE. 999.))

         I = I + 1
         READ(2,*) ZKFR(I), KFR(I), RHO(I)
      ENDDO
      DIM2 = I - 1
* get the heat flow and the Specific Heat ...
      READ(2,*)
      READ(2,*) Q, C
* get the desired search times ...
      I = 1
      READ(2,*)
      READ(2,*) TIMES(I)
      DO WHILE (TIMES(I) .NE. 999.)
         I = I + 1
         READ(2,*) TIMES(I)
      ENDDO
      DIM3 = I - 1
      CLOSE(2)
      RETURN
      END
*
********************** {END OF SUBROUTINE: ZTPRED} **********************
*
C SUBROUTINE ZTTRAN  1990-05-01-PPS
* Module to read times and water depths desired for transect. Also reads
* frozen conductivity depth, frozen conductivity, density, heat flow and
* Specific Heat.
*
* Parameters :
*    FILEIN - name of file that contains the measured data (Char).
*    ZP - predicted depth for calculated temperature [m] (Real).
*    ZKFR - frozen conductivity depth [m] (Real).
*    KFR - frozen conductivity [W/(m*K) (Real).
*    RHO - density [Kg/m^3] (Real).
*    Q - heat flow [Wm^-2] (Real).
*    C - Specific Heat (Real).
*    TIMES - desired search times (Real).
*    W_DEPTH - desired water depth (Real).
```

```
*     DIM1, DIM2, DIM3 - array dimensions (Integer).
* Variables :
*     I - index (Integer).
*-----------------------------------------------------------------------
*
      SUBROUTINE ZTTRAN(FILEIN,ZP,ZKFR,KFR,RHO,Q,C,TIMES,W_DEPTH,DIM1,
     +                  DIM2,DIM3)
*
* declare variables ...
      REAL ZP,ZKFR,KFR,RHO,Q,C,TIMES,W_DEPTH
      INTEGER I,DIM1,DIM2,DIM3
      CHARACTER FILEIN*12
      DIMENSION ZP(DIM1),ZKFR(DIM2),KFR(DIM2),RHO(DIM2),TIMES(DIM3)
      DIMENSION W_DEPTH(DIM3)
* get the measured data ...
      CALL GETFID('Enter Data File Name -----> ',FILEIN)
      OPEN (2, FILE=FILEIN, STATUS='OLD')
      REWIND(2)
* store the depth data into an array ...
      I = 1
      READ(2,*)
      READ(2,*) ZP(I)
      DO WHILE (ZP(I) .NE. 999.)
         I = I + 1
         READ(2,*) ZP(I)
      ENDDO
      DIM1 = I - 1
* store the frozen conductivity depth, frozen conductivity, and density
* data into arrays ...
      I = 1
      READ(2,*)
      READ(2,*) ZKFR(I), KFR(I), RHO(I)
      DO WHILE ((ZKFR(I) .NE. 999.) .AND. (KFR(I) .NE. 999.) .AND.
     +          (RHO(I) .NE. 999.))

         I = I + 1
         READ(2,*) ZKFR(I), KFR(I), RHO(I)
      ENDDO
      DIM2 = I - 1
* get the heat flow and the Specific Heat ...
      READ(2,*)
      READ(2,*) Q, C
* get the desired search times and the water depth  ...
      I = 1
      READ(2,*)
      READ(2,*) TIMES(I),W_DEPTH(I)
      DO WHILE (TIMES(I) .NE. 999. .AND. W_DEPTH(I) .NE. 999.)
         I = I + 1
         READ(2,*) TIMES(I),W_DEPTH(I)
      ENDDO
      DIM3 = I - 1
      CLOSE(2)
      RETURN
      END
```

```
*
********************** {END OF SUBROUTINE: ZTPRED} **********************
```

APPENDIX III

SAMPLE CALCULATIONS FOR FROZEN THERMAL CONDUCTIVITIES : NERLERK WELLSITE

Data

Temperature Gradient Unfrozen ($T_{unfrozen}$) = 24.5 mK/m (measured from Weaver et al. [8])

Terrestrial Heat Flow (Q) = 49 mW/m^2 (analysis by Majorowicz et al. [18])

Thermal Conductivity Of Ice (K_{ice}) = 2.2 $W \cdot m^{-1} \cdot K^{-1}$

Thermal Conductivity Of Water (K_{water}) = 0.59 $W \cdot m^{-1} \cdot K^{-1}$

Volumetric Porosity (Φ) = 30 % (assumed)

Heat Flow

$$Q = K \cdot T \qquad \text{(eq. 1)}$$

assuming equal heat flow (ie. equilibrium)

$$K_{frozen} \cdot T_{frozen} = K_{unfrozen} \cdot T_{unfrozen}$$

Thermal Conductivity

$$K = K_g^{(1-\Phi)} K_{water}^{\Phi}_{ice} \qquad \text{(eq. 2)}$$

where

K_g = Thermal conductivity due to grains

From equation 1

$$K_{unfrozen} = \frac{Q}{T_{unfrozen}} = \frac{49}{24.5} = 2.00 \ W \cdot m^{-1} \cdot K^{-1}$$

From equation 2

$$K_{unfrozen} = K_g^{(1-\Phi)} \cdot K^{\Phi}_{water} \qquad \text{(eq. 3)}$$

Solving for K_g in equation 3 and substituting the value for Φ and K_{water}

$$K_g = 3.375 \ W \cdot m^{-1} \cdot K^{-1}$$

Also from equation 2

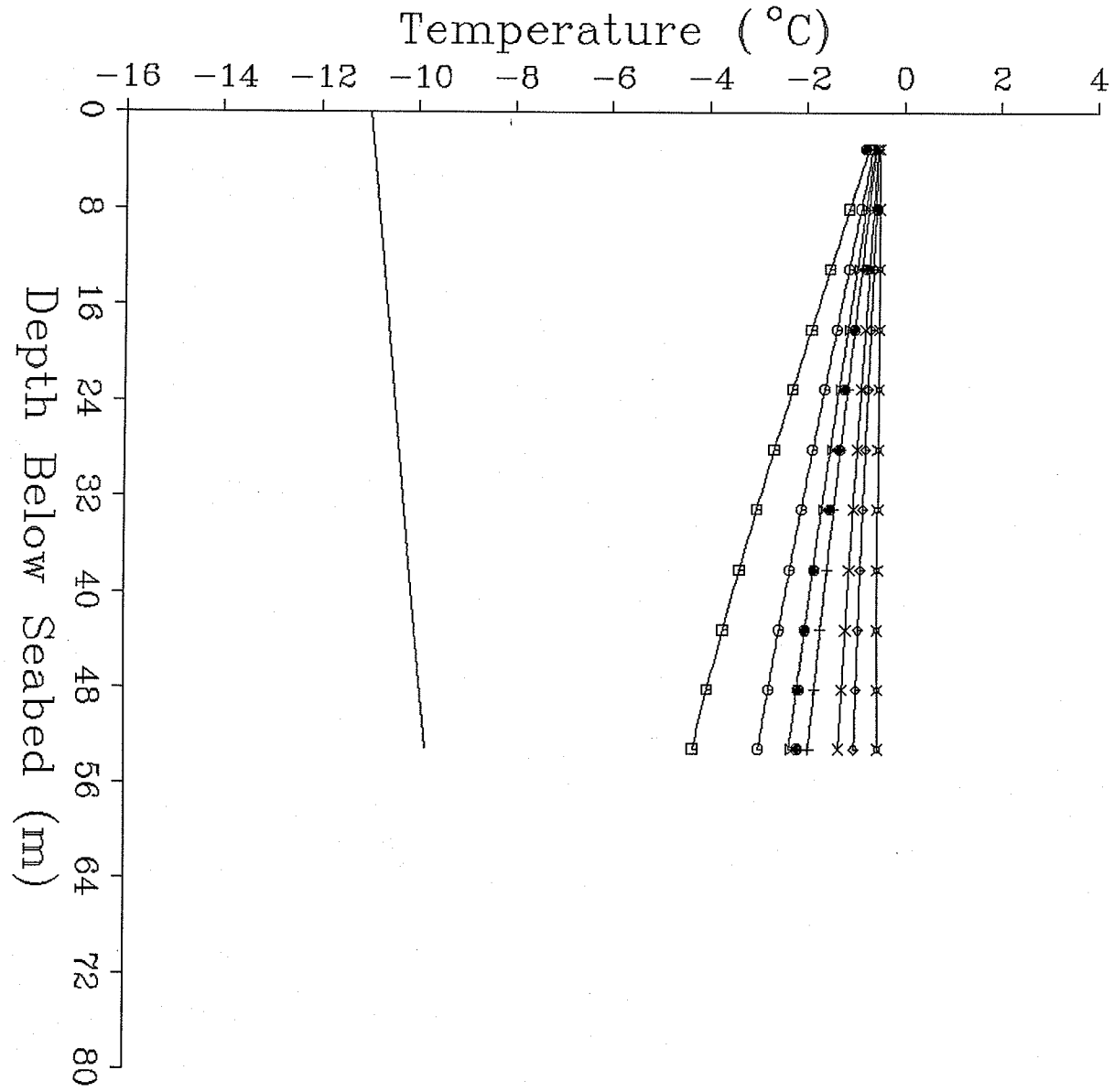
$$K_{frozen} = K_g^{(1-\Phi)} \cdot K^{\Phi}_{ice} \qquad \text{(eq. 4)}$$

Substituting the value for Φ, K_g, and K_{ice}

$$K_{frozen} = 2.968 \ W \cdot m^{-1} \cdot K^{-1}$$

APPENDIX IV

ADDITIONAL MODELLED TEMPERATURE PROFILES



ESSO ANGASAK

Fitting Parameters

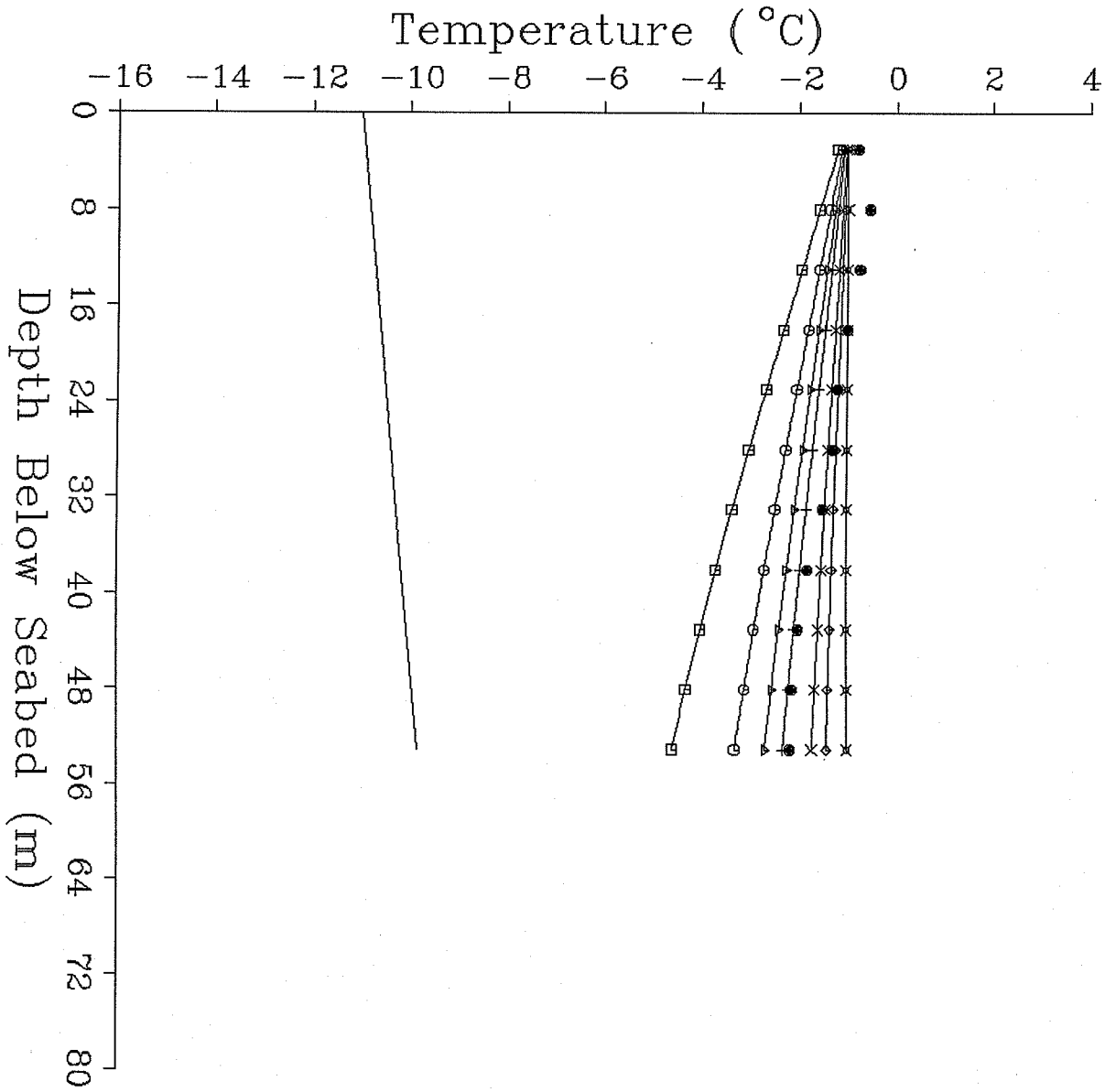
THO = -11.00
THS = -0.50
THB = -0.50

Input File : ANGASAK.DT1

Output File : ANGASAK.OT1

Time Since Inundation [yrs]

□ 100.0
○ 200.0
△ 300.0
+ 400.0
× 700.0
◇ 1000.0
⋈ 3000.0
● Measured Data



ESSO ANGASAK

Fitting Parameters

THO = -11.00
THS = -1.00
THB = -1.00

Input File : ANGASAK.DT1

Output File : ANGASAK.OT3

Time Since Inundation [yrs]

□ 100.0
◯ 200.0
△ 300.0
+ 400.0
× 700.0
◇ 1000.0
⋈ 3000.0
● Measured Data

File : ANGASAK.OT4

Date : 1990- 5- 3
Time : 12:01:02

Depth Measured (Z) in [m]	Temperature Measured (T) in [C]	Gradient Anomaly in [mK/m]
13.00	-.760	N/A
18.00	-1.020	-.6617E+02
23.00	-1.200	-.5117E+02
28.00	-1.310	-.5217E+02
33.00	-1.500	-.7317E+02
38.00	-1.820	-.7217E+02
43.00	-2.000	-.5417E+02
48.00	-2.140	-.3917E+02
53.00	-2.170	N/A

Frozen Conductivity Depth (ZKFR) in [m]	Frozen Conductivity (KFR) in [W/(m*K)]	Density (RHO) in [Kg/m^3]
.00	4.00	2500.00
10.00	4.00	2500.00
20.00	4.00	2500.00
30.00	4.00	2500.00
40.00	1.80	2500.00
50.00	2.50	2500.00
55.00	1.80	2500.00

Gradient (GRADM) ----------> -17.00 [mK/m]
Equilibrium Gradient (GRADEQM) -----> 22.17 [mK/m]
Heat Flow (Q) ----------> .07 [W/m^2]
Specific Heat (C) ----------> 1150.00 [J/(Kg*K)]

DERIVED PHYSICAL PROPERTIES

Average Thermal Conductivity (KFRAVE) -----> 3.16 [W/(m*K)]
Average Density (RHOAVE) ----------> 2500.00 [Kg/m^3]
Thermal Diffusivity (ALPHA) ----------> .1098E-05 [m^2/s]

FITTING PARAMETERS

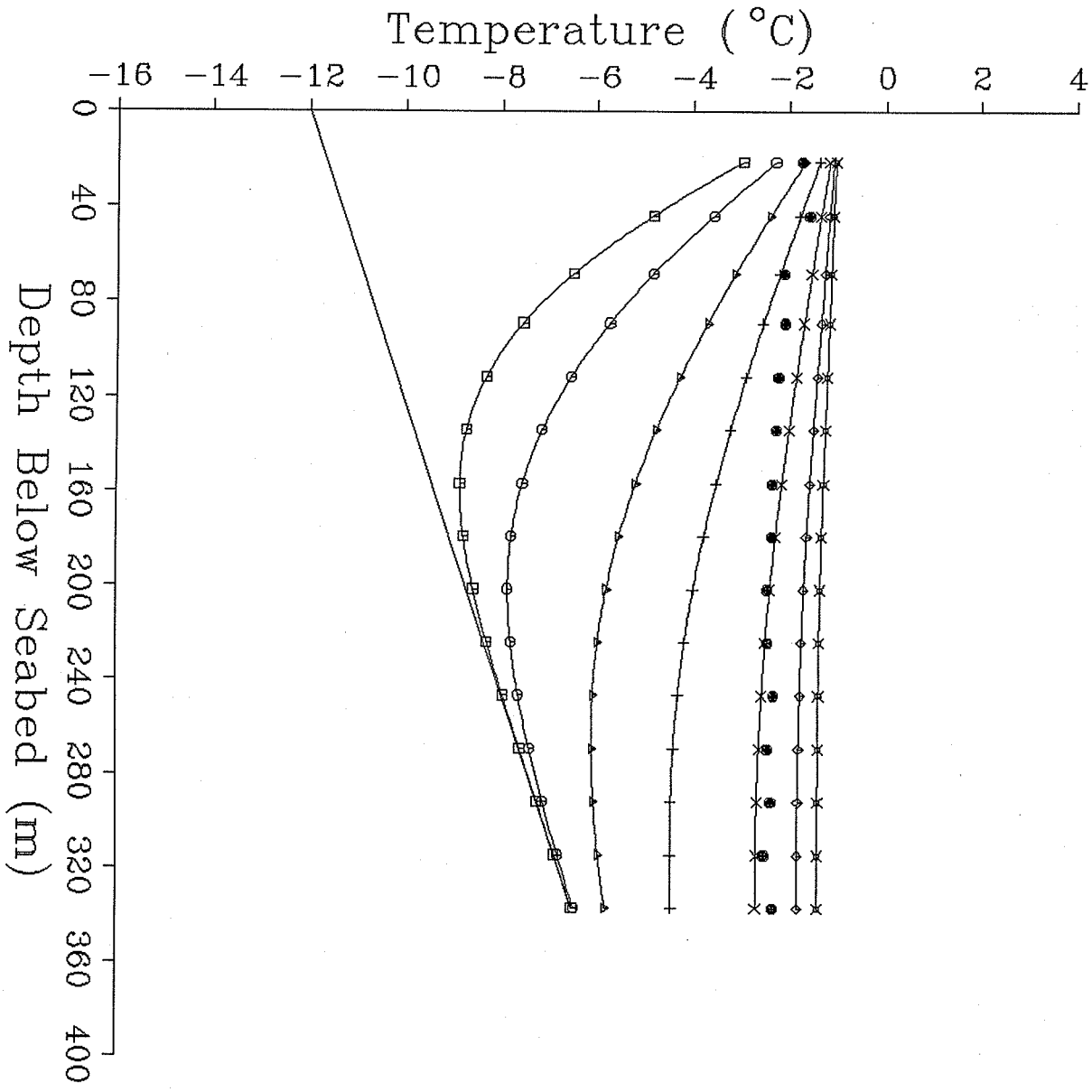
Temperature Change Upon Shoreline Movement (A) -----> 12.00 [K]
Distance From Present Shoreline (X) ----------> 2000.00 [m]

Depth (Z) / Time [yrs] ----->	200.0	243.8	297.2	362.3	441.6
	Gradient Anomaly [mK/m]				
13.00	-.8086E+02	-.7331E+02	-.6646E+02	-.6024E+02	-.5459E+02
18.00	-.8040E+02	-.7298E+02	-.6621E+02	-.6005E+02	-.5446E+02
23.00	-.7981E+02	-.7254E+02	-.6588E+02	-.5981E+02	-.5427E+02
28.00	-.7908E+02	-.7199E+02	-.6548E+02	-.5951E+02	-.5405E+02
33.00	-.7821E+02	-.7134E+02	-.6499E+02	-.5915E+02	-.5378E+02
38.00	-.7722E+02	-.7060E+02	-.6443E+02	-.5873E+02	-.5347E+02
43.00	-.7610E+02	-.6976E+02	-.6380E+02	-.5826E+02	-.5311E+02
48.00	-.7486E+02	-.6882E+02	-.6310E+02	-.5773E+02	-.5272E+02
53.00	-.7351E+02	-.6780E+02	-.6233E+02	-.5715E+02	-.5229E+02

Depth (Z) / Time [yrs] ----->	538.4	656.3	800.0
	Gradient Anomaly [mK/m]		

13.00	-.4947E+02	-.4483E+02	-.4061E+02
18.00	-.4937E+02	-.4475E+02	-.4056E+02
23.00	-.4923E+02	-.4465E+02	-.4048E+02
28.00	-.4907E+02	-.4452E+02	-.4039E+02
33.00	-.4886E+02	-.4437E+02	-.4028E+02
38.00	-.4863E+02	-.4420E+02	-.4015E+02
43.00	-.4837E+02	-.4400E+02	-.4000E+02
48.00	-.4808E+02	-.4379E+02	-.3984E+02
53.00	-.4775E+02	-.4354E+02	-.3966E+02

Measured Gradient Anomaly in [mK/m]	Time in [yrs]
-.6617E+02	297.6
-.5117E+02	497.8
-.5217E+02	474.8
-.7317E+02	230.9
-.7217E+02	232.3
-.5417E+02	423.5



GULF AMAULIGAK

Fitting Parameters

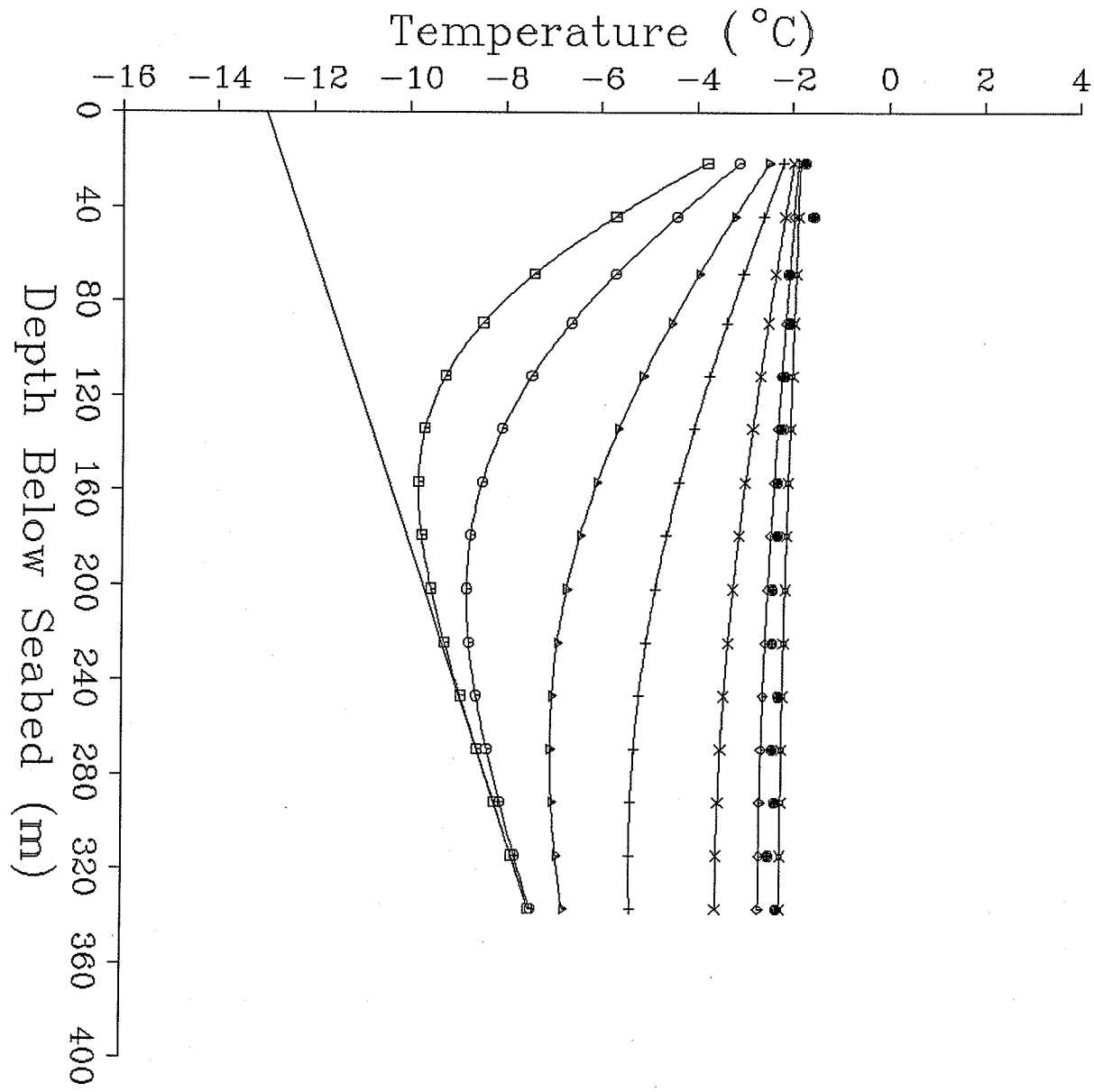
THO = −12.00
THS = −1.00
THB = −1.00

Input File : AMAUL.DT1

Output File : AMAUL.OT1

Time Since Inundation [yrs]

□ 100.0 × 2000.0
○ 200.0 ◇ 3000.0
△ 500.0 ⋈ 4000.0
+ 1000.0 ● Measured Data



GULF AMAULIGAK

Fitting Parameters

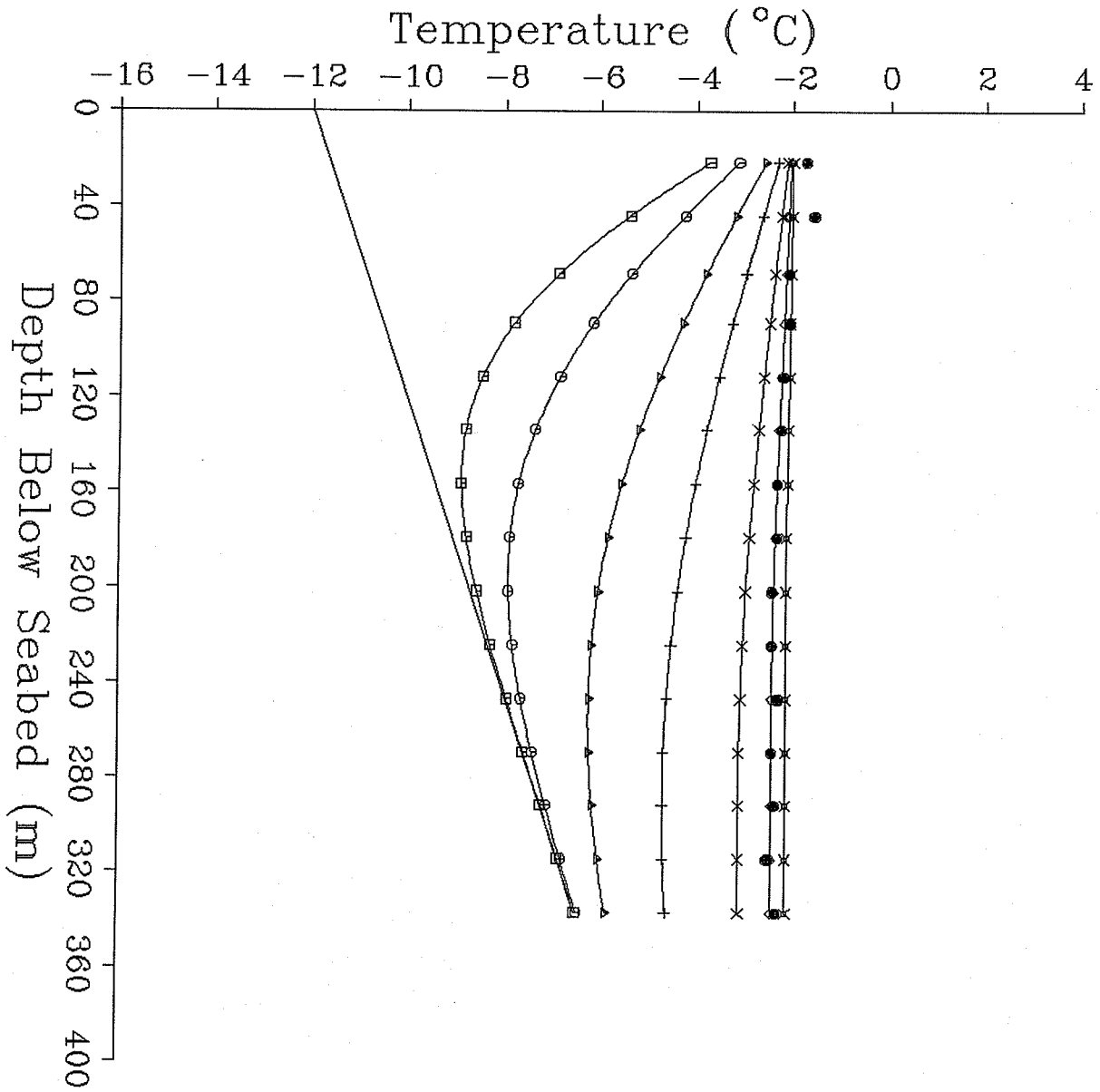
THO = −13.00
THS = −1.80
THB = −1.80

Input File : AMAUL.DT1

Output File : AMAUL.OT3

Time Since Inundation [yrs]

□ 100.0
○ 200.0
△ 500.0
+ 1000.0
× 2000.0
◇ 3000.0
⋈ 4000.0
● Measured Data



GULF AMAULIGAK

Fitting Parameters

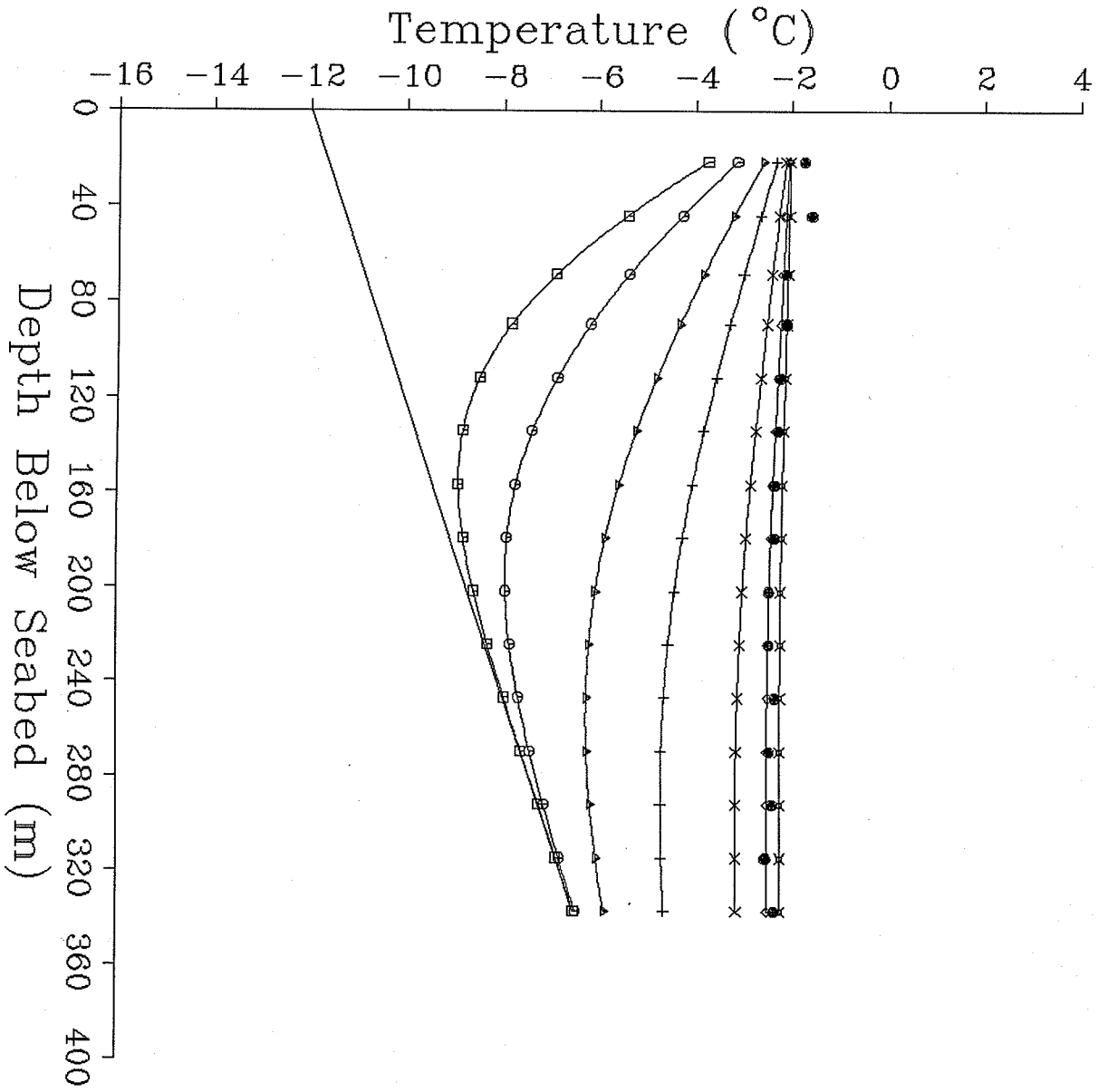
THO = −12.00
THS = −2.00
THB = −1.80

Input File : AMAUL.DT1

Output File : AMAUL.OT4

Time Since Inundation [yrs]

□ 100.0
○ 200.0
▷ 500.0
+ 1000.0
× 2000.0
◇ 3000.0
⊠ 4000.0
● Measured Data



GULF AMAULIGAK

Fitting Parameters

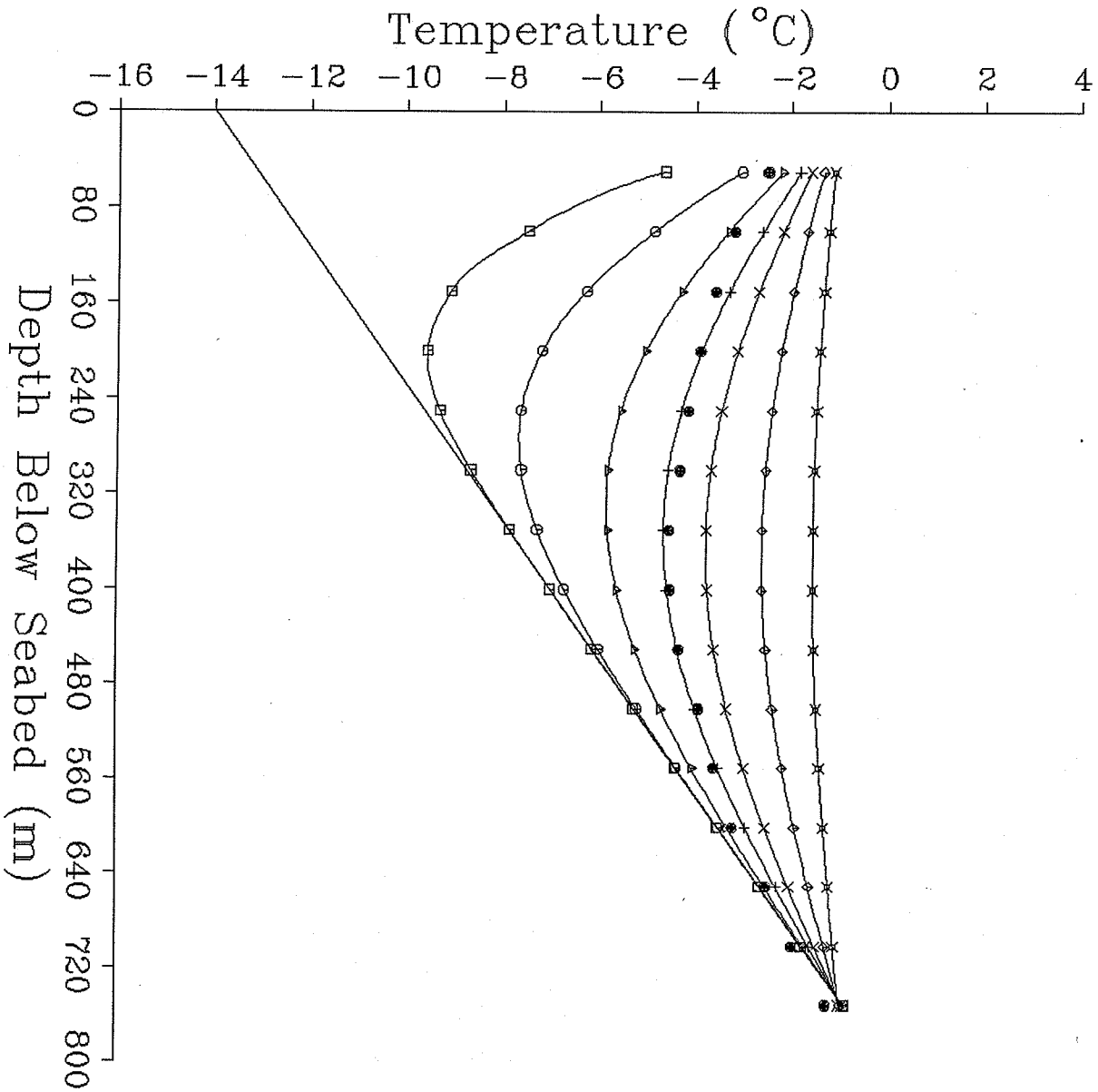
THO = -12.00
THS = -2.00
THB = -2.00

Input File : AMAUL.DT1

Output File : AMAUL.OT5

Time Since Inundation [yrs]

□ 100.0
○ 200.0
△ 500.0
+ 1000.0
× 2000.0
◇ 3000.0
⊠ 4000.0
● Measured Data



KOAKOAK

Fitting Parameters

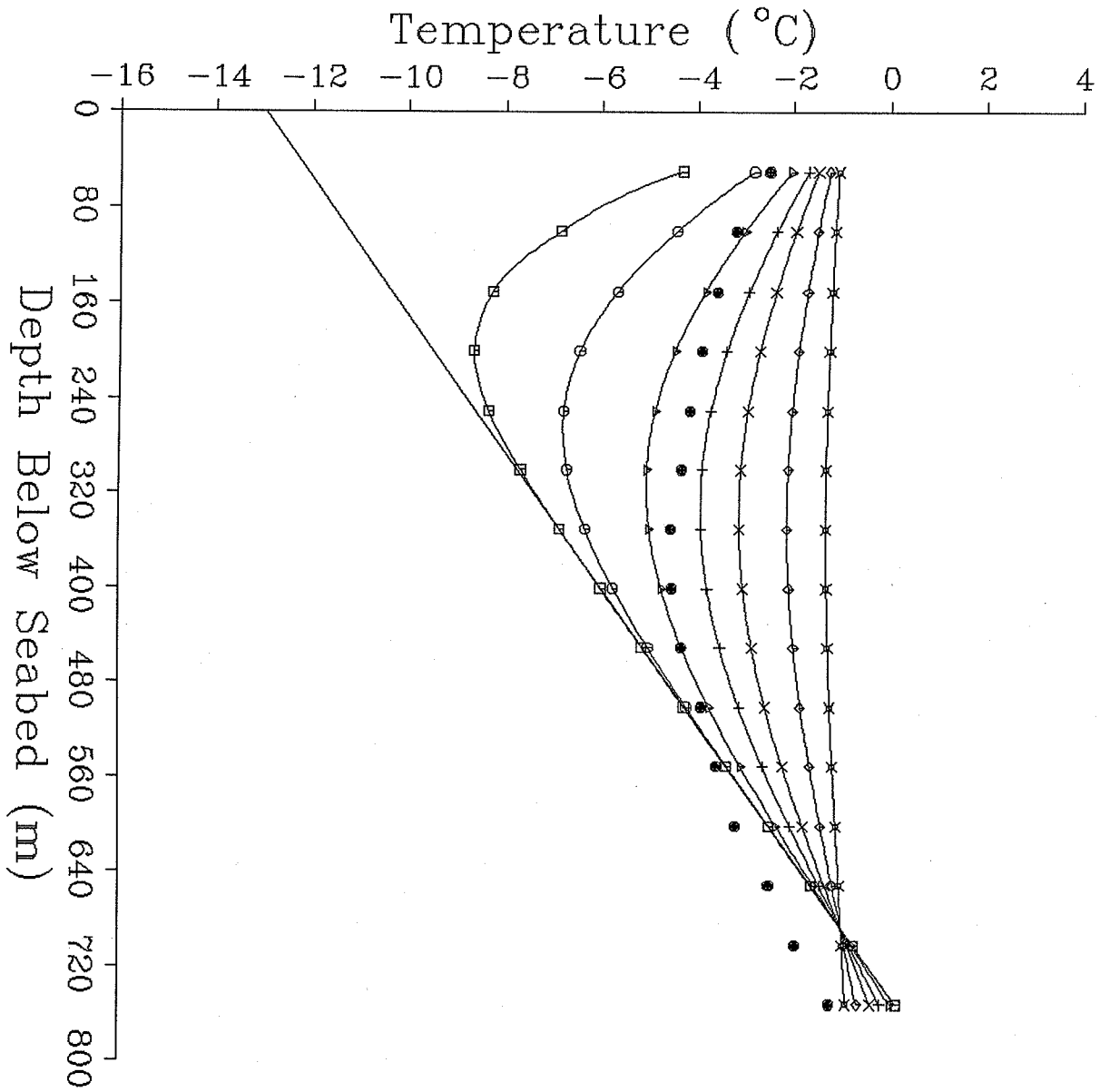
THO = −14.00
THS = −1.00
THB = −1.00

Input File : KOAKOAK.DT1

Output File : KOAKOAK.OT2

Time Since Inundation [yrs]

200.0
500.0
1000.0
1500.0
2000.0
3000.0
5000.0
Measured Data



KOAKOAK

Fitting Parameters

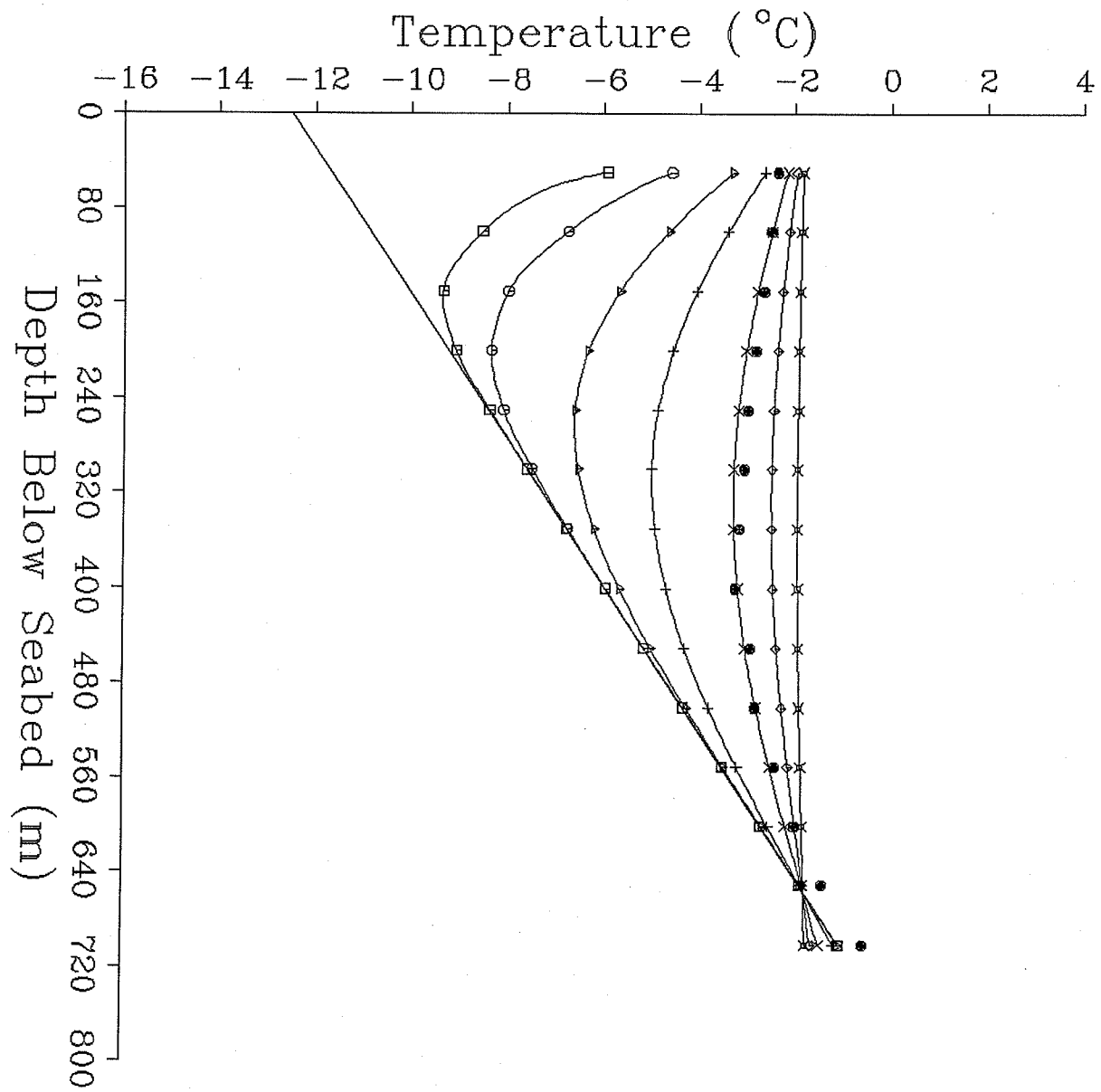
THO = -13.00
THS = -1.00
THB = -1.00

Input File : KOAKOAK.DT1

Output File : KOAKOAK.OT1

Time Since Inundation [yrs]

□ 200.0
○ 500.0
△ 1000.0
+ 1500.0
× 2000.0
◇ 3000.0
⋈ 5000.0
● Measured Data



NERLERK

Fitting Parameters

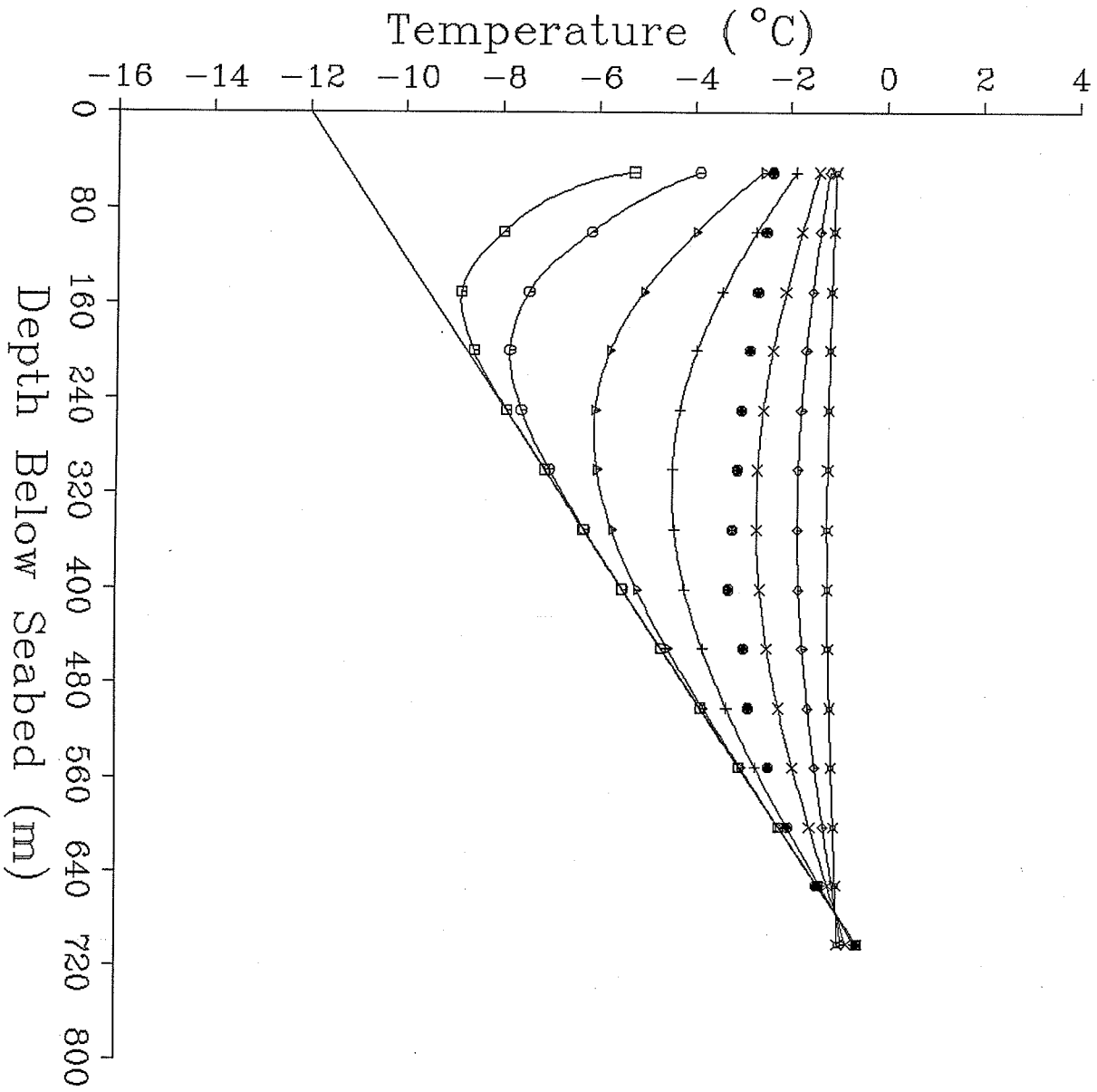
THO = −12.50
THS = −1.80
THB = −1.80

Input File : NERLERK.DT1

Output File : NERLERK.OT3

Time Since Inundation [yrs]

□ 100.0
◯ 200.0
▽ 500.0
+ 1000.0
× 2000.0
◇ 3000.0
⋈ 5000.0
● Measured Data



NERLERK

Fitting Parameters

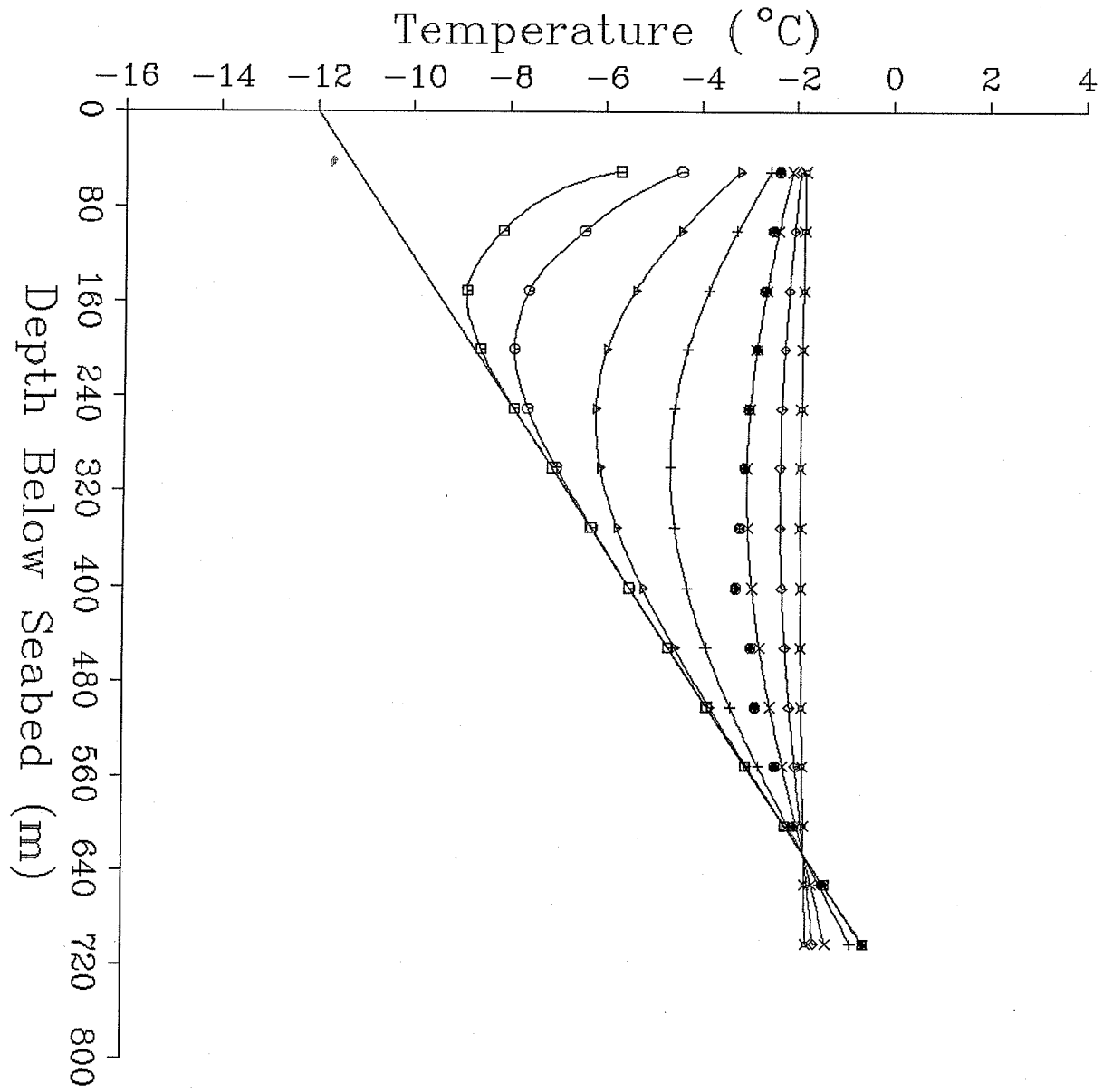
THO = −12.00
THS = −1.00
THB = −1.00

Input File : NERLERK.DT1

Output File : NERLERK.OT1

Time Since Inundation [yrs]

100.0
200.0
500.0
1000.0
2000.0
3000.0
5000.0
Measured Data



NERLERK

Fitting Parameters

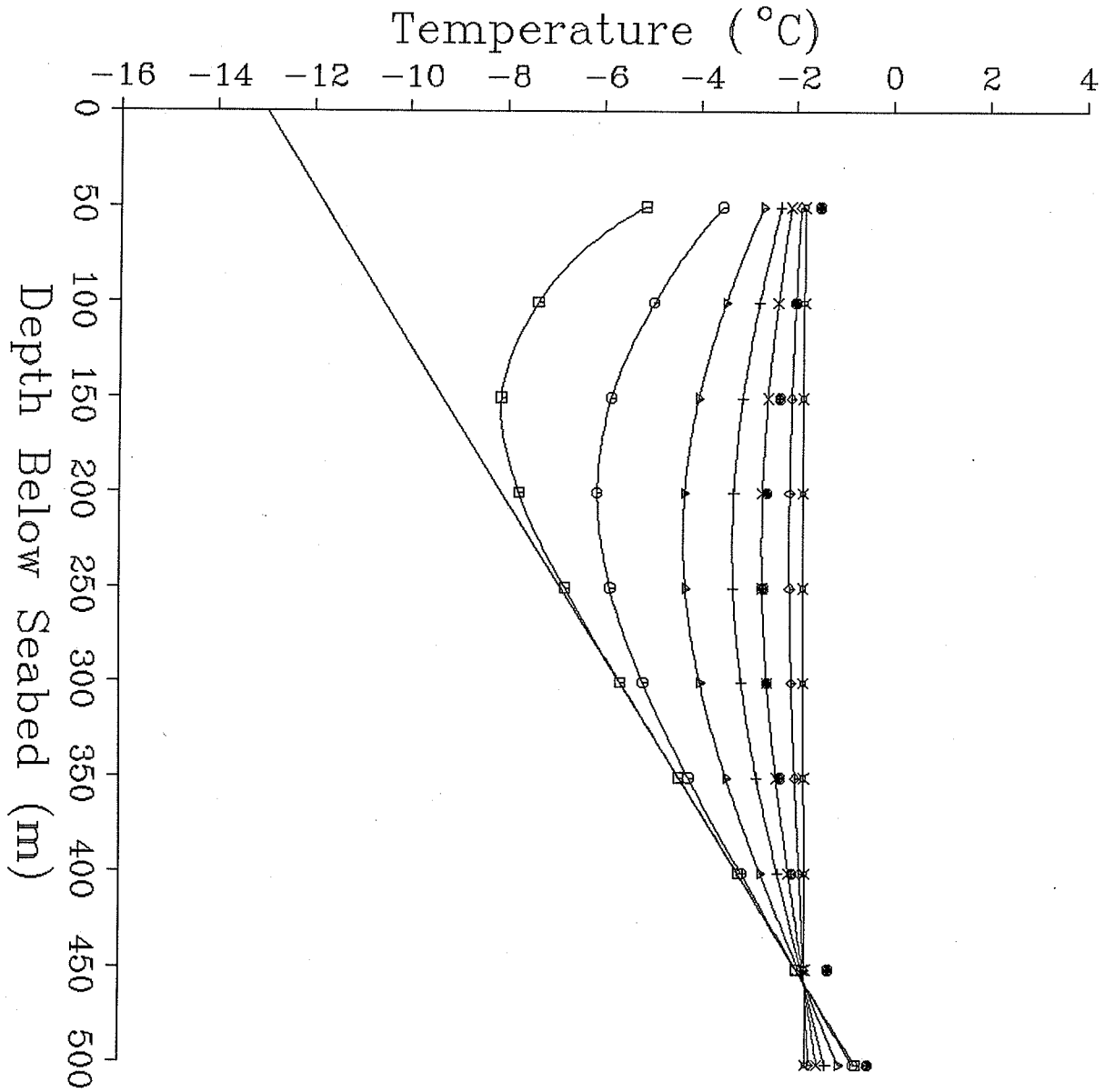
THO = -12.00
THS = -1.80
THB = -1.80

Input File : NERLERK.DT1

Output File : NERLERK.OT2

Time Since Inundation [yrs]

□ 100.0 × 2000.0
○ 200.0 ◇ 3000.0
△ 500.0 ⊠ 5000.0
+ 1000.0 ● Measured Data



KOPANOAR

Fitting Parameters

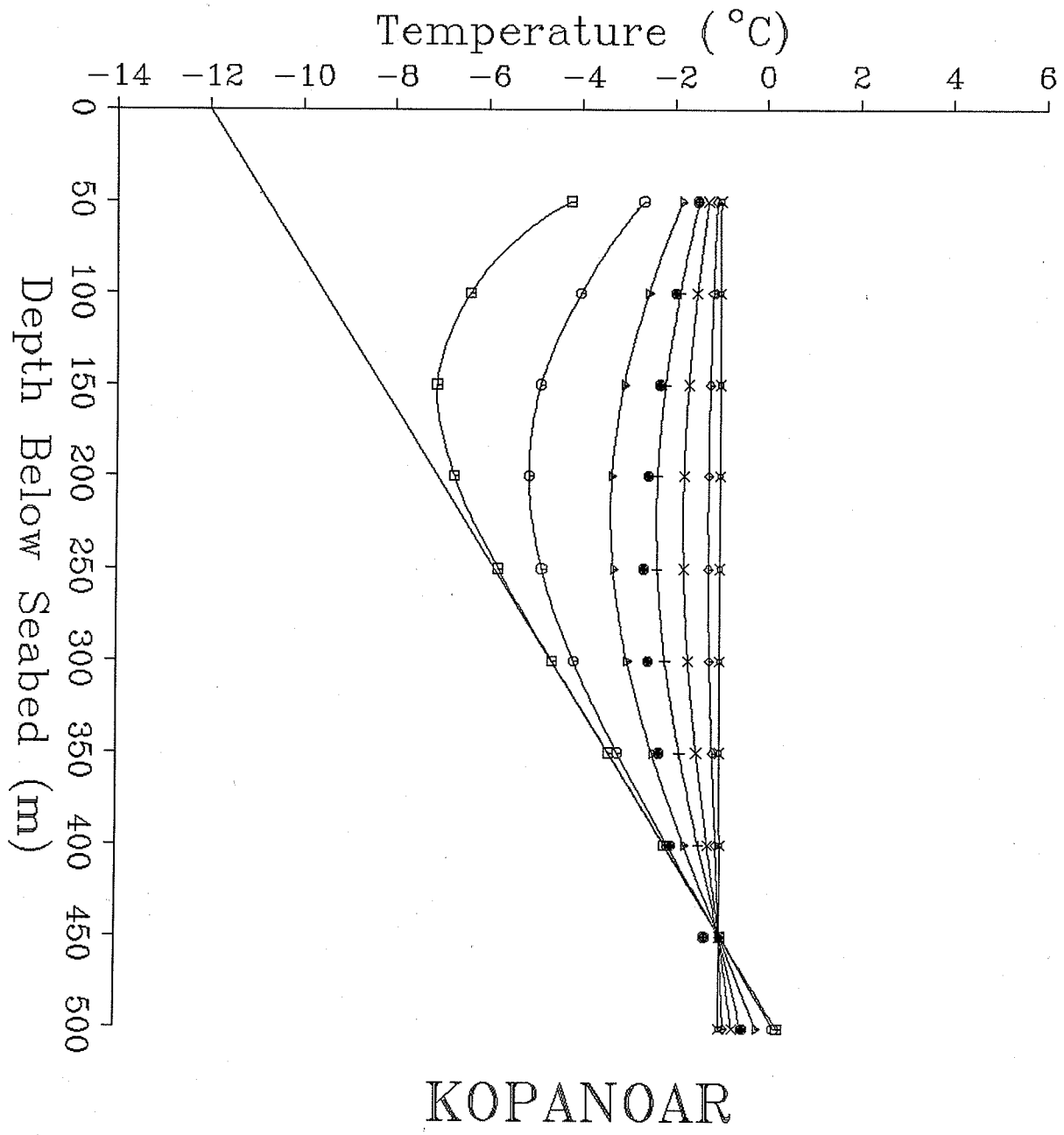
THO = -13.00
THS = -1.80
THB = -1.80

Input File : KOPAN.DT1

Output File : KOPAN.OT4

Time Since Inundation [yrs]

□ 200.0 × 2000.0
○ 500.0 ◇ 3000.0
△ 1000.0 ⋈ 5000.0
+ 1500.0 ● Measured Data



KOPANOAR

Fitting Parameters

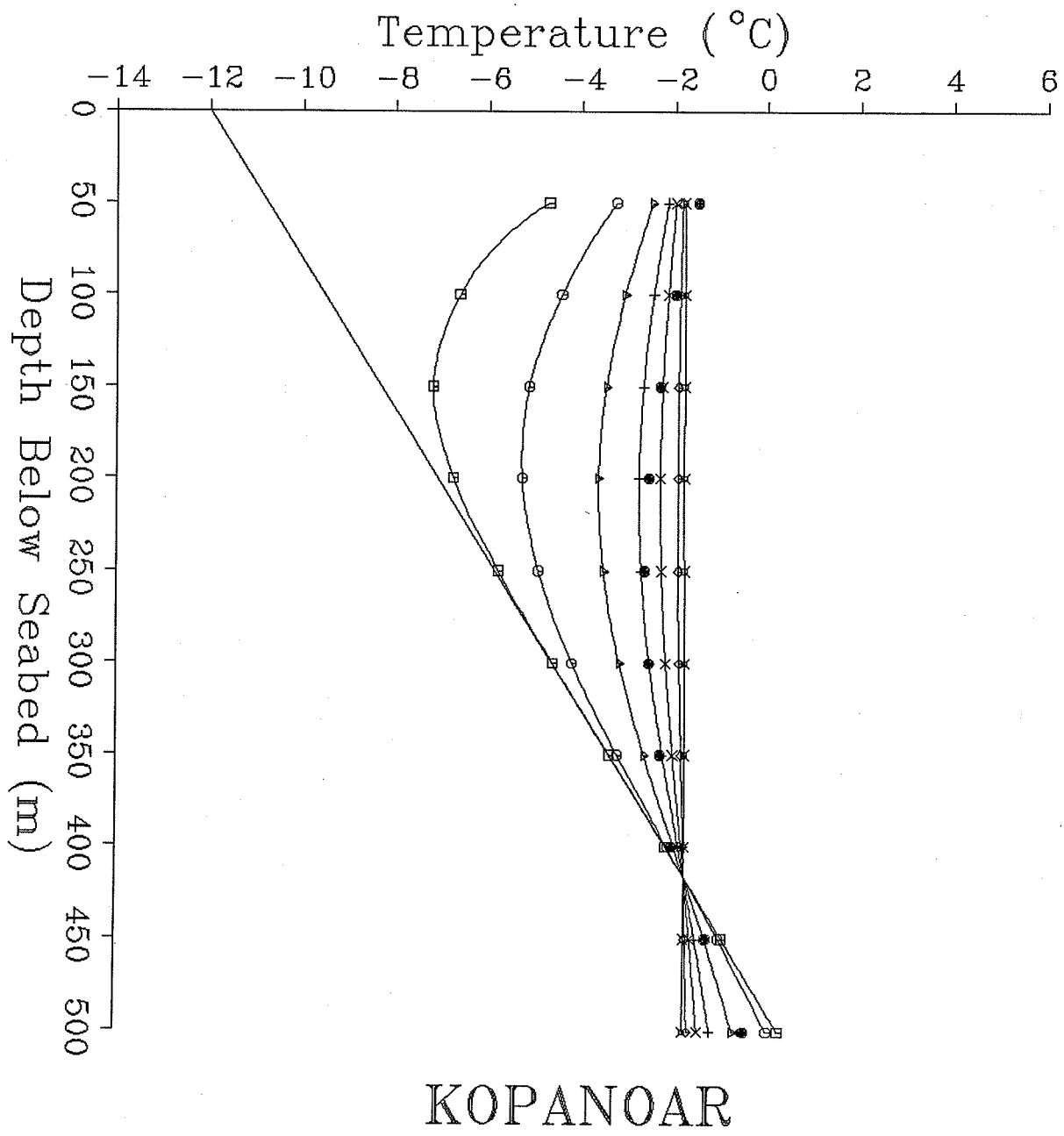
THO = −12.00
THS = −1.00
THB = −1.00

Input File : KOPAN.DT1

Output File : KOPAN.OT1

Time Since Inundation [yrs]

□ 200.0
○ 500.0
△ 1000.0
+ 1500.0
× 2000.0
◇ 3000.0
⊠ 5000.0
● Measured Data



KOPANOAR

Fitting Parameters

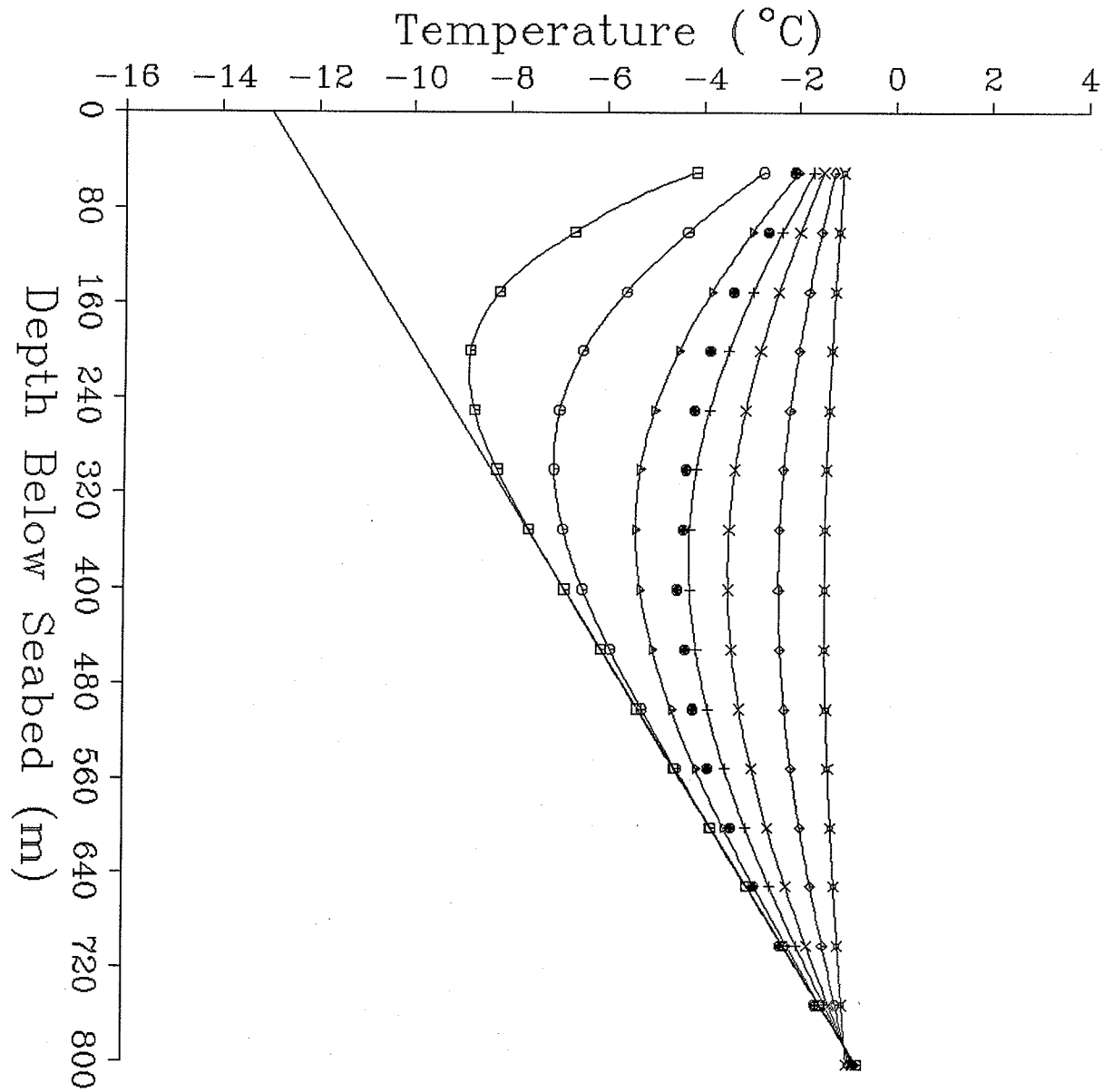
THO = -12.00
THS = -1.80
THB = -1.80

Input File : KOPAN.DT1

Output File : KOPAN.OT2

Time Since Inundation [yrs]

200.0
500.0
1000.0
1500.0
2000.0
3000.0
5000.0
Measured Data



UKALERK

Fitting Parameters

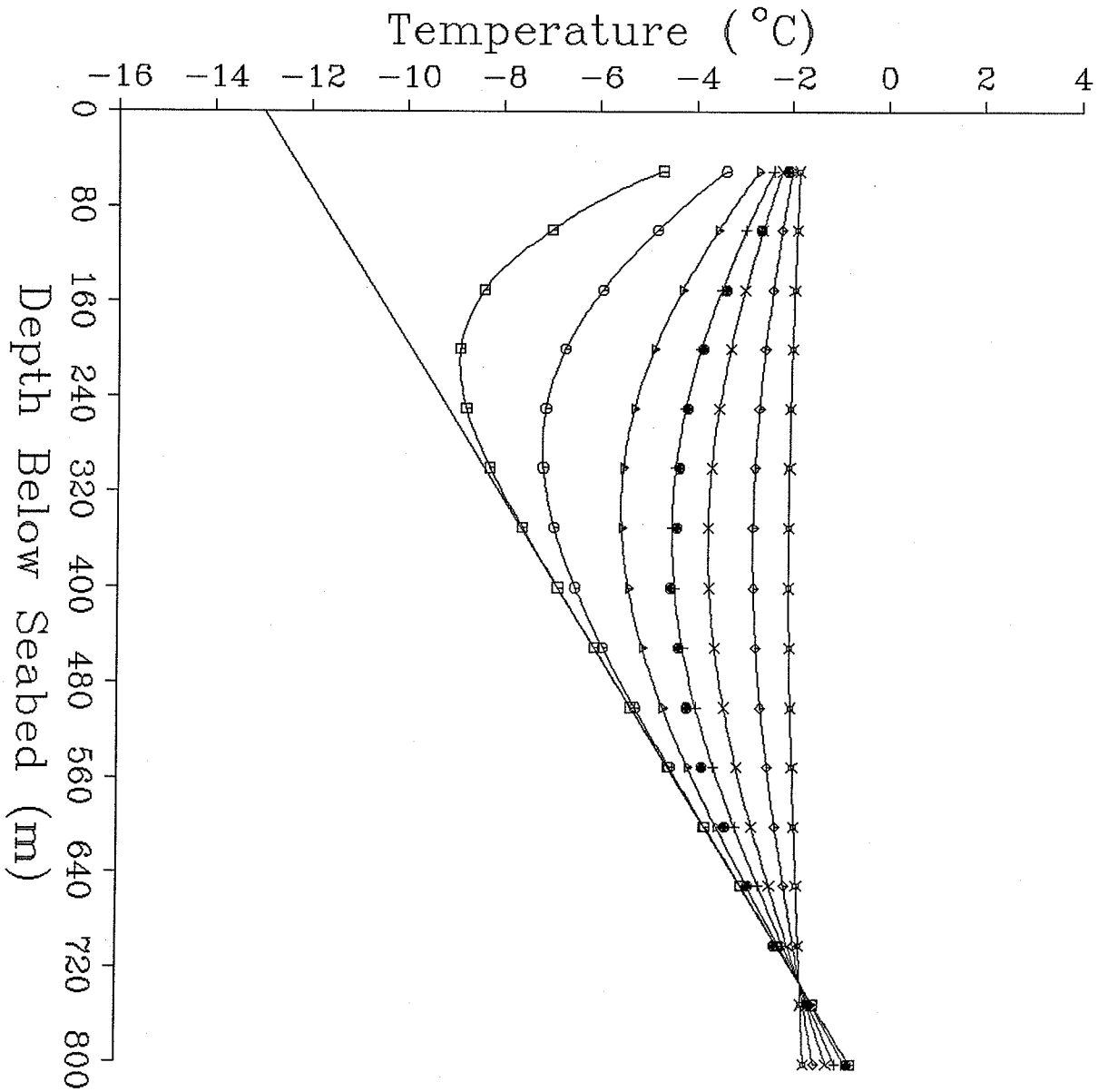
THO = -13.00
THS = -1.00
THB = -1.00

Input File : UKALERK.DT1

Output File : UKALERK.OT2

Time Since Inundation [yrs]

200.0
500.0
1000.0
1500.0
2000.0
3000.0
5000.0
Measured Data



UKALERK

Fitting Parameters

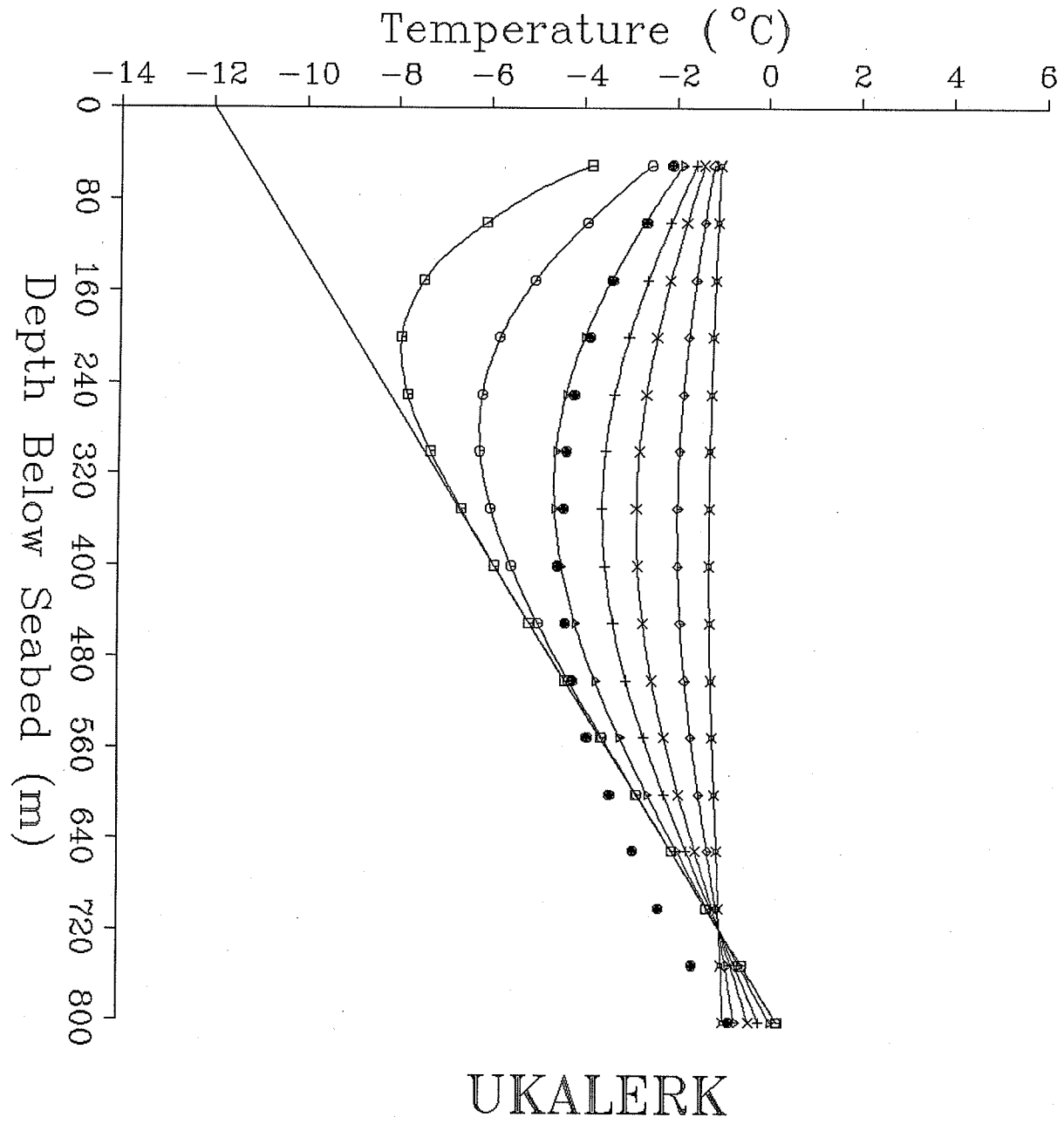
THO = −13.00
THS = −1.80
THB = −1.80

Input File : UKALERK.DT1

Output File : UKALERK.OT3

Time Since Inundation [yrs]

200.0
500.0
1000.0
1500.0
2000.0
3000.0
5000.0
Measured Data



UKALERK

Fitting Parameters

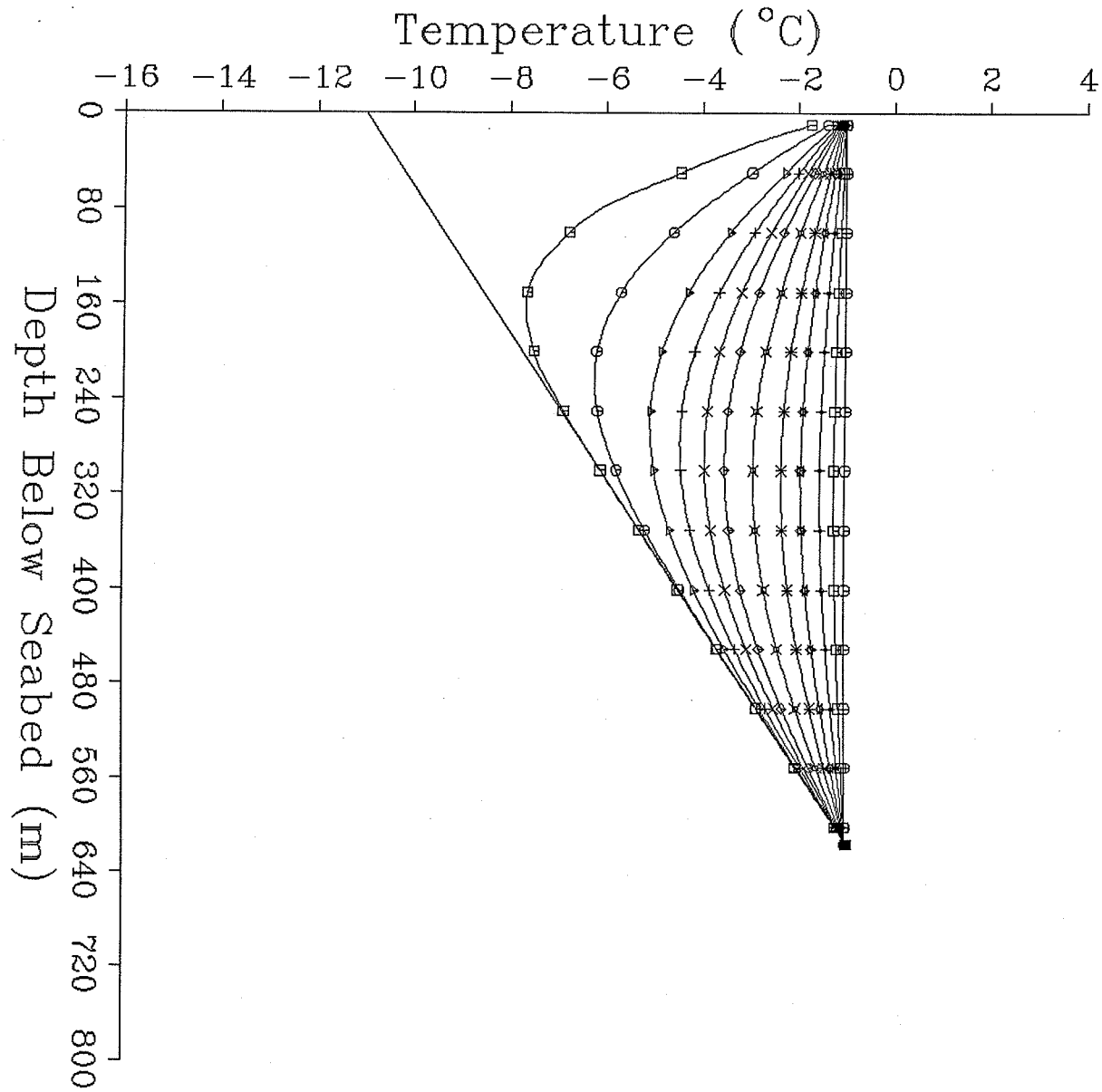
THO = −12.00
THS = −1.00
THB = −1.00

Input File : UKALERK.DT1

Output File : UKALERK.OT1

Time Since Inundation [yrs]

□ 200.0	× 2000.0
○ 500.0	◇ 3000.0
△ 1000.0	⋈ 5000.0
+ 1500.0	● Measured Data



OFFSHORE TRANSECT

Fitting Parameters

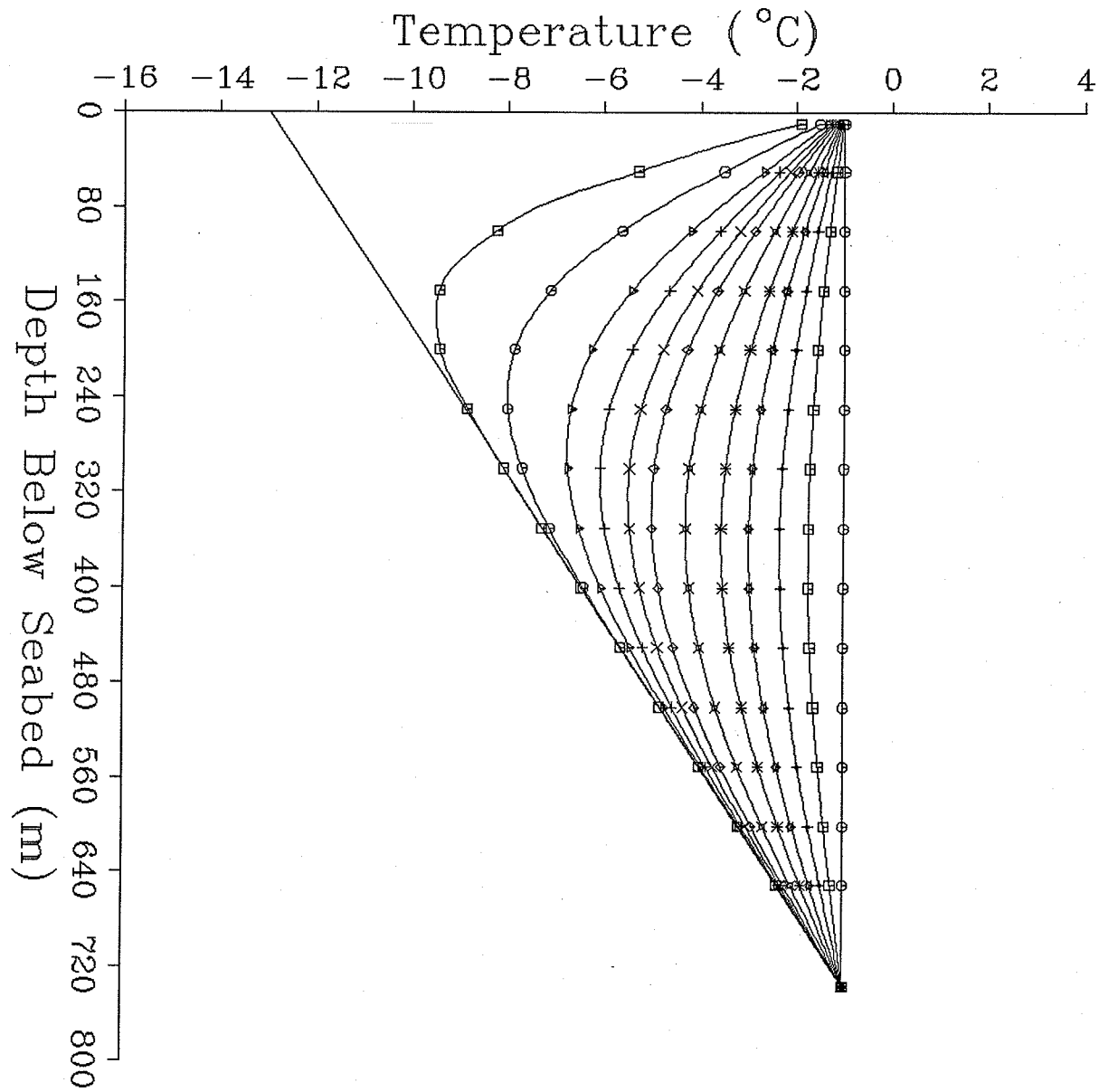
THO = −11.00
THS = −1.00
THB = −1.00

Input File : TRANSECT.DT1

Output File : TRANSECT.OT1

Water Depth [m]

1.0	8.0
2.0	10.0
3.0	12.0
4.0	15.0
5.0	20.0
6.0	70.0



OFFSHORE TRANSECT

Fitting Parameters

THO = −13.00
THS = −1.00
THB = −1.00

Input File : TRANSECT.DT3

Output File : TRANSECT.OT3

Water Depth [m]

1.0	8.0
2.0	10.0
3.0	12.0
4.0	15.0
5.0	20.0
6.0	70.0