Int. J. Pure Appl. Sci. Technol., 25(2) (2014), pp. 46-62

International Journal of Pure and Applied Sciences and Technology

ISSN 2229 - 6107

Available online at www.ijopaasat.in

Research Paper

Computation of Low Velocity Layer Correction Parameters by Computer Models

A.O. Fajana[1,*], M.A. Oladunjoye[2] and A.I. Olayinka[2]

[1] Department Geophysics, Federal University Oye, Nigeria

[2] Department of Geology, University of Ibadan, Nigeria

* Corresponding author, e-mail: (akindeji.fajana@fuoye.edu.ng)

(Received: 2-11-14; Accepted: 8-12-14)

Abstract: *Near-surface low velocity layer is a critical zone in seismic reflection surveys because shots fired within this zone tend to yield low-frequency data, as the loose dirt does not transmit high frequencies adequately causing time delay to rays passing through it, thereby obscuring the reflection records. A computer program was designed in Visual Basic Computer programming language to carry out computation of low-velocity layer correction parameters used in statics computation and to determine energy source drilling depth. Such analysis by computer program minimizes slowness, time wastage and make results more reliable. The method of analysis used includes: Algorithms design and Visual Basic codes generation, employing the Least-Squares approximation, stepwise multi-linear regression, computing velocities from the reciprocals of the slopes of the lines and determining thickness for multi-layer cases. The program was debugged and test-run on Microsoft Windows using a set of thirty near-surface seismic velocity data acquired by seismic refraction technique within the South-Central Niger Delta. Based on the sampling density of thirty (30) refraction surveys, the average consolidated layer velocity of the area is 1790m/s, the weathered layer has a velocity gradient of 8.4(m/s)/m and an average velocity of 337m/s. The shot drilling depth is 3.7m in order to remove the effect of the low velocity surface layer and reduces all reflection times to a common height datum. The easy and quick nature of implementation and the closeness of the result to those of other researchers on the weathered layer in the Niger Delta indicate that the program, FAJSEIS is a reliable measure for the analysis of the near-surface seismic data.*

Keywords: Velocity analysis, Modelling, Acoustics.

Background of the Study

The analysis of near- surface velocity data in the Niger Delta has become indispensable in the search for oil and gas, and underground water. Refraction surveys are widely used to study the water table and, for engineering purposes, the poorly consolidated layers near the ground surface, and in determining near-surface corrections for deep reflection traces (Milson, 2003; Whiteley, 1994).It utilizes seismic energy that returns to the surface after traveling through the ground along refracted ray paths in form of seismic waves (Kearey and Brooks, 1991).

In these days of fast computers, powerful softwares have been extensively used by the major oil exploration companies. However, the softwares available in these companies remained classified and not readily accessible to users outside the company. In line with this trend, one of such computer programs has been developed in this work in Visual Basic Computer programming language to carry out the computation of low velocity layer correction parameters.

Generally, seismic reflection surveys are conducted in the oil industry to determine commercially viable hydrocarbon reservoirs but in most cases reflection records are obscured by wave behaviours in weathering layer. This false indication of significant structural relief features on underlying reflectors being portrayed on seismic sections are caused by near-surface irregularities which have the effect of shifting reflection events on adjacent traces out of the correct time relationships.

This study also entailed application of the developed software to analyze sets of seismic refraction data from Niger- Delta to delineate the low velocity layers in terms of depth, thickness and velocity useful in static correction of reflection time data of the area, to predict depth to which energy sources can be buried and attempt the mapping of the water table over the study area.

Goal and Objectives

The objective of this study is to develop computer models to help geoscientists to reduce the labor required to characterize the near-surface low velocity layer in terms of velocity and thickness.
Other achievable goals are:

- Determination of the average weathering velocity, $V_{w.}$
- Determination of the average weathering thickness, $D_{w.}$
- Computation of the average sub-weathering layer velocity, $V_{sw.}$
- To investigate the variation of weathering layer velocity with depth of burial and to predict the depth to which energy source should be buried.

Scope and Methodology

The scope and methodology of this research are summarized as follows:

- Developing of the underpinning mathematical relationships for seismic refraction models
- Program Composition
- Algorithm Design
- Building a Visual Basic Application
  Step 1 Draw the interface
  Step 2 Set Properties
  Step 3 Write the events code
- Debugging
- Testing and storage
- Maintenance
- Generating the isovelocity and the isopach maps for the various layers defined.

There are difference methods of investigation the low velocity layer. The choice of any of these often depends on factors such as cost, nature of crew available, previous geological information, efficiency and accuracy of the method. For this study, data from the upholesurvey and short spread refraction methods were used.

Uphole surveys are the most direct measures of near surface properties (Knox, 1967, Rogers, 1981). The uphole method has the advantage of further resolving thin beds (hidden layers). (Marsden, 1993), thus making interpretation less problematic. Shots are fired at varying depths in a deep drilled hole, arrival times to detectors spread on the surface near the shot hole were recorded. In measuring these arrivals times, the first breaks were picked as consistently as possible.

Using FAJSEIS, the arrival times were plotted against their corresponding depths, thus giving the time-depth plot. The time-depth plot allows interpretation and evaluation of the weathering velocity, thickness, and the sub-weathering velocity respectively.

Review of Previous Work

Various oil and seismic companies and geoscience researchers have worked successfully on the Niger Delta basin and assertion made abounds in literature but few are on mapping near-surface low velocity layer (seismic layer) in the basin.

The work of Alaminiokuma et al (2007) corroborated the presence of this near-surface low velocity layer in the south- central Niger Delta. Akpabio and Onwusiri (2004) computed low velocity layer (LVL) correction parameters in parts of western Niger Delta, (OML 62) and stated that the weathering velocity ranges from 210 to 994m/s with variation in thickness from 2.8m to 52.8m.

Okolie et al (2007) worked on 4-D hydrocarbon prospecting in parts of Imo state. They used uphole survey to show a 3- layer zone which comprises weathering, sub-weathering and consolidated layers with the weathering layer velocity and thickness ranging from 340-1000m/s and 2.0m- 3.5m, respectively. They stated that the velocity of the consolidated layer falls within the range used for static correction on the reflection records and further predicted weathering/static corrections elimination at about 3.0m deep in the field and similar fields.

Oladunjoye (1999) recommended the integration of seismic interpretation with well logs in mapping stratigraphic units so as to detect lateral facies changes and vertical lithological transition in the Niger Delta Basin.

Uko et al (1992) conducted refraction experiments over parts of the east-central Niger Delta in order to investigate the weathering structure. They made use of velocity and depth calculated from critically refracted arrivals using flat-layer models. Their result revealed that the thickness of the low velocity weathered layer in the region is highly variable, from between about 2.9m and 45.5m, with about 20.0m on a regional average. An average compressional-wave velocity of about 500m/s and 1732m/s was observed for the weathered layer and the refractor beneath it.

Seismic Refraction Theory

Seismic refraction survey is a geophysical method to map geologic structures by using head waves. Head waves involve energy which enters a high velocity medium near a critical angle and travels in the high – velocity medium nearly parallel to the refractor surface. The main quantity measured in the seismic refraction method is the time between the initiation of the seismic wave and the first arrival of energy along the path between shot point and detectors. The first arrivals are evidences of the fastest traveling waves which are longitudinal waves. By observing first arrivals for a variety of shot detector distance, a time-distance curve can be constructed. The time-distance relations can be analyzed in term of the variation with distance of minimum time paths. Then the elastic discontinuities can be

deduced from this variation, and can be interpreted in terms of the nature, depth, and attitude of geologic units below the surface.

The objective is to determine the arrival times of the head waves to map the depth to the refractors in which they travel. It is a faster method that often finds application in locating the ground water table, determining depth to bedrock and configuration, mapping near-surface poorly consolidated/slow velocity layer, among others uses.

Principle of Seismic Refraction Method

In a homogenous, isotropic medium (Fig.1), a circular wave is initiated by an explosion at point A, which is on the surface. The longitudinal wave emitted travels with a velocity V_1. Let this medium overlie a second medium in which the wave travels with a higher velocity V_2.

Waves striking at the contact of these two velocity discontinuities will be partially reflected back into the upper medium and partially reflected into the lower medium. In general, the method is governed by Snell's law which relates the angle of incidence (i) and refraction (r) to the seismic velocities in the media; (fig 2)

$$\frac{\sin i_1}{V_1} = \frac{\sin i_2}{V_2} \qquad \text{……1}$$

$$\frac{\sin i_c}{V_1} = \frac{\sin 90}{V_2} \qquad \text{…… 2}$$

$$i_c = \arcsin\left(\frac{V_1}{V_2}\right) \qquad \text{..……..3}$$

$$\sin i_c = \frac{V_1}{V_2} \qquad \text{....……4}$$

Where, V_1 and V_2 are the velocities of the 1st and 2nd medium respectively.

If V_2 is greater than V_1; refraction will be towards the interface, if sin i equals V_1/ V_2, the refracted ray will be parallel to the interface and some of its energy will return to the surface as a head wave that leaves the interface at the original angle of incidence (Fig.2). At greater angles of incidence there can be no refracted ray and all the energy is total internally reflected.

The elastic waves generated by the detonation of explosives near the ground surface or sledge hammer striking a metal plate on the ground, travel downwards into the various rock layers and are refracted back to the surface from the interfaces between different rock layers. Geophones at various points on the ground surface picked up the waves and recorded by seismometer. The seismogram shows when the energy started and when it was picked up at the surface.

The velocity of propagation of the wave through each rock layer can be calculated from the arrival times of different waves at different offset distances from the energy source. The velocities are characteristic of particular rocks in specific conditions, i.e. dry, jointed, saturated with water, weathered, etc. The refracted waves arrive at the surface only on the condition that the velocity of the propagation in the underlying layer is higher than that in the overlying area. The deeper a horizon is buried, the thicker it must be to properly refract the shock wave.

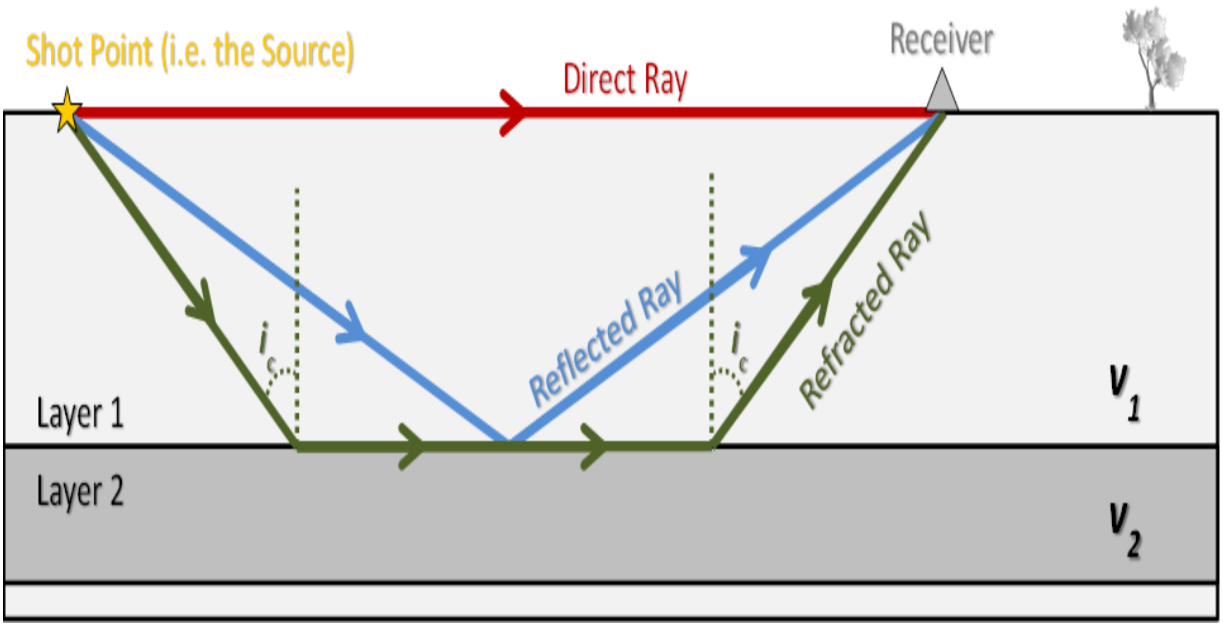

Fig 1: A homogeneous, isotropic medium showing subsurface ray paths

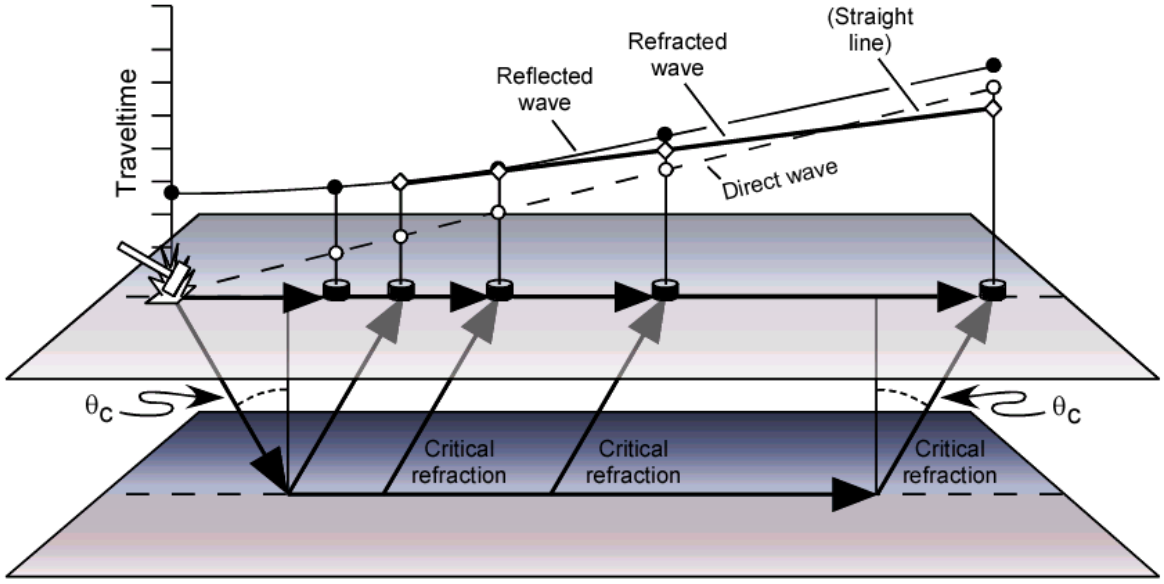

Fig 2 Relation of Critically Refracted (Head) Wave to Direct Wave and Reflected Wave

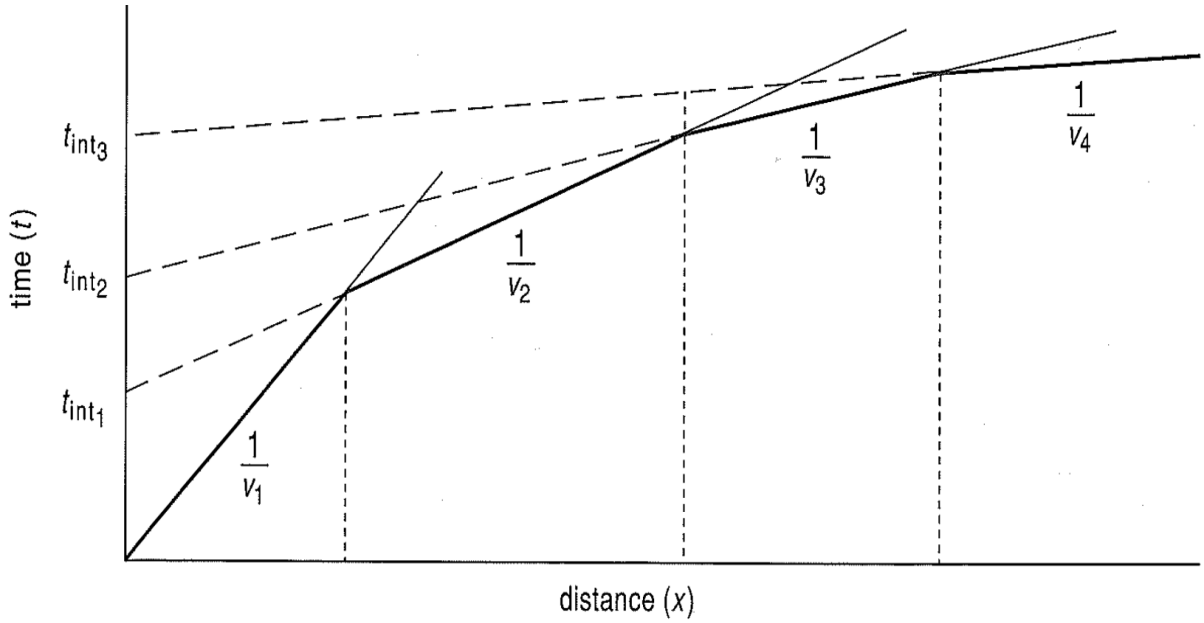

Fig 3a: Time- Distance graph of multiple horizontal layer case

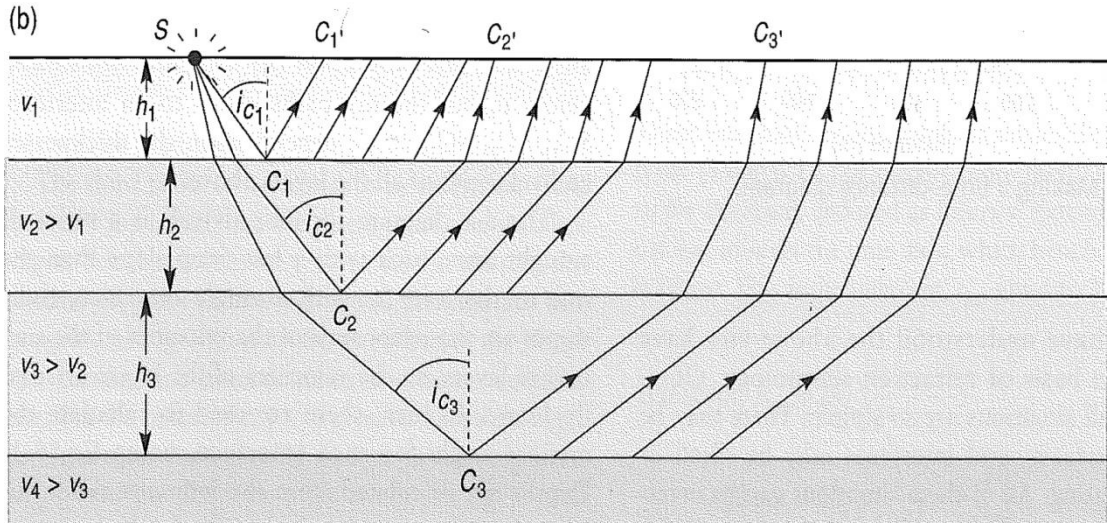

Fig 3b: Subsurface ray path geometry of multiple horizontal Layer case

Multiple Horizontal Layer Case

Consider the multiple layer case in the Fig 3a which the strata consists of several horizontal layers of thicknesses h_1, h_2, h_3, etc. The paths of rays penetrating to various beds and refracted at the discontinuities are determined by Snell's law. The delay time for each layer can be determined and the travel time along the paths can be computed by substituting the delay time of each layer.

By Snell's law

$\sin i_3 = V_3/V_4$

$\sin i_2 = (V_2/V_3) \sin 1_3 = V_2/V_4$

$\sin i_1 = (V_1/V_2) \sin 1_2 = V_1/V_4$

$\sin 1_0 = (V_0/V_1) \sin 1_1 = V_0/V_4$

Therefore;

$$\cos i_3 = \sqrt{1 - (v_3 / v_4)^2}$$

$$\cos i_2 = \sqrt{1 - (v_2 / v_4)^2}$$

$$\cos i_1 = \sqrt{1 - (v_1 / v_4)^2}$$

$$\cos i_0 = \sqrt{1 - (v_0 / v_4)^2}$$

Thus, the general expression for the time a ray penetrating to a layer n can be written as

$$T_n = \frac{x}{v_{i4}} - 2\sum_{m=0}^{n} \frac{zm\sqrt{v_n^2 - v_m^2}}{v_n v_m} \quad \text{.....5}$$

Description of the Computer Program and Method of Calculation

Theory and Model

The Visual Basic computer programming Language is a graphical language with its Integrated Development Environment (IDE) allowing Programmers to create, run and debug programs conveniently without a Window programming expert (Bradley and Millspaugh, 2002; Halvorson, 2005 and Balena, 2005). It eliminates the need for the programmer to write codes that generates the Form, codes for all the Form's properties, codes for Form placement on the screen, codes to create foreground and background colours and so on. its event-driven programming, that is codes that responds to events or notifications, allows the user to dictate the order of programming (Deitel et al, 1999). The implementation of this program by this language follows the standard procedures used in any software development.

The algorithm for uphole survey has been designed as followed:

Step 1: Input values for T (ms) axis and Z (m) axis respectively.
Step 2: Plot the points scatter for T (ms) on the vertical axis against Z (m) on the horizontal axis
Step 3: Identify the number of layers (lines on graph) plotted.
Step 4: Draw the lines of best fit for T-Z graph
Step 5: Determine the layers thicknesses (depths) and compute layers velocities.
Step 6: Display T-Z plot, layers thicknesses and velocities.
Step 7: Stop.

Description of Model 2 for Surface Refraction Method

The first step was to modify the model 1 for uphole survey computer program to be able to perform surface spread refraction and to be compatible with any Microsoft operating system.

The program can perform n- layers with layer velocity increasing with layer depth. The general structure of the program is described in three parts as shown on the block diagram.

Part 1: The computer reads and stores the input information for the problem from a notepad format. The data input format is as follow: all in a column.

Part 2: The velocity of layer 2 is estimated by two methods. The first method is equivalent to drawing a line through points of a time-distance graph and determining the inverse slope of the line. The second method used to estimate velocity was developed by the Geological Survey of Canada (Hobson and Overton, 1968). The estimate is made by use of the following formula:

$$V = \frac{\sum_{\Delta Xi} 2 - (\sum \Delta Xj)^2 / \text{n}}{\sum \Delta Xi \Delta t_1 - (\sum \Delta Xj)(\sum \Delta ti) / \text{n}}$$

When v is the desired refraction velocity,

$\Delta xi$is the difference between distance to geophone 1 from shot points on opposite ends in the spread,
$\Delta t_i$is the difference between arrivals times at geophone 1 from the same two opposing shot points,
*N*is the total number of geophones used in the computation.

After the velocity of layer 2 is established, the program estimates the position of the base of the surface layer by a computer adaptation of the delay time method. Then the accuracy of this result is improved by the stepwise multilinear regression technique.

The delay time depth points are computed for each geophone which receives the refracted waves along the interface between layers 1 and 2. If more than one depth point is available at a geophone, the depth values are averaged to establish a better estimate of the position directly beneath each geophone. For geophones with missing values, depth points are established by interpolation or extrapolation from available points nearby. Then this initial depth estimate is tested for validity by cross over distance procedure and subsequent adjustment. Starting from each depth point previously established, rays are traced upward from the velocity discontinuity to the actual positions of shot points and geophones. The direction of each is established by Snell's law form the velocities of layers 1 and 2,

Next, the travel times associated with the slant ray-segments in the surface layer are subtracted from all observed arrival times, and the velocities of all layers beneath layer 2 are estimated by the regression technique and the Hobson-Overton technique described previously. When this is completed, the user can obtain a plot of the time-distance graph

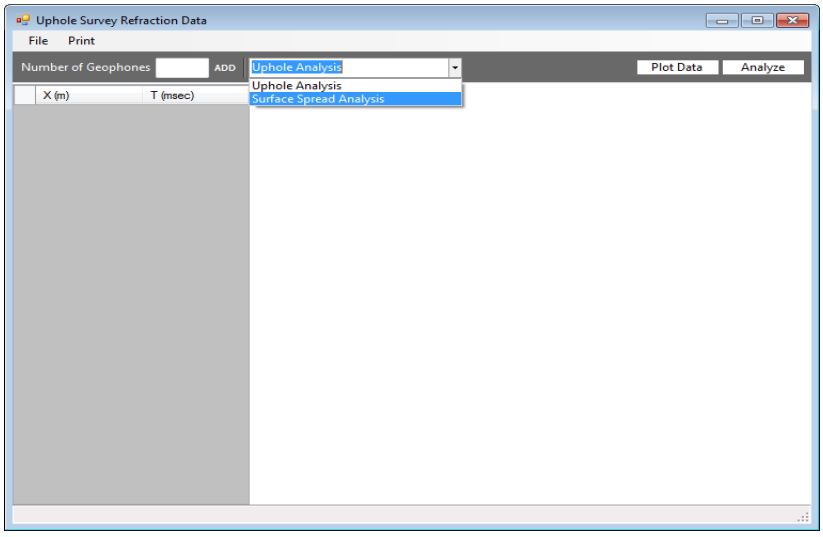

Fig 5a: Refraction Survey type selection form for the program

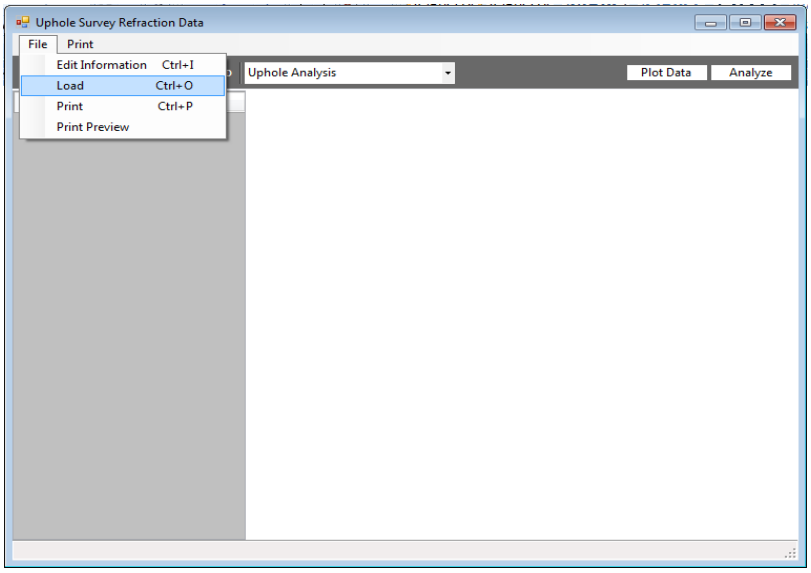

Fig. 5b: Menu bar of the program showing how to load a set of data installation

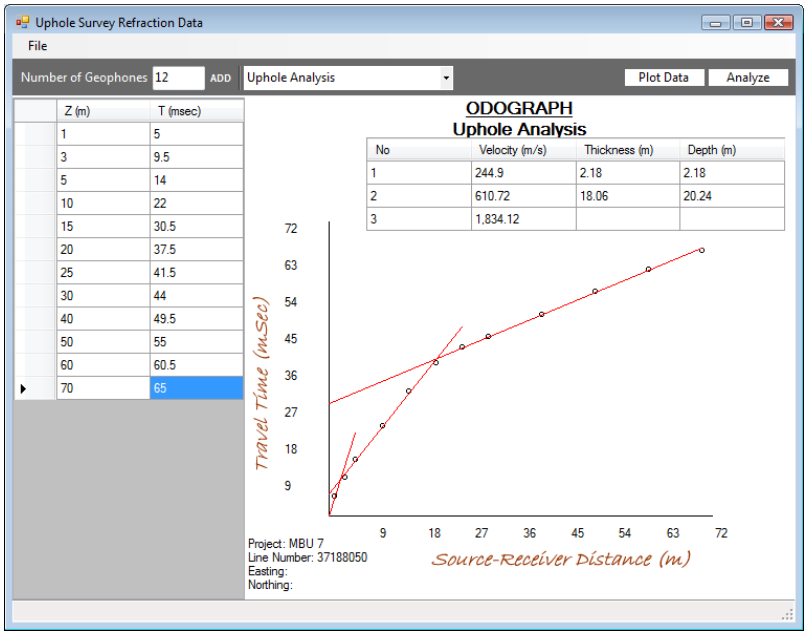

Fig 5c: Result of a typical 3-layer model of uphole survey

Conclusion and Recommendation

The computer program, FAJSEIS, has been presented for easy, fast and accurate analysis of the near-surface velocity data acquired by the Up-hole seismic method and surface spread refraction method. It has been simplified to allow for easy understanding and adoption by any earth scientist with minimal or no programming experience. The basic mathematical equations employed are simple and program codes are devoid of clumsy codes or "jumping around" due to indiscriminate branching. By its nature of implementation, FAJSEIS allows the user to be in control of the interpretation. It is flexible, that is, user-interactive and does not restrict the user on determining the expected results.

Three subsurface layers have been identified in the test data study area; the clay/clayey sand/sand topsoil which is the weathering layer; the fine/coarse sand (sub-weathering layer) and the consolidated layer of fine/coarse sand/gravel. The thicknesses for the three layers are 0.9-5.4m, mean= 3.7 ± 1.2 and 4.3-25.3m, mean= 15.1 ± 6.6 respectively while the velocities are 222 – 426m/s, mean= 337.0 ± 38.7, 450-1360m/s, mean= 661.2 ± 177.8 and 1707-2163m/s, mean= 1814.6 ± 117.4 respectively.
Based on the sampling density of 30 points, it can be inferred that the average consolidated layer velocity of the area is about 1790 m/s. This can be applied as correction velocity in static computation and 3.7 m depth is predicted as shot drilling depth in order to remove the effect of the low velocity surface layer and reduces all reflection times to a common height datum. In conclusion, the easy and quick nature of implementation and the closeness of the result to those of other researchers on the weathered layer in the Niger Delta indicate that the program is a reliable measure for the analysis of the near-surface seismic data

Appendix A: Programming Source Code

```
Imports System.Runtime.InteropServices
Public Class frmGraph
Dim nl As Integer
Dim Time(0), Depth(0) As Double
Dim maxGeoPhone = 12
Dim bpp(0) As Integer
Dim vel(0) As Double
Dim dep(0) As Double
Dim intercept(0) As Double
Dim slope(0) As Double
Dim n
Private Enum DrawingOptions
PRF_CHECKVISIBLE = &H1
PRF_NONCLIENT = &H2
PRF_CLIENT = &H4
PRF_ERASEBKGND = &H8
PRF_CHILDREN = &H10
PRF_OWNED = &H20
End Enum
Private Const WM_PRINT As Integer = &H317
<DllImport("user32.dll", CharSet:=CharSet.Auto)> _
Private Shared Function SendMessage(ByVal handle As IntPtr, ByVal msg As Integer, ByVal wParam As IntPtr, ByVal lParam As Integer) As IntPtr
End Function
Private bmp As Bitmap
Private Sub PrintDocument1_PrintPage(ByVal sender As Object, ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles PrintDocument1.PrintPage
e.Graphics.DrawImage(bmp, 0, 0)
End Sub
Private Sub btnSolve_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSolve.Click
If Val(txtN.Text) < maxGeoPhone Then MsgBox("Geophones must be more than or equal to " & maxGeoPhone, , "Error") : Exit Sub
dgItems.EndEdit()
n = dgItems.RowCount - 1
ReDim Time(n)
ReDim Depth(n)
For i As Integer = 0 To n
Time(i) = dgItems.Item(1, i).Value
Depth(i) = dgItems.Item(0, i).Value
Next
drawPoints()
btnPlot.Enabled = True
End Sub
Public Sub ClearPic()
Dim gg As Graphics = picGraph.CreateGraphics()
Dim g As Graphics = picGraph.CreateGraphics()
g = Graphics.FromImage(bmp)
picGraph.Image = bmp
End Sub
Public Sub drawGraph()
Try
Dim g As Graphics = picGraph.CreateGraphics()
picGraph.Controls.Clear()
g.Clear(Color.White)
bmp = New Bitmap(picGraph.Width, picGraph.Height)
If n <= 0 Then Exit Sub
Dim Ymax As Double, Ymin As Double
Dim Xmax As Double, Xmin As Double
Dim X As Integer, Y As Integer
Ymax = Time(0) : Ymin = Time(0) : Xmax = Depth(0) : Xmin = Depth(0)
For i = 0 To n
    If Ymax < Time(i) Then Ymax = Time(i)
    If Ymin > Time(i) Then Ymin = Time(i)
If Xmax < Depth(i) Then Xmax = Depth(i)
If Xmin > Depth(i) Then Xmin = Depth(i)
Next i
g = Graphics.FromImage(bmp)
Dim x1, x2, y1, y2
Dim margin = 80
Dim intervalY = Int(Ymax / 8) + 1
Dim intervalX = Int(Xmax / 8) + 1
x1 = margin
y1 = picGraph.Height - margin
x2 = picGraph.Width - margin
```

69. y2 = picGraph.Height - margin
70. g.DrawLine(Pens.Black, x1, y1, x2, y2)
71. g.DrawLine(Pens.Black, x1, x1, x1, y2)
72. Dim lbl As New Label
73. lbl.Width = 20
74. lbl.Height = 300
75. Dim ypos = Int((margin))
76. lbl.Location = New System.Drawing.Point(5, ypos)
77. lbl.Image = Global.Deji.My.Resources.Resources.yaxis
78. lbl.BackColor = Color.Transparent
79. Dim lbl2 As New Label
80. lbl2.Width = 300
81. lbl2.Height = 20
82. lbl2.Location = New System.Drawing.Point(picGraph.Width - ypos - 300, picGraph.Height - 50)
83. lbl2.Image = Global.Deji.My.Resources.Resources.xaxis
84. lbl2.BackColor = Color.Transparent
85. picGraph.Controls.Add(lbl)
86. picGraph.Controls.Add(lbl2)
87. Dim Yspacer = (picGraph.Height - (2 * margin)) / 8
88. Dim Xspacer = (picGraph.Width - (2 * margin)) / 8
89. X = 1000
90. Dim ppY = picGraph.Height - (margin + Yspacer)
91. Dim ppX = margin + Xspacer
92. Dim j = 1
93. For i = 0 To 8
    a. If i <> 0 Then
    b. Dim lblY As New Label
    c. Dim lblX As New Label
    d. With lblY
    e. .TextAlign = ContentAlignment.TopRight
    f. .BackColor = Color.Transparent
    g. .Text = i * interval Y
    h. .Location = New System.Drawing.Point(35, ppY)
    i. .AutoSize = True
    j. End With
    k. With lblX
    l. .TextAlign = ContentAlignment.MiddleLeft
    m. .BackColor = Color.Transparent
    n. .Text = i * intervalX
    o. .Location = New System.Drawing.Point(ppX, picGraph.Height - 70)
    p. .AutoSize = True
    q. End With
    r. ppY = ppY - Yspacer
    s. ppX = ppX + Xspacer
94. picGraph.Controls.Add(lblY)
95. picGraph.Controls.Add(lblX)
    a. End If
96. Next i
97. Dim xm As Integer = Xspacer * 8
98. Dim ym As Integer = Yspacer * 8
99. Dim xv As Integer = intervalX * 8
100. Dim yv As Integer = intervalY * 8
101. For dd As Integer = 0 To dgItems.RowCount - 1
102. Dim xp As Integer = margin + Val(Depth(dd)) * xm / xv
103. Dim yp As Integer = picGraph.Height - (margin + Val(Time(dd)) * ym / yv)
104. g.DrawEllipse(Pens.Black, xp - 2, yp - 2, 4, 4)
105. Next
106. Dim A(n) As Double
107. Dim B(n) As Double
108. For ii As Integer = 0 To n
    a. A(ii) = Depth(ii)
    b. B(ii) = Time(ii)
109. Next
110. Dim xt = 0
111. For hh As Integer = 0 To nl - 1
    a. If hh = 0 Then
    b. Dim xx1 As Integer = margin
    c. Dim yy1 As Integer = picGraph.Height - margin
    d. Dim xx2 As Integer = xt + margin + A(bpp(hh + 1)) * (xm / xv)
    e. Dim yy2 As Integer = picGraph.Height - margin - (LeastSquaresLine(A, B, 0, 0)(0) * A(bpp(hh + 1))) * (ym / yv) - (LeastSquaresLine(A, B, 0, 0)(0) * xt)
112. g.DrawLine(Pens.Red, xx1, yy1, xx2, yy2)
    a. vel(hh) = ((LeastSquaresLine(A, B, 0, 0)(0)) ^ -1) * 1000
    b. intercept(hh) = (LeastSquaresLine(A, B, 0, 0)(1))
    c. slope(hh) = LeastSquaresLine(A, B, 0, 0)(0)
    d. Else
    e. If hh <> nl - 1 Then
    f. Dim xx1 As Integer = margin

g. Dim yy1 As Integer = margin + (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(1)) * (ym / yv)
h. Dim xx2 As Integer = xt + margin + (A(bpp(hh + 1)) * (xm / xv))
i. Dim yy2 As Integer = (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0) * xt) + (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0) * A(bpp(hh + 1)) + LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(1)) * (ym / yv)

113. g.DrawLine(Pens.Red, xx1, picGraph.Height - yy1, xx2, picGraph.Height - (margin + yy2))
    a. vel(hh) = ((LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0)) ^ -1) * 1000
    b. intercept(hh) = (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(1))
    c. slope(hh) = LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0)
    d. Else
    e. Dim xx1 As Integer = margin
    f. Dim yy1 As Integer = margin + (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(1)) * (ym / yv)
    g. Dim xx2 As Integer = xt + margin + (A(bpp(hh + 1) - 1) * (xm / xv))
    h. Dim yy2 As Integer = (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0) * xt) + (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0) * A(bpp(hh + 1) - 1) + LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(1)) * (ym / yv)
114. g.DrawLine(Pens.Red, xx1, picGraph.Height - yy1, xx2, picGraph.Height - (margin + yy2))
    a. vel(hh) = ((LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0)) ^ -1) * 1000
    b. intercept(hh) = (LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(1))
    c. slope(hh) = LeastSquaresLine(A, B, bpp(hh), bpp(hh + 1))(0)
    d. End If
    e. End If
115. Next
116. picGraph.Image = bmp
117. Dim wdt = 400
118. Dim lblInfo As New Label
119. lblInfo.Location = New System.Drawing.Point(0, picGraph.Height - 60)
120. lblInfo.Height = 60
121. lblInfo.Width = 200
122. lblTitle.Text = "ODOGRAPH" & ChrW(10) & ToolStripComboBox1.Text
123. lblInfo.Text = "Project: " & SIproject & ChrW(10) & "Line Number: " & SIlineshortpointno & ChrW(10) & "Easting: " & SIeasting & ChrW(10) & "Northing: " & SInorthing
124. picGraph.Controls.Add(lblInfo)
125. Dim dg As New DataGridView
126. picGraph.Controls.Add(dg)
127. dg.Location = New System.Drawing.Point(picGraph.Width - wdt - 10, 0)
128. dg.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill
129. dg.RowHeadersVisible = False
130. dg.Width = wdt
131. dg.Height = 100
132. dg.AllowUserToAddRows = False
133. dg.BackgroundColor = Color.White
134. dg.BorderStyle = BorderStyle.None
135. dg.DefaultCellStyle.SelectionBackColor = Color.White
136. dg.Columns.Add("No", "No")
137. dg.Columns.Add("Velocity (m/s)", "Velocity (m/s)")

```
dg.Columns.Add( "Thickness (m)", "Thickness (m)")
dg.Columns.Add( "Depth (m)", "Depth (m)")
For iii As Integer = 0 To nl - 1
    dg.Rows.Add()
dg.Item(0, iii).Value = iii + 1
dg.Item(1, iii).Value = Format(vel(iii), "#,###.##")
If iii <> nl - 1 Then
    If type = 1 Then
    Dim yy = ((vel(iii + 1) + vel(iii)) / (vel(iii + 1) - vel(iii))) ^ 0.5
    Dim kk = ((intercept(iii + 1) - intercept(iii)) / (slope(iii) - slope(iii + 1))) / (2 * yy)
    dg.Item(3, iii).Value = (Format(kk, "#,###.##"))
    Else
    Dim kk = (intercept(iii + 1) - intercept(iii)) / (slope(iii) - slope(iii + 1))
    dg.Item(3, iii).Value = (Format(kk, "#,###.##"))
    End If
    End If
Next
For iii As Integer = 0 To nl - 2
    If iii = 0 Then dg.Item(2, iii).Value = dg.Item(3, iii).Value
    If iii <> 0 Then dg.Item(2, iii).Value = Val(dg.Item(3, iii).Value) - Val(dg.Item(3, iii - 1).Value)
Next
dg.Item(0, 0).Selected = False
Catch ex As Exception
End Try
End Sub
Public Sub drawPoints()
Dim g As Graphics = picGraph.CreateGraphics()
picGraph.Controls.Clear()
g.Clear(Color.White)
bmp = New Bitmap(picGraph.Width, picGraph.Height)
If n <= 0 Then Exit Sub
Dim Ymax As Double, Ymin As Double
Dim Xmax As Double, Xmin As Double
Dim X As Integer, Y As Integer
Ymax = Time(0) : Ymin = Time(0) : Xmax = Depth(0) : Xmin = Depth(0)
For i = 0 To n
If Ymax < Time(i) Then Ymax = Time(i)
If Ymin > Time(i) Then Ymin = Time(i)
If Xmax < Depth(i) Then Xmax = Depth(i)
If Xmin > Depth(i) Then Xmin = Depth(i)
Next i
g = Graphics.FromImage(bmp)
Dim x1, x2, y1, y2
Dim margin = 80
Dim intervalY = Int(Ymax / 8) + 1
Dim intervalX = Int(Xmax / 8) + 1
x1 = margin
y1 = picGraph.Height - margin
x2 = picGraph.Width - margin
y2 = picGraph.Height - margin
g.DrawLine(Pens.Black, x1, y1, x2, y2)
g.DrawLine(Pens.Black, x1, x1, x1, y2)
Dim lbl As New Label
lbl.Width = 20
lbl.Height = 300
Dim ypos = Int((margin))
lbl.Location = New System.Drawing.Point(5, ypos)
lbl.Image = Global.Deji.My.Resources.Resources.yaxis
lbl.BackColor = Color.Transparent
Dim lbl2 As New Label
lbl2.Width = 300
lbl2.Height = 20
lbl2.Location = New System.Drawing.Point(picGraph.Width - ypos - 300, picGraph.Height - 50)
lbl2.Image = Global.Deji.My.Resources.Resources.xaxis
lbl2.BackColor = Color.Transparent
picGraph.Controls.Add(lbl)
picGraph.Controls.Add(lbl2)
Dim Yspacer = (picGraph.Height - (2 * margin)) / 8
Dim Xspacer = (picGraph.Width - (2 * margin)) / 8
X = 1000
Dim ppY = picGraph.Height - (margin + Yspacer)
Dim ppX = margin + Xspacer
Dim j = 1
For i = 0 To 8
If i <> 0 Then
    Dim lblY As New Label
    Dim lblX As New Label
    With lblY
    .TextAlign = ContentAlignment.TopRight
    .BackColor = Color.Transparent
    .Text = i * intervalY
    .Location = New System.Drawing.Point(35, ppY)
    .AutoSize = True
    End With
    With lblX
```

```
        k. .TextAlign = ContentAlignment.MiddleLeft
        l. .BackColor = Color.Transparent
        m. .Text = i * intervalX
        n. .Location = New System.Drawing.Point(ppX, picGraph.Height - 70)
        o. .AutoSize = True
        p. End With
        q. ppY = ppY - Yspacer
        r. ppX = ppX + Xspacer
201. picGraph.Controls.Add(lblY)
202. picGraph.Controls.Add(lblX)
203. End If
204. Next i
205. Dim xm As Integer = Xspacer * 8
206. Dim ym As Integer = Yspacer * 8
207. Dim xv As Integer = intervalX * 8
208. Dim yv As Integer = intervalY * 8
209. For dd As Integer = 0 To dgItems.RowCount - 1
210. Dim xp As Integer = margin + Val(Depth(dd)) * xm / xv
211. Dim yp As Integer = picGraph.Height - (margin + Val(Time(dd)) * ym / yv)
212. g.DrawEllipse(Pens.Black, xp - 2, yp - 2, 4, 4)
213. Next
214. picGraph.Image = bmp
215. End Sub
216. Private Sub txtN_KeyDown(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyEventArgs) Handles txtN.KeyDown
217. If e.KeyData = Keys.Return Then
218. If Val(txtN.Text) < maxGeoPhone Then MsgBox("Geophones must be more than or equal to " & maxGeoPhone, , "Error") : Exit Sub
    updateRow(Val(txtN.Text))
219. dgItems.Focus()
220. End If
221. End Sub
222. Public Sub updateRow(ByVal n)
223. dgItems.Rows.Clear()
224. For tt As Integer = 1 To n
225. dgItems.Rows.Add()
226. Next tt
227. For tt As Integer = 0 To n - 1
228. dgItems.Rows(tt).HeaderCell.Value = (tt + 1).ToString
229. Next
230. End Sub
231. Private Sub btnAddRow_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnAddRow.Click
232. dgItems.Rows.Add()
233. dgItems.Rows(dgItems.Rows.Count - 1).HeaderCell.Value = (dgItems.Rows.Count).ToString
234. txtN.Text = Val(txtN.Text) + 1
235. End Sub
236. Private Sub Form1_Resize(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Resize
237. drawGraph()
238. End Sub
239. Public Function LeastSquaresLine(ByVal X() As Double, ByVal Y() As Double, ByVal start As Integer, ByVal finish As Integer) As Double()
240. If start = 0 And finish = 0 Then
241. Dim st As Double = 0
242. Dim sa As Double = 0
243. For I As Integer = 0 To bpp(1) - 1
        a. st += Y(I) / X(I)
244. Next
245. Dim Slope = st / (bpp(1))
246. Return New Double() {Slope, 0}
247. Else
248. Dim SumX, SumXSquared, SumY, SumXY As Double
249. For I As Integer = start - 1 To finish - 1
        a. SumX += X(I)
        b. SumXSquared += X(I) * X(I)
        c. SumY += Y(I)
        d. SumXY += X(I) * Y(I)
250. Next
251. Dim Slope, Intercept, Determinant As Double
252. Determinant = (finish - start + 1) * SumXSquared - SumX * SumX
253. Slope = ((finish - start + 1) * SumXY - SumY * SumX) / Determinant
254. Intercept = (SumY * SumXSquared - SumX * SumXY) / Determinant
255. Return New Double() {Slope, Intercept}
256. End If
257. End Function
258. Private Sub EditInformationToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles EditInformationToolStripMenuItem.Click
259. frmSurveyInfo.ShowDialog()
260. End Sub
261. Private Sub frmGraph_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
262. ToolStripComboBox1.SelectedIndex = 0
263. updateType(0)
264. End Sub
265. Private Sub ToolStripButton1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnPlot.Click
266. Try
267. If Val(txtN.Text) < maxGeoPhone Then MsgBox("Geophones must be more than or equal to " & maxGeoPhone, , "Error") : Exit Sub
268. dgItems.EndEdit()
```

```
n = dgItems.RowCount - 1
ReDim Time(n)
ReDim Depth(n)
For i As Integer = 0 To n
    Time(i) = dgItems.Item(1, i).Value
    Depth(i) = dgItems.Item(0, i).Value
Next
drawPoints()
nl = InputBox("Number of layers", "Input Layer")
If nl >= 2 Then
    ReDim Preserve bpp(nl)
    ReDim Preserve vel(nl - 1)
    ReDim Preserve slope(nl - 1)
    bpp(nl) = Depth.Length
    ReDim Preserve dep(nl - 1)
    ReDim Preserve intercept(nl - 1)
    bpp(0) = 1
    If nl = 2 Then
    bpp(1) = InputBox("Breakpoint", "Input Layer Breakpoint")
    Else
    For ttt As Integer = 1 To nl - 1
    bpp(ttt) = InputBox("Breakpoint " & ttt)
    Next
    End If
End If
drawGraph()
flagPlot = True
Catch InvalidCastException As Exception
End Try
End Sub
Private Sub LoadToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles LoadToolStripMenuItem.Click
Dim dg As New OpenFileDialog
dg.Filter = "Text files (*.txt)|*.txt|All files (*.*)|*.*"
dg.ShowDialog()
Dim filename = dg.FileName
If filename <> "" Then loadFile(filename)
End Sub
Public Sub loadFile(ByVal filename)
dgItems.Rows.Clear()
FileOpen(1, filename, OpenMode.Input)
Input(1, n)
txtN.Text = n
Dim jj
For ii As Integer = 0 To n - 1
dgItems.Rows.Add()
Next
For i As Integer = 0 To n - 1
Input(1, jj)
dgItems.Item(0, i).Value = jj
Next
For i As Integer = 0 To n - 1
Input(1, jj)
dgItems.Item(1, i).Value = jj
Next
FileClose(1)
End Sub
Private Sub ToolStripComboBox1_SelectedIndexChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles ToolStripComboBox1.SelectedIndexChanged
updateType(ToolStripComboBox1.SelectedIndex)
End Sub
Private Sub updateType(ByVal index As Integer)
If index = 0 Then
dgItems.Columns(0).HeaderText = "Z (m)"
Me.Text = "Uphole Survey Refraction Data"
type = 0
Else
dgItems.Columns(0).HeaderText = "X (m)"
Me.Text = "Surface Spread Refraction Data"
type = 1
End If
If flagPlot = True Then
drawGraph()
lblTitle.Text = "ODOGRAPH" & ChrW(10) & ToolStripComboBox1.Text
End If
End Sub
Private Sub ToolStripButton1_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs)
frmSurveyInfo.ShowDialog()
End Sub
Private Sub PrintToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PrintToolStripMenuItem1.Click
If (PrintDialog1.ShowDialog = Windows.Forms.DialogResult.OK) Then
PrintDocument1.PrinterSettings = PrintDialog1.PrinterSettings
bmp = New Bitmap(Panel3.DisplayRectangle.Width, Panel3.DisplayRectangle.Height)
Dim G As Graphics = Graphics.FromImage(bmp)
Dim Hdc As IntPtr = G.GetHdc()
SendMessage(Panel3.Handle, WM_PRINT, Hdc, DrawingOptions.PRF_OWNED Or DrawingOptions.PRF_CHILDREN Or DrawingOptions.PRF_CLIENT Or DrawingOptions.PRF_NONCLIENT)
SendMessage(Panel3.Handle, WM_PRINT, Hdc, DrawingOptions.PRF_OWNED Or DrawingOptions.PRF_CHILDREN Or DrawingOptions.PRF_CLIENT Or DrawingOptions.PRF_NONCLIENT)
G.ReleaseHdc(Hdc)
G.Dispose()
PrintDocument1.Print()
Else
Exit Sub
End If
End Sub
Private Sub PrintPreviewToolStripMenuItem_Click(ByVal sender As System.Object,
```

```
ByVal e As System.EventArgs) Handles PrintPreviewToolStripMenuItem.Click
bmp = New Bitmap(Panel3.DisplayRectangle.Width, Panel3.DisplayRectangle.Height)
Dim G As Graphics = Graphics.FromImage(bmp)
Dim Hdc As IntPtr = G.GetHdc()
SendMessage(Panel3.Handle, WM_PRINT, Hdc, DrawingOptions.PRF_OWNED Or DrawingOptions.PRF_CHILDREN Or DrawingOptions.PRF_CLIENT Or DrawingOptions.PRF_NONCLIENT)
G.ReleaseHdc(Hdc)
G.Dispose()
PrintPreviewDialog1.Document = PrintDocument1
PrintPreviewDialog1.ShowDialog()
End Sub
End Class
Public Class frmHome
Private Sub btnContinue_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnContinue.Click
frmSurveyInfo.Show()
Me.Close()
End Sub
End Class
Public Class frmSurveyInfo
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
frmGraph.Show()
Me.Hide()
End Sub
Private Sub Button1_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
updateSurveyInfo()
frmGraph.Show()
If flagPlot = True Then
frmGraph.drawGraph()
End If
Me.Close()
End Sub
Public Sub updateSurveyInfo()
SIproject = txtProject.Text
SIsite = txtSite.Text
SIclient = txtClient.Text
SIcontractor = txtContractor.Text
SIrecordinginstrument = txtRecordingInstrument.Text
SIchannels = txtChannels.Text
SIenergysource = txtEnergySource.Text
SIdepth = txtDepth.Text
SIlineshortpointno = txtLineShortPointNo.Text
SIacqday = txtAcqDay.Text
SIacqmonth = txtAcqMonth.Text
SIacqyear = txtAcqYear.Text
SIstationno = txtStationNumber.Text
SIeasting = txtEasting.Text
SInorthing = txtNorthing.Text
SIelevation = txtElevation.Text
SIcomments = txtComments.Text
End Sub
Private Sub frmSurveyInfo_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
txtProject.Text = SIproject
txtSite.Text = SIsite
txtClient.Text = SIclient
txtContractor.Text = SIcontractor
txtRecordingInstrument.Text = SIrecordinginstrument
txtChannels.Text = SIchannels
txtEnergySource.Text = SIenergysource
txtDepth.Text = SIdepth
txtLineShortPointNo.Text = SIlineshortpointno
txtAcqDay.Text = SIacqday
txtAcqMonth.Text = SIacqmonth
txtAcqYear.Text = SIacqyear
txtStationNumber.Text = SIstationno
txtEasting.Text = SIeasting
txtNorthing.Text = SInorthing
txtElevation.Text = SIelevation
txtComments.Text = SIcomments
End Sub
End Class
Module vars
Public type As String
Public SIproject
Public SIsite
Public SIclient
Public SIcontractor
Public SIrecordinginstrument
Public SIchannels
Public SIenergysource
Public SIdepth
Public SIlineshortpointno
Public SIacqday
Public SIacqmonth
Public SIacqyear
Public SIstationno
Public SIeasting
Public SInorthing
Public SIelevation
Public SIcomments
Public flagPlot As Boolean
End Module
```

References

[1] H.D. Ackermann, L.W. Pankratz and D. Dansereau, Resolution of ambiguities of shallow refraction travel-time curves, *Geophysics*, 51(1986), 223-235.

[2] I.O. Akpabio and H.O. Onwusiri, Computation of low velocity layer correction parameters in parts of western Niger Delta, (OML 62), *Global Journal of Geological Sciences*, 2(1) (2004), 45-50.

[3] G.I. Alaminiokuma, E.D. Uko and C. Israel-Cookey, Near-surface seismic velocity data: A computer program for analysis, *Nigeria Journal of Physics*, 19(2) (2007), 253-266.

[4] J.C. Bradley and A.C. Millspaugh, Programming in Visual Basic 6.0 (Update Edition), McGraw-Hill Companies, New York, 2000.

[5] H. Doust and E. Omatsola, Niger Delta, In: J.D. Edwards and P.A. Santagrosseid (eds.), Divergent/Passive margin basins, *America Association of Petroleum Geologists Memoir*, 48(1990), 239-248.

[6] C.L. Eze, E.E. Okwueze and E.D. Uko, The velocity thickness characteristics of the mangrove: Swamp Low Velocity Layer (LVL) South Central Niger Delta, Nigeria, *Global Journal of Pure and Applied Sciences*, 9(3) (2003), 369-374.

[7] P. Kearey and M. Brooks, *A Introduction to Geophysical Exploration*, Blackwell Science Ltd., (1991), 21-115.

[8] O. Ogunkorode, Application of seismic refraction prospecting method in characterizing seismic weathered layer: Case Study of mbu field, Onshore Niger-Delta, *Unpublished M.Sc. Dissertation*, Geology Dept., University of Ibadan, Ibadan, (2007).

[9] E.C. Okolie, F.C. Ugbe and B.O. Uyouyou, Analytical determination of low velocity layer in 4-D hydrocarbon prospecting in parts of Imo state, *Journal of Applied Mathematics and Physics*, 11(2007), 393-402.

[10] E.D. Uko, A.S. Ekine, J.O. Ebeniro and C.O. Ofoegbu, Weathering structure of the east-central Niger Delta, Nigeria, *Geophysics*, 57(1992), 1228-1233.

[11] R.J. Whiteley, Seismic refraction testing - A tutorial, In R.D. Woods (ed.), Geophysical Characterization of Sites, Oxford & IBH Publishing, (1994), 45-47.