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 different methods of investigating 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.

Up hole surveys are the most direct measures of near surface properties (Knox, 1967, Rogers, 1981). The up hole 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 994 m/s with variation in thickness from 2.8 m to 52.8 m.

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 340-1000 m/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 2.9 m and 45.5 m, with about 20.0 m on a regional average. An average compressional-wave velocity of about 500 m/s and 1732 m/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)

\[
\frac{\sin i_1}{V_1} = \frac{\sin i_2}{V_2} \quad \text{...............1}
\]

\[
\frac{\sin i_c}{V_1} = \frac{\sin 90}{V_2} \quad \text{...............2}
\]

\[
i_c = \arcsin \left( \frac{V_1}{V_2} \right) \quad \text{...............3}
\]

\[
\sin i_c = \frac{V_1}{V_2} \quad \text{...............4}
\]

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
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

\[ \sin i_3 = \frac{V_3}{V_4} \]
\[ \sin i_2 = \left( \frac{V_2}{V_3} \right) \sin i_3 = \frac{V_2}{V_4} \]
\[ \sin i_1 = \left( \frac{V_1}{V_2} \right) \sin i_2 = \frac{V_1}{V_4} \]
\[ \sin i_0 = \left( \frac{V_0}{V_1} \right) \sin i_1 = \frac{V_0}{V_4} \]

Therefore;

\[ \cos i_3 = \sqrt{1 - \left( \frac{v_1}{v_4} \right)^2} \]
\[ \cos i_2 = \sqrt{1 - \left( \frac{v_2}{v_4} \right)^2} \]
\[ \cos i_1 = \sqrt{1 - \left( \frac{v_1}{v_4} \right)^2} \]
\[ \cos i_0 = \sqrt{1 - \left( \frac{v_0}{v_4} \right)^2} \]

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

\[ T_n = \frac{x}{v_{i_4}} - 2 \sum_{m=0}^{m=n} \frac{zm \sqrt{v_n^2 - v_m^2}}{v_n v_m} \]

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:

\[ V = \frac{\sum_{i} \Delta X_i^2 - (\sum \Delta X_j)^2}{n} \]

\[ \frac{\sum \Delta X_i \Delta t_i - (\sum \Delta X_j)(\sum \Delta t_i)}{n} \]

When \( v \) is the desired refraction velocity,

\( \Delta x \) is the difference between distance to geophone 1 from shot points on opposite ends in the spread,

\( \Delta t \) is the difference between arrival 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 from 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
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 error.

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

```vbnet
1. Imports System.Runtime.InteropServices
2. Public Class frmGraph
3. Dim n As Integer
4. Dim Time(n) As Double
5. Dim maxGeoPhone = 12
6. Dim bpp(0) As Integer
7. Dim vel(0) As Double
8. Dim dep(0) As Double
9. Dim intercept(0) As Double
10. Dim slope(0) As Double
11. Dim n
12. Private Enum DrawingOptions
13. PRF_DRAW = &H1
14. PRF_NONCLIENT = &H2
15. PRF_CLIENT = &H4
16. PRF_ERASEBK = &H8
17. PRF_CHILDREN = &H10
18. PRF_OWNERED = &H20
19. End Enum
20. Private Const WM_PRINT As Integer = &H317
21. <DllImport("user32.dll", CharSet:=CharSet.Auto)> _
22. Private Shared Function SendMessage(ByVal
23. End Function
24. Private bmp As Bitmap
26. Dim g As Graphics
27. End Sub
28. Private Sub btnSolve_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
29. If Val(txtN.Text) <= 0 Then
30. n = dgItems.RowCount - 1
31. ReDim Time(n)
32. ReDim Depth(n)
33. For i As Integer = 0 To n
34. Time(i) = dgItems.Item(i, i).Value
35. Depth(i) = dgItems.Item(i, i).Value
36. Next i
37. btnPlot.Enabled = True
38. End Sub
39. Public Sub ClearPic()
40. Dim gg As Graphics = picGraph.CreateGraphics()
41. Dim g As Graphics = picGraph.CreateGraphics()
42. Dim b As Integer = Xmin = Depth(0)
43. g = Graphics.FromImage(bmp)
44. picGraph.Image = bmp
45. End Sub
46. Public Sub drawGraph()
47. Try
48. Dim g As Graphics = picGraph.CreateGraphics()
49. picGraph.Controls.Clear()
50. g.Clear(Color.White)
51. bmp = New Bitmap(picGraph.Width, picGraph.Height)
52. If n <= 0 Then
53. Dim Ymax As Double, Ymin As Double
54. Dim Xmax As Double, Xmin As Double
55. Dim X As Integer, Y As Integer
56. Ymax = Time(0) : Ymin = Time(0) : Xmax = Depth(0) : Xmin = Depth(0)
57. For i = 0 To n
58. If Ymax < Time(i) Then Xmax = Depth(i)
59. If Ymin > Time(i) Then Xmin = Depth(i)
60. Next i
61. g = Graphics.FromImage(bmp)
62. Dim x1, x2, y1, y2
63. Dim margin = 80
64. Dim intervalX = Int(Xmax / 8) + 1
65. Dim intervalY = Int(Ymax / 8) + 1
66. x1 = margin
67. y1 = picGraph.Height - margin
68. x2 = picGraph.Width - margin
```

Appendix A: Programming Source Code

1. Imports System.Runtime.InteropServices
2. Public Class frmGraph
3. Dim n As Integer
4. Dim Time(n) As Double
5. Dim maxGeoPhone = 12
6. Dim bpp(0) As Integer
7. Dim vel(0) As Double
8. Dim dep(0) As Double
9. Dim intercept(0) As Double
10. Dim slope(0) As Double
11. Dim n
12. Private Enum DrawingOptions
13. PRF_DRAW = &H1
14. PRF_NONCLIENT = &H2
15. PRF_CLIENT = &H4
16. PRF_ERASEBK = &H8
17. PRF_CHILDREN = &H10
18. PRF_OWNERED = &H20
19. End Enum
20. Private Const WM_PRINT As Integer = &H317
21. <DllImport("user32.dll", CharSet:=CharSet.Auto)> _
22. Private Shared Function SendMessage(ByVal
23. End Function
24. Private bmp As Bitmap
26. Dim g As Graphics
27. End Sub
28. Private Sub btnSolve_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
29. If Val(txtN.Text) <= 0 Then
30. n = dgItems.RowCount - 1
31. ReDim Time(n)
32. ReDim Depth(n)
33. For i As Integer = 0 To n
34. Time(i) = dgItems.Item(i, i).Value
35. Depth(i) = dgItems.Item(i, i).Value
36. Next i
37. btnPlot.Enabled = True
38. End Sub
39. Public Sub ClearPic()
40. Dim gg As Graphics = picGraph.CreateGraphics()
41. Dim g As Graphics = picGraph.CreateGraphics()
42. Dim b As Integer = Xmin = Depth(0)
43. g = Graphics.FromImage(bmp)
44. picGraph.Image = bmp
45. End Sub
46. Public Sub drawGraph()
47. Try
48. Dim g As Graphics = picGraph.CreateGraphics()
49. picGraph.Controls.Clear()
50. g.Clear(Color.White)
51. bmp = New Bitmap(picGraph.Width, picGraph.Height)
52. If n <= 0 Then
53. Dim Ymax As Double, Ymin As Double
54. Dim Xmax As Double, Xmin As Double
55. Dim X As Integer, Y As Integer
56. Ymax = Time(0) : Ymin = Time(0) : Xmax = Depth(0) : Xmin = Depth(0)
57. For i = 0 To n
58. If Ymax < Time(i) Then Xmax = Depth(i)
59. If Ymin > Time(i) Then Xmin = Depth(i)
60. Next i
61. g = Graphics.FromImage(bmp)
62. Dim x1, x2, y1, y2
63. Dim margin = 80
64. Dim intervalX = Int(Xmax / 8) + 1
65. Dim intervalY = Int(Ymax / 8) + 1
66. x1 = margin
67. y1 = picGraph.Height - margin
68. x2 = picGraph.Width - margin
Dim j = 1
margin + Xspacer
Yspacer = (margin +
picGraph.Width
X = 1000
Dim ppY = picGraph.Height -
(margin +
Yspacer)
Dim ppX = margin + Xspacer
Dim j = 1
For i = 0 To 8
a. If i < 0
Then
b. Dim
lblY As
New
Label
x2 As
Integer
= xp +
margin
+ A(bpp(h
h + 1))
* (xm /
yx) =
Dim
yy2 As
Integer
= picGraph.
h.Height
- margin
- (LeastS
quaresL
ineA,
B, 0, 0)
0(0) *
End If
Else
End For
 lblX = 
New
Label
Dim ypos = Int((margin))
Dim lbl.Location = New
System.Drawing.
Point(5, ypos)
Dim lbl.Image =
esources.Resource
s.xaxis
Dim lbl.BackColor =
Color.Transparent
Dim lbl2 As New
Label
Dim lbl2.Width = 300
Dim lbl2.Height = 20
Dim lbl2.Location = New
System.Drawing.
Point(picGraph.W
idth - ypos = 300,
picGraph.Height - 50)
Dim lbl2.Image =
esources.Resource
s.xaxis
Dim lbl2.BackColor =
Color.Transparent
Dim picGraph.Controls.
.Add(lbl2)
Dim DimYs =
(picGraph.Height
- (2 * margin)) / 8
Dim Xs =
(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
a. If i < 0
Then
b. Dim
lblY As
New
Label

For hh As Integer
Next
a. vel(hh) =
(LeastS
quaresL
ineA,
B, 0, 0)
0(0) *
b. intersect
(t(hh)) =
(LeastS
quaresL
ineA,
B, 0, 0)
0(0) *
c. slope(hh)
= LeastSq
quaresLi
ne(A, B,
0, 0(0)
0(0))
d. Else
Then
f. Dim
x21 As
Integer
= -
margin
"Int. J. Pure Appl. Sci. Tec
92.
91.
89.
88.
86.
83.
80.
77.
72.
69.
67.
66.
65.
64.
63.
62.
61.
60.
59.
58.
57.
56.
55.
54.
53.
52.
51.
50.
49.
48.
47.
46.
45.
44.
43.
42.
41.
40.
39.
38.
37.
36.
35.
34.
33.
32.
31.
30.
29.
28.
27.
26.
25.
24.
23.
22.
21.
20.
19.
18.
17.
16.
15.
14.
13.
12.
11.
10.
9.
8.
7.
6.
5.
4.
3.
2.
1.
0.
"
g. Dim YY1 As Integer
= margin + (LeastSquaresL
ine(A, B, bpp(hh), bpp(hh) * + 1)(0)
+ (y)v)
+ 1)(1) * (ym / yv)

h. Dim xX2 As Integer
= xt + margin + (A(bpp(hh) + 1)(0) * (xv / xv))

i. Dim YY2 As Integer
= (LeastSquaresL
ine(A, B, bpp(hh), bpp(hh) + 1)(0) * xt) + (LeastSquaresL
ine(A, B, bpp(hh), bpp(hh) + 1)(0) * (y)v)

j. Dim yX1 As Integer
= margin + (LeastSquaresL
ine(A, B, bpp(hh), bpp(hh) + 1)(0) * (y)v)

k. Dim yY1 As Integer
= (LeastSquaresL
ine(A, B, bpp(hh), bpp(hh) + 1)(0) * (y)v)

l. Dim xX2 As Integer
= xt + margin + (A(bpp(hh) + 1)(0) * (xv / xv))
If iii <> nl

Format(vel(iii),
nl + 1)

Dim kk

= ((interce 
pt(iii) + 
1) - 
intersect 
t(iii)) / 
(slope(ii) 
i) - 
slope(ii) 
+ 1)) / 
(2 * yy)

d. dg.Item( 
3, 
iii).Valu 
e = (Format 
(kk, 
"#,#.##") 
: Xmin = 
Depth(0)

161. For i = 0 To n

162. If Ymax < 
Time(i) Then 
Ymin = Time(i)

163. If Ymin > Time(i) Then 
Xmin = Time(i)

164. If Xmin < 
Depth(i) Then 
Xmax = Depth(i)

165. If Xmin > 
Depth(i) Then 
Xmin = Depth(i)

166. Next i

167. g = 
Graphics.FromIm 
age(bmp)

168. Dim x1, x2, y1, y2

169. Dim margin = 80

170. Dim intervalY = 
Int(Ymax / 8) + 1

171. Dim intervalX = 
Int(Xmax / 8) + 1

172. x1 = margin

173. y1 = 
picGraph.Height - 
margin

174. x2 = 
picGraph.Width - 
margin

175. y2 = 
picGraph.Height - 
margin

176. g.DrawLine(Pens. 
Black, x1, y1, x2, 
y2)

177. g.DrawLine(Pens. 
Black, x1, x1, y1, 
y2)

178. Dim lbl As New 
Label

179. lbl.Width = 20

180. lbl.Height = 300

181. Dim ypos = 
Int((margin))

182. lbl.Location = 
New 
System.Drawing. 
Point(5, ypos)

183. lbl.Image = 
esources.Resource 
s.xaxis

184. lbl.BackColor = 
Color.Transparent

185. Dim lbl2 As New 
Label

186. lbl2.Width = 300

187. lbl2.Height = 20

188. lbl2.Location = 
New 
System.Drawing. 
Point(picGraph.W 
idth - ypos - 300,

Xmin = 

picGraph.Height - 
50)

189. lbl2.Image = 
esources.Resource 
s.xaxis

190. lbl2.BackColor = 
Color.Transparent

191. picGraph.Controls 
.Add(lbl)

192. picGraph.Controls 
.Add(lbl2)

193. Dim Ysparser = 
(picGraph.Height - 
(2 * margin)) / 8

194. Dim Xsparser = 
(picGraph.Width - 
(2 * margin)) / 8

195. X = 1000

196. Dim ppY = 
picGraph.Height - 
(margin + 
Ysparser)

197. Dim ppX = 
margin + Xsparser

198. Dim j = 1

199. For i = 0 To 8

200. If i <> 0 Then

a. Dim 
lblY As 
New 
Label

b. Dim 
lblX As 
New 
Label

c. With 
lblY

d. .TextAli 
gn = 
Content 
Alignm 
ent.Top 
Right

e. BackCo 
lor = 
Color.Tr 
anspare 
t
f. .Text = i 
interval 
Y

g. .Locatio 
n = 
New 
System.

h. .AutoSi 
ze = 
True

i. End 
With

j. With 
lblX
212. g.DrawEllipse(Pe ns.Black, xp - 2, yp - 2, 4, 4)
213. Next
214. picGraph.Image = bmp
215. End Sub
216. Private Sub
217. If e.KeyData = Keys.Return Then
218. If Val(txtN.Text) >= maxGeoPhone
219. dgItems.Focus()
220. End If
221. End Sub
222. Public Sub
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
232. dgItems.Rows.Add()
233. txtN.Text = Val(txtN.Text) + 1
234. For I As Integer = 0 To bpp(1) - 1
235. Dim Slope As Double, Y(I) As Double, Intercept As Double
236. Dim Slope As Double
237. Dim slope As Double
238. Dim Sub
239. Public Function
240. If start = 0 And finish = 0 Then
241. Dim st As Double
242. Dim sa As Double
243. For I As Integer = 1 To n
244. Dim Next
245. Dim slope = st / (bpp(1))
246. Return New Double() {Slope, Intercept}
247. Else
248. Dim SumX, SumXSquared, SumY, SumXY As Double
249. For I As Integer = start - 1 To finish - 1
250. Next
251. Dim Slope, Intercept, Determinant As Double
252. Determinant = (finish - start + 1)
269. n = dgItems.Rows.Count - 1
270. ReDim Time(n)
271. ReDim Depth(n)
272. For i As Integer = 0 To n
a. Time(i) = dgItems.Item(i).Value
b. Depth(i) = dgItems.Item(i).Value
c. DepthLeng = dgItems.Item(0).Value

273. Next
274. drawGraphs()
275. nl = InputBox("Number of layers", "Input Layer")
276. If nl > 2 Then
a. ReDim Preserve bpp(nl)
b. ReDim Preserve vel(nl - 1)
c. ReDim Preserve slope(nl - 1)
d. bpp(nl) = DepthLeng
  Then
  loadFile(filename)
  End If

277. Next
278. drawGraphs()
279. flagPlot = True
  If i = 1 Then
    PublishOption(BypassProc)
  Else
    PublishItem()
  End If

280. Catch
281. End Try
282. End Sub
283. Private Sub LoadToolStripMe
284. Dim dg As New
285. dg.Filter = "Text files (*.txt)" AllFiles (*.*)
286. dg.ShowDialog()
287. Dim filename = dg.FileName
288. If filename <> "" Then
  loadFile(filename)
  End If
289. End Sub
290. Public Sub openFileDialog()
291. dgItems.Rows.Clear
292. FileOpen(1, filename, 1)
293. Input(1, n) = 1
294. txtN.Text = n
295. Dim jj = 2
296. For i As Integer = 0 To n - 1
a. bpp(i) = dgItems.Rows.Ad
  t="Brea"
  kpoint", 
  kpoint = dgItems.Rows.Ad
  Next
  For i As Integer = 0 To n - 1
b. Layer = 300. Input(1, jj)
  For i = 1 To 300. dgItems.Item(0, i).Value = jj
  Next

j. Else
k. For N = 302. Next
  For As Integer = 0 To n - 1

304. Input(1, jj) = 1 To 305. dgItems.Item(1, i).Value = jj
  Next

306. Next
307. FileClose(1)
308. End Sub
309. Private Sub ToolstripComboBoxB
310. UpdateType(toolstripComboBoxB)
311. End Sub
312. Private Sub toolbarComboBoxB
313. End Sub
314. Dim G As Graphics
315. Dim Hdc As IntPtr
316. type = 0
317. Else
318. dgItems.Columns(1).HeaderValue = "X (m)"
319. Me.Text = "Surface Spread Refraction Data"
320. type = 1
321. End If
322. If flagPlot = True Then
  Then
  PrintToolStripMe
  End If
323. drawGraphs()
324. tblTitle.Text = "ODOGRAPH & ChtW(10)"
325. End If
326. End Sub
327. Private Sub ToolstripButton1_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs)
328. frmSurveyInfo.ShowDialog()
329. Else
330. End Sub
331. If (PrintDialog1.ShowDialog() = DialogResult.OK)
Then
Then
333. bmp = New Bitmap(PictureBox1.Width, PictureBox1.Height)
334. Dim G As Graphics
335. Dim Hdc As IntPtr
336. SendMessage(PictureBox1.Handle, WM_PRINT, hdc, DrawingOptions.PDF, DrawingOptions.PDF)
References


