Evaluation of Some Physics and Atmospheric Models using Newly Developed Servers and Software Application

D.A. Adenugba
Department of Physics
The Federal University of Technology, Akure
P.M.B 704, Akure, Ondo State. Nigeria.

ABSTRACT
The significant distance between growth and backwardness, developed and under-developed is an inch away from good Physics and functioning software package. Using VB.NET forty-one atmospheric and physics models were computed. Four code components and a client package were developed for all the models evaluated. Microsoft DataGridView control (DGV) was customized and its strengths and weaknesses shown in its application to the models. The functionalities in our servers and the codes in the application are adequately reported. Results obtained corroborate existing facts and provide data and information. It is found that DGV is adequate for holding inputs and/or displaying result (s) in tabular form, and for formatting both inputs and outputs to the level of needs. However, it has no inbuilt Physics and Atmospheric formulae, and the required rows and columns for use are to be inserted each time. Software developers will find our accurate and flexible servers salutary in their works. Our application will tremendously assists Physicists, Atmospheric Scientists and other researchers and tutors to quickly estimate the significant parameters treated in this work.

General Terms
Physics, Server, Software application

Keywords
Atmosphere, Microsoft DataGridView control (DGV), Physics, servers.

1. INTRODUCTION
The difference between month and mouth is a single middle letter. Space creates a lucid distinction between again and a gain (profit). This little but vital difference produces outstanding quality which cannot be overlooked. Similarly, static and dynamic creates excellence, which functioning software package gives all the time. Teaching and studying of good Physics, investigating variables of interest in the laboratory and outside the laboratory, as well as their evaluation with an accurate software application guarantees speed and accuracy which manual procedure could not always achieve. The conventional method of chalk and blackboard is static and could not effectively and speedily provide solutions when needed, especially when large number of variables are involved. To create an accurate distinction and uncommon speed in evaluating and studying physics and atmospheric parameters, there is need for dynamic server and application, which good software application could provide.

A code component is a server, and a class library in VB.NET is nothing but a server (Evangelos, 2010) upon which a flexible and accurate client application could be built. He who develops a server provides a foundation for dynamism and growth. He creates a platform upon which unique development and progress; enterprising study and learning and meaningful research could be achieved. Observations had shown that there is no server on some physics and atmospheric models, besides lack of client application for the study and evaluation of these models. The availability of computer and growing number of programming languages demand that all subjects be taught dynamically and any evaluation be performed by pressing buttons.

The purpose of this paper is to evaluate forty-one physics and atmospheric models using a client application developed with Microsoft Visual Studio 2008. Besides, we shall customize Microsoft DataGridView control (DGV) for use and in the process show the strengths and debilities of the control. Specifically, we shall estimate pressure and its related parameters. Different forces including Coulombs and Gravitational forces, as well as duct transmission losses and terrain effects on propagation models will receive adequate attention. Four code components will be produced and employed for developing a client application which could be used to evaluate all the models and their related parameters.

2. PHYSICS MODELS
A class is a collection of related objects which, once correctly defined, can be instantiated several times for use (Evangelos, 2010; Thearon and Bryan, 2010). It contains codes and local variables which are loaded into memory; and a new set of local variables created, when instantiated. A class constructor is a procedure declared as New with or without argument. In all the class libraries developed all necessary variables that should have initial values are initialized in the constructor provided. An abstract class was created for the force models. An abstract class is a class that must be inherited, which is declared with a modified MustInherit keyword. It cannot be used on its own, thus has no independence existence. It is a blueprint for the methods it exposes (Evangelos, 2010). Its provision gives other researchers the privilege to inherit and write

small codes and modify what had been produced without tampering with existing codes.

The derived class, dvCalPhyCls inherited the base class, dvPhysicsCls and implemented all the functions in the parent class to compute all the models in Tables 1 and 3. Since the models considered are well-established models, we shall concentrate our attention on what was done to them in our servers. A package which treats only models without their related parameters is lean. With given values any unknown should be computed accurately and timely too. Examiners are found asking questions requesting for variable of interest while the rest variables are given or to be obtained before further computation. Therefore, due attention will be given to all the related parameters of the models treated. Generally, in calling the derived class the data order is expected to be all numerator(s) first then the denominator(s), else error will occur. Constants are not expected to be supplied as this is handled effectively internally. Also, the number of results to be displayed will be kept low, so as to show many results for the accuracy of our codes to be ascertained.

2.1 Pressure

Pressure, P (hPa) is given by eqn (i) in Table 1. Where F is force (N) and A is the area (m^2), L is length (m) and B is Breadth (m), m is mass (kg), v is volume (m^3), ρ is density (g/m^3), h as height (m). There are other five ways of calculating P depending on *how* F and A are given. For example, if F is given as the product of mass m and acceleration due to gravity, g, P could be computed but not until F is obtained from m and g; for other possible combinations for P, F and A, see Table 1. The Equations column helps us to see the relationship between P, F and A, and their related parameters instantly. All the twenty equations in Table 1 are calculated using the values given and the results obtained were found to be accurate. For instance, Table 2 gives the result for eqn. (i). Figure 1 is the result for eqn. (v), a plot of P against h.

Table 1: Pressure Model and its Related Parameters

Parameters	**Equations**
Pressure, P	(i) $P = F/A$
	(ii) $P = F / (L \times B)$
	(iii) $P = (m \times g) / A$
	(iv) $P = (m \times g) / (L \times B)$
	(v) $P = hg\rho$
	(vi) $P = hg.(m/v)$
Area, A	(vii) $A = F / P$
Force, F	(ix) $F = P.A$
	(x) $F = P (L \times B)$
Mass, m	(xi) $m = PA / g$
	(xii) $m = P(L \times B) / g$
	(xiii) $m = Pv / hg$
Volume, v	(xiv) $v = mgh / P$
Density, ρ	(xv) $\rho = P/ hg$
Height, h	(xvi) $h = P / \rho g$
Length, L	(xvii) $L = F / (P \times B)$
	(xviii) $L = (m \times g) / (P \times B)$
Breadth, B	(xix) $B = F / (P \times L)$
	(xx) $B = (m \times g) / (P \times L)$

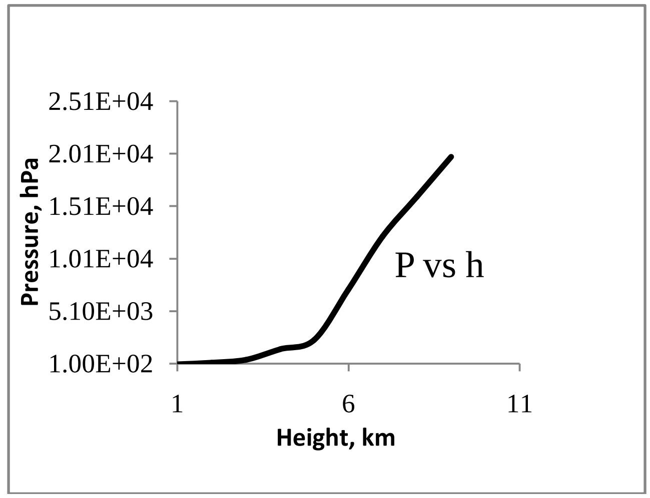

Figure 1: Plot of Pressure, P against height, h

Table 2: Typical Result of Pressure (eqn.(i))

F (N)	**A(m*m)**	**Press(hPa)**
60.00	20.00	3.00
0.34	1.00	0.34
77.00	10.00	7.70
342.00	1000.00	0.34
0.92	0.34	2.71

Table 3: Force Models and their Related Parameters

Force, F	(i) $F = ma$
	(ii) $m = F/a$
	(iii) $a = F/m$
	(iv) $F = mg$
Centripetal Force	(v) $F = mV^2/r$
	(vi) $r = mV^2/F$
	(vii) $m = Fr/V^2$
	(viii) $V = (Fr/m)^{0.5}$
Frictional Force	(ix) $F = \mu N$
	(x) $N = F/\mu$
Coulomb's Law	(xi) $F = k_o q_1 q_2 / r^2$
	(xii) $r = (k_o q_1 q_2 / F)^{0.5}$
	(xiii) $q_1 = Fr^2 / k_o q_2$
	(xvi) $q_2 = Fr^2 / k_o q_1$
Gravitational Law	(xv) $F = Gm_1 m_2 / r^2$
	(xvi) $r = (Gm_1 m_2/F)^{0.5}$
	(xvii) $m_1 = Fr^2/Gm_2$
	(xviii) $m_2 = Fr^2/Gm_1$

2.2 Force Models

Table 3 gives the force models and their related parameters. V is velocity (m/s), r (m) is the distance between masses or charges, μ is the coefficient of friction, N is the normal reaction. Coulomb's constant, $k_o = 1/4\pi\varepsilon_o = 9 \times 10^9$ Nm^2/C^2 and ε_o is space permittivity. The gravitational constant, $G = 6.67 \times 10^{-11} Nm^2/kg^2$. Other terms as previously defined. What Tables 1 and 3 indicate is that for any givens the unknown could be evaluated by calling the appropriate function via specified enumerate class declared for the models. Enumerate class permits meaningful name to be assigned to represent an integer for picking desired items.

2.3 Methods

The dvCalPressure method has two arguments: a reference DGV control which holds the inputs and then the calculated result; the enumerate class declared possesses six members for the first six models in Table 1. Other enumerates are declared for other parameters in Table 1, except gravitational force which uses that declared for Coulomb law since they are identical in structure. However, the message each displays is set through a property, and the enumerate class assists us to call appropriate function to compute result for the specified model and return the result in a DGV control. The reason why we split the models into units is for ease of maintenance in the future.

The dvCalCoulombs method, for computing Coulomb law, also has two arguments just like that for pressure, but with only two members in the enumerate class for picking either Coulomb or Newton gravitational force of attraction. Tables 4 and 5 are the results for selecting the two members of the enumerate class for Coulomb and gravitational force respectively. The dvForceDiv method has three members for dividing up to three numbers. The first overload assumes a constant denominator number which should be supplied through a property. If not supplied before calling the method, the default value will be used. It was used to compute eqns. (ii), (iii) and (x) in Table 3. The order of input data supply is very important also here: numerator(s) should be supplied before denominator(s), else wrong result will be obtained.

Table 4: Coulomb force result (CR)

q1	q2	r	CR
3.00E-06	5.00E-06	0.20E+00	3.38E+00
0.35E+00	0.10E+00	0.10E+00	3.11E+10
1.20E+00	1.00E-11	1.00E+00	1.08E-01
3.00E+00	2.11E+00	3.50E+00	4.65E+09
1.60E-19	1.60E-19	5.30E-11	8.19E-08
5.00E-06	0.03E+00	0.55E+00	5.06E+03
1.00E-07	4.23E+00	1.32E+00	2.19E+03
8.00E-06	5.00E-06	0.30E+00	4.00E+00

Table 5: Gravitational force result

m1	m2	r	F(N)
1.00	1.00	0.1	9.00E+11
5.00	3.30	1.5	6.62E+10
10.00	0.55	0.6	1.38E+11
0.15	0.90	0.5	4.88E+09
100.00	80.00	5.0	2.88E+12

The dvForceTimes and dvForceMinus methods for multiplying and subtracting three numbers respectively have similar structure as dvForceDiv method. dvForceMany method, unlike the previous methods, accepts and returns multiple results. It has a reference DGV control for inputs, and after computation the result is returned in the last column of the same input control. The second argument is an enumerate class which permits the method to determine which function to call among dvForceDiv, dvForceTimes and dvForceMinus methods. The column of data inputs in the DGV control determines which member of the methods to call. For instance, when three inputs for eqns. (v) - (viii) in Table 3 are supplied, and the correct enumerate specified, the correct result is returned in the last column with absolute ease.

3. ATMOSPHERIC MODELS

A class library, dvClsAtmos was prepared for the atmospheric models considered in this work. It was separated from Physics models class for ease of maintenance. Their choice is based on non-availability of server on them. There is need to develop a dynamic link library for them, so that developers will be intrigued to produce a software package for the use of the scientific community using the server which will hasten their work.

3.1 Duct Transmission Losses

Losses in duct propagation, L_b is related to that of free space path loss, L_{bf} by [CCIR, 1982]

$$L_b = L_{bf} - 10\log(dd) + A \qquad (1)$$

Where dd is the distance within the duct. A is a factor responsible for other attenuation mechanisms such as leakage losses due to duct irregularities or losses due to ground reflection and so on. The free space path loss is related to frequency, f (GHz) by [CCIR, 1982].

$$L_{bf} = 92.44 - 20\log(d) + 20\log(f) \qquad (2)$$

where d is the propagation path, not that within the duct (dd). The two methods in our server are overloaded for computing free space path loss and duct transmission losses respectively. When object arguments are sent in, free space path loss and duct transmission losses will be evaluated and a single result will be returned. However, the second overload has a reference DGV which holds both the inputs and output. Tables 6-8 are the typical results for free space path loss, duct transmission loss and combination of the losses when dd= 0.5 and A= 0.8. Where f is frequency, PPath is propagation path and FreeSPL is free space path loss, SDuctTL is space duct transmission loss.

Table 6: Result for free space path loss

f	PPath	FreeSPLoss
10	5	98.46
15	8	97.90
30	20	95.96
40	30	94.94
50	40	94.38

Table 7: Typical Space Duct Result

f	Dist	SDuctTL
10	5	1.02E+02
15	8	1.02E+02
30	20	9.98E+01
40	30	9.87E+01
50	40	9.82E+01

Table 8: Free space path loss and Duct Result

f	Dist	FSPLoss	SDuctTL
10	5	9.85E+01	1.02E+02
15	8	9.79E+01	1.02E+02
30	20	9.60E+01	9.98E+01
40	30	9.49E+01	9.87E+01
50	40	9.44E+01	9.82E+01

3.2 Terrain effects on propagation

Terrain, such as hills, buildings, water and trees, of a given area is a potential threat to propagation signal. Microwave paths with reflective terrain could cause second signal to the receiving antenna leading to loss and variation of receiving signal. Diffraction loss, otherwise known as obstruction loss, could also occur due to obstruction of the propagation path by terrains. It has been established that this type of loss increases with decreasing path clearance from zero loss to about 10dB for zero clearance (ITUR, 1989). For a path of approximately $0.6F_1$, diffraction loss is zero. F_1 is first Fresnel zone radius, and the nth zone is given by (ITUR, 1989).

$$F_n = (n\lambda d_1 d_2 / d)^{0.5} \quad (3)$$

Where d_1, d_2 are distances from the link terminals (km), d is the link length (km), λ is the wave length (km) and is given by c / f, f is frequency (GHz), c is the speed of light and its square root is 17.321 km. When n=1, we obtain

$$F_1 = 17.321((d_1 d_2 / f.d\ )\ \text{^}0.5) \quad (4)$$

Two overload methods were written for F_n for single and multiple results. The method that returns single result gives details of the workings as seen below:

Data Used
Fresnel Zone, n=2
f = 18: d1 = 24: d2 = 24: LinkLength = 22
Fresnel Zone Formula
Fn = 17.321.[((n.d1.d2)/(f.d))^0.5]
First Fresnel Zone,n=1
F1=17.321.[((1.24.24)/(18.22))^0.5]
F1=1.21E+00
Other Fresnel Zone(s), n=2
F2=17.321.[((2.24.24)/(18.22))^0.5]
F2=1.71E+00

The multiple results method has reference DGV control argument for both inputs and output. Table 9 is a typical result for the values shown.

Table 9: Fresnel Zone Result

f	ProgL	d1	d2	n	FN	FN1
10	5	10	5	1	1.00E+00	1.00E+00
20	10	15	10	2	1.22E+00	8.66E-01
30	20	25	15	3	1.37E+00	7.91E-01
40	30	30	20	4	1.41E+00	7.07E-01
50	40	35	25	5	1.48E+00	6.61E-01
60	50	40	30	6	1.55E+00	6.32E-01

4. CUSTOMIZING MICROSOFT DATAGRIDVIEW CONTROL

Microsoft DataGridView control (DGV) is a powerful intrinsic control that could be employed to present data and result in tabular forms. It possesses several functionalities that could be utilized and further be customized for use. In Table 10 there are three sections and sixteen methods which names are italiced and the task perform by each of the customized method given. By polymorphizing the methods the name of the method is drastically reduced and analysis is made easy, since the methods differ only by number of arguments or type of data type. These methods are in a class library named dvDatGrdViewCls. To use the class, it has to be instantiated with new keyword like that (dgv) used in our work. Note that dgv is correctly instantiated at the class level, since it is used in several places.

Table 10: Customized Properties and Methods for DGV Control

Section 1: Formatting Properties and Methods

dvChgCellDGVBackColour. It formats the background colour of a DGV control cell, using the column index and colour supply.

dvChgCellDGVForeColour. It formats the fore colour of a DGV control cell, using the column index and colour supply.

dvChgCellDGVFont. It changes a cell font of a DGV control.

dvChgCellDGVBackForeColour. It formats the back-and fore-colours of a DGV control cell,using the column index and colours supply.

dvChgCellDGVBackFontColour. It formats the back colour of a DGV control cell and changes the font, using the column index, colour and the font supply.

dvChgCellDGVBackForeFontColour. It formats the back- and fore-colours of a DGV control cell and also changes the font, using the column index, colours and the font supply.

dvChgCollDGVBackColour. It changes the back colour of a supply column (not just a single cell only) of a reference DGV control.

dvChgCollDGVForeColour. It works as the one for back colour, but it changes the fore colour of the supply column.

dvChgCollDGVFont. As above but alters the font of the supply column (not a single cell).

Section 2: Minimum and Maximum Values Methods

dvGetMinValue. It has four overloads members for retrieving minimum value from a DGV control and formatting the identified value. The back colour, fore colour and font, as well as mixture of these could be altered to meet different needs. This is achieved through colour enumerate class defined for it with eight members by calling formatting methods giving above.

dvGetMaxValue. It has four overloads members for retrieving maximum value from a DGV control and formatting the identified value. Other tasks as in dvGetMin method.

dvGetMinMaxValue As in dvGetMinValue and dvGetMaxValue methods above for retrieving minimum and maximum values from a DGV control and formatting the identified values. Other tasks as in dvGetMin method.

Section 3: Other Methods

dvAddRowToDatGrdView. It adds row(s) to DGV control using the integer number of rows required; and returns the index of the last row(s) added. There are five different types with the same name, but varying number of arguments for adding row(s) to control at specified places.

dvGetRow_CollDatGrdV. It returns the number of rows and columns in a DGV control. It has five members.

dvLoadDatGrdVRowValue. It loads an existence specified row with data from a collection object. If the specified row does not exist, it gives an informed message and gracefully exits the method. If no row is supplied, the second overload is summoned and the data will be loaded in the last row.

dvCopyDatGrdView. It copies all the columns of the source DGV control into a destination DGV control. The second overload is called to copy, filter space and insert column header in the destination DGV control.

The DGV control item property departs from the usual row-column known for populating tables when indexes are supplied. Through the item property you can set and get content of a cell, but remember that the column index comes first not row. For instance, dgv.item(column ,row).value = 10 will insert 10 (or any object) in the specified column and row location of instantiated DGV control (dgv). Apart from two indexes, a named string of the column could be supplied along with the row index to set or retrieve what is required.

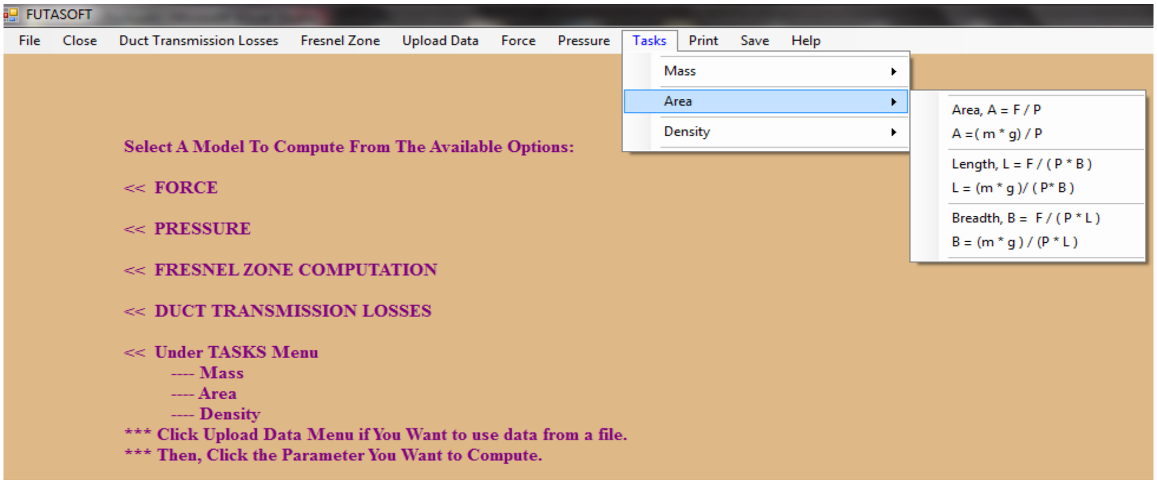

Figure 3: FUTASoft Interface

5. DISCUSSION

The equations treated in this work are fundamental to the study of physics and evaluation of atmospheric prediction models. Results obtained corroborate existing facts and show the accuracy of our codes. Instructions are provided at every point (see figure 3), so user of our package is not left in doubt as to what to do. Besides, our menu text is instructive, showing what could be achieved using it.

Microsoft DataGridView control (DGV) enables multiple inputs to be sent in for computation and from where a row of data could be read. Thus, it is compact for several inputs. It has capability to display results in various formats with attractive colours. In addition, by customizing DGV control, we observed that it drastically reduces the length of codes in all our methods and functions; assists us to quickly produce our methods, as well as functions and sub-procedures, apart from its reliability having been tested and found accurate. Indeed, it is a robust, useful and powerful control that could be employed for scientific uses. There are, however, no inbuilt Physics and atmospheric formulae in the DGV control. Besides, the knowledge to adeptly manipulate the rows and columns (which are zero-based) is required for successful use. By customizing DGV, we are able to extend its functionalities for formatting, retrieving rows and columns, copying, loading data and retrieving minimum and maximum numbers and formatting them for emphasis. DGV allows both input(s) and output(s) to be displayed with ease as seen in all the tables presented. Results obtained were compared with the results from

Microsoft Excel application and previous results and found to be accurate. For instance, the first, fifth and last data in Table 4 (bold font for emphasis) are from Frederick and Eugene (2006), and found to be in agreement with ours.

Colours and fonts could be used to emphasis facts. To colour the cell of a DGV control our properties are very handy. Back- and/or fore- colours and font could be applied to DGV to achieve desired effects. The effects obtained were obtained separately from computing the result. After obtaining the results, Effects menu was clicked to display seven submenus each in turn has three submenus from where we obtained our results. Although DGV control does not save the customized attributes with it, it could be employed to underscore salient facts real time when teaching or learning.

6. CONCLUSION

Forty-one well-established physics and atmospheric models have been evaluated using developed servers and a client application. Microsoft DataGridView (DGV) control has been customized for use and the functionalities generated function to specifications. However, the customized formatting attributes of the DGV control could not be saved along with the data. It could, however only be utilized to emphasis facts and figures real time. Also, it could conveniently hold hundred thousand rows of data and results, our application limit, which is sufficient for Physics and atmospheric use. DGV control permits data and results to be viewed together thus hastening comparison and analysis of results. The client application that could be used by all and sundry with little knowledge of the computer is flexible and accurate as computed results depicted when compared with other package results. The four accurate servers upon which our package, FUTASoft is based are carefully and entirely modularized for ease of maintenance and flexibility of call and use. Our dvClsAtmo class library is expected to be put to full use with real measured data. Microsoft DataGridView control (DGV) should be further customized for more scientific use and its robust capacity tapped to the fullest.

7. REFERENCES

[1] Evangelos Petroutsos. 2010. Mastering Microsoft Visual Basic 2010. Wiley Publishing Inc.

[2] Frederick J. Bueche and Eugene Hecht. 2006, College Physics. McGraw Hill. 10^{th} ed. 241-251.

[3] Thearon and Bryan. 2010. Beginning Microsoft Visual Basic 2010. Wiley Publishing Inc.

[4] CCIR, Propagation in Non-ionized Media, Recommendations and Reports, XVth Plenary Assembly- Vol. V. 1982.

[5] ITUR-R (453-9). The Radio Refractive Index Its Formula And Refractivity Data, 2003.

[6] ITUR-R. Int. Rep. IC/89/23. Physics of the Tropospheric Radio Propagation. 1989, 1-54.