Spreadsheets in Education (eJSiE)

Volume 6 | Issue 1 Article 2

December 2012

Dynamic Surface Charts for Scattered 4-D Data in Excel Spreadsheets

Daniel Hsieh
Rutgers University - New Brunswick/Piscataway, daniel.x.hsieh@gmail.com

Vikas Nanda
Rutgers University - New Brunswick/Piscataway, nanda@cabm.rutgers.edu

Dynamic Surface Charts for Scattered 4-D Data in Excel Spreadsheets

Abstract
Visualizations that are low-cost in memory are desirable. We present a method for stitching three-dimensional scattered data from multiple worksheets into a dynamic "animation-like" surface chart in Excel. This method is useful when (1) the user hard-codes the data points to conserve memory; employing such strategy scales better than soft-coding data values, (2) the data values are hard-coded by an unknown source, or (3) the function is complex and requires a user-defined function to output values into cells. In particular, we demonstrate an application in biology where rigid motion (rotation and translation are the only transformations applied to an object in 3-D space) is used to model the free energy gain/loss by surveying various placements and orientations of membrane proteins with respect to their environment. Our strategy involves a simple concept of scrolling through an order of worksheets, and can be extended to even more dimensions (i.e. scrolling through workbooks if necessary)

Keywords
dynamic surface charts, visualization, animation

Dynamic Surface Charts for Scattered 4-D Data in Excel Spreadsheets

Daniel Hsieh
BioMaPS Institute for Quantitative Biology
Rutgers, The State University of New Jersey
daniel.x.hsieh@gmail.com

Vikas Nanda
University of Medicine and Dentistry of New Jersey
nanda@cabm.rutgers.edu

Abstract

Visualizations that are low-cost in memory are desirable. We present a method for stitching three-dimensional scattered data from multiple worksheets into a dynamic "animation-like" surface chart in Excel. This method is useful when (1) the user hard-codes the data points to conserve memory; employing such strategy scales better than soft-coding data values, (2) the data values are hard-coded by an unknown source, or (3) the function is complex and requires a user-defined function to output values into cells. In particular, we demonstrate an application in biology where rigid motion (rotation and translation are the only transformations applied to an object in 3-D space) is used to model the free energy gain/loss by surveying various placements and orientations of membrane proteins with respect to their environment. Our strategy involves a simple concept of scrolling through an order of worksheets, and can be extended to even more dimensions (i.e. scrolling through workbooks if necessary)

Keywords: Dynamic surface charts, animation, scrollbar binding, visualization

1. Introduction

We show how to create a four-dimensional "dynamic" surface chart by binding a scrollbar to a surface chart. The trick is to bind the value of a scrollbar to a property of the sheet relating to its name, which implies the sheets need to be in some countable order, but there are more details to cover in this walkthrough. This proof-of-concept of generating three dimensional snapshots of a four-dimensional scattered dataset will be invaluable to all fields that analyze data whose dimensions exceed the three we are normally comfortable with.

Multidimensional analysis and visualization are still some of the most underdeveloped features even in the latest Microsoft Excel compared to those of peer math/statistics software such as Mathematica, MATLAB, R, SAS, SPSS and ParaView. Yet, they are highly demanded skills in education and technical fields [1]. Visualizing three dimensional data often involves the surface chart or a contour plot. As suggested by the initiative and contributed articles of Spreadsheets in Education [2,3,4], forum discussions [5,6], technical blog posts [7,8,9] and webpages [10,11,12], the ever-growing user community has also long acknowledged Excel's potential in multidimensional math and science education in addition to its well-known roles in financial analysis [13,14].

There are two ways to building tools you need to analyze multidimensional data in Excel. The first is to invest money in professional third-party software (i.e. Nevron, ComponentOne, Infragistics, etc.) These software are the result of programmers, businesspeople and mathematicians who recognize that 1) this demand is heavy in business- and industry-side analytics and that 2) creating a toolset that is compatible with Excel and other Microsoft Office products most likely improves workflow and therefore productivity.

The second is to invest time in learning the programming. A reward for doing this is being able to create a tool catered to your own needs. Microsoft has provided a programming language within Microsoft Office Applications such as Excel, called Visual Basic for Applications (VBA). Learning this language not only helps minimize tedium through automation (i.e. macro recording and replaying), it can also help create interactive spreadsheets and complex forms, extend functionality by calling Windows-native and third-party software functions, deploy web services, and more [15].

Microsoft has developed tools to more easily develop and sell applications/software beyond those third-party software specialized in extending Office automation. This foresight in the early 2000s brings forth two important products to those interested in getting their feet wet in programming/software engineering: the .NET framework and Visual Studio. The .NET framework is a software development framework that facilitates programming as well as development of software across multiple operating systems and web applications across multiple browsers. Visual Studio is a free integrated development environment – it is a software that helps the developer organize and test run the layout (form) and the source code behind (function) of his/her developing product. Visual Studio 2008/2010 Professional, along with other professional-edition software, is also freely available to students with a valid school email address from Microsoft's DreamSpark website [16]. Both of these tools are thoroughly documented [17, 18].

With the introduction of these two main products, Microsoft offers VBA developers a .NET-based alternative for automating Office called Visual Studio Tools for Office (VSTO) [19]. VSTO seamlessly combines Office and desktop development, thus expanding the boundaries of software development. However, because VBA has been integrated deeply into all kinds of business and industrial systems, Microsoft plans to keep VBA in future shipments of Office [20, 21].

Even with Office automation raised to a whole new level, one of the limiting factors in promoting creative visualizations is the charting toolset itself. Chart Controls for .NET [22] help extend creating the same type of charts for more environments such as web services, but does not further improve the flexibility and types of charts. It cannot be emphasized more that visualizing multidimensional data is a skill of increasing demand within educational and professional settings. How can students learn to use these tools if they don't know what the tools are used for? In order to address this Catch-22 situation, the Excel (and Office) development team need to consider a strategy in making data analysis and visualization tools easier to discover and access in order to attract and motivate students towards learning the higher

math, statistics and sciences necessary to perform such skills within a multidisciplinary technology-driven era.

Our work adds to a suite of complex visualization tools that can be built by Excel programming but do not yet exist conveniently for its users as pre-built Excel features. To demonstrate the fact that Excel spreadsheet-based multidimensional analysis is useful in the sciences, we shall first review some prerequisite visualization concepts and present an application in biology. A walkthrough is then provided to those interested in understanding how to setup the visualization.

2. Application: A Depth-dependent Energy for Membrane Protein Insertion

2.1. What is Energy and What Qualifies it as a Multidimensional Problem?

Energy is an indirectly observable quantity that allows a physical object, living or non-living, to exist in one state and be in a different state. There are many types of energy, such as: kinetic, potential, chemical, electrical, magnetic, sound, and nuclear. In physics, energy is often described as the ability for a system to do work. Work is directly computable as a function of forces applied to the system. Depending on the factors that are assumed to contribute to the work a system can do (or the work done on a system), one can calculate the change in energy through the difference of two observable states.

For the sake of simplicity, let's take the gravitational potential energy as an example.

$$\Delta E = mg\Delta h \qquad (1)$$

What are we calculating? The difference of potential energy between two states: an object at a certain height, and the same object at a different height. Our model assumes two things: 1) we are on Earth and therefore using the gravitational constant associated with Earth, 2) the object is uniform in composition and has the same material throughout (this characteristic is called isotropy) and therefore, can be treated as a point in space. If we rewrite this potential energy as a function of the remaining one variable (parameter), height, then we have:

$$\Delta E(h) = mg\Delta h \qquad (2)$$

This is an example of depth-dependent potentials. With a function of one variable, we can draw a classic two-dimensional graph of ΔE as a function of h. If we allow our model to further account for varying mass assuming the same gravitational constant, we have ourselves a three-dimensional problem because there are two independent variables, mass of the object and its height:

$$\Delta E(m\ ,\ h) = (\Delta m)g(\Delta h) \qquad (3)$$

If you are contemplating how to draw a three dimensional graph of this potential function, draw a Cartesian plane on a piece of paper or chalkboard. Label the axes: mass and height. The logical thing for the value of this potential at any coordinate pair $(\Delta m, \Delta h)$ would be the vertical height away from the Cartesian

plane. The resulting graph you would be observing is a three-dimensional one, called a surface chart, and is already difficult for humans to draw.

The way we resolve the problem of being unable to draw in 3-D space is to draw 2-D projections of a three-dimensional object. When you observe an artist trying to draw a 3-D object on a canvas, this artist is visualizing what the 3-D object looks like from a certain perspective and drawing that mental snapshot. If the reader is interested in the theory behind the proper construction of these 2-D projections of the 3-D object (in this case, a surface chart), we encourage the reader to study Banecka's recent article [4] and to further research on the topic of projective geometry.

Let us return to the gravitational potential function. What if the gravitational constant is observed on many celestial bodies besides the Earth? Then we will have introduced a third variable into the equation:

$$\Delta E(m, g, h) = (\Delta m)(\Delta g)(\Delta h) \qquad (4)$$

Now, not only do we have to draw 2-D projections of the 3-D surface chart, there is a third variable in play making the 3-D surface chart change. The entire surface changes because of a unit change in this third variable. One way to visualize this is to draw multiple surface charts for each value of the third variable. Depending on the continuity (and the interval step size) along this dimension of this third variable, and therefore of these snapshots, it may benefit the person visualizing the dynamic surface chart as either a movie (small step size) or a collection of snapshots. We will demonstrate how to draw 3-D projections of a 4-D chart in Excel. The figure below shows that we can easily generate such an animation chart using a simple formula for the cell range E2:O12 and binding the value of g to the scrollbar (Fig. 1).

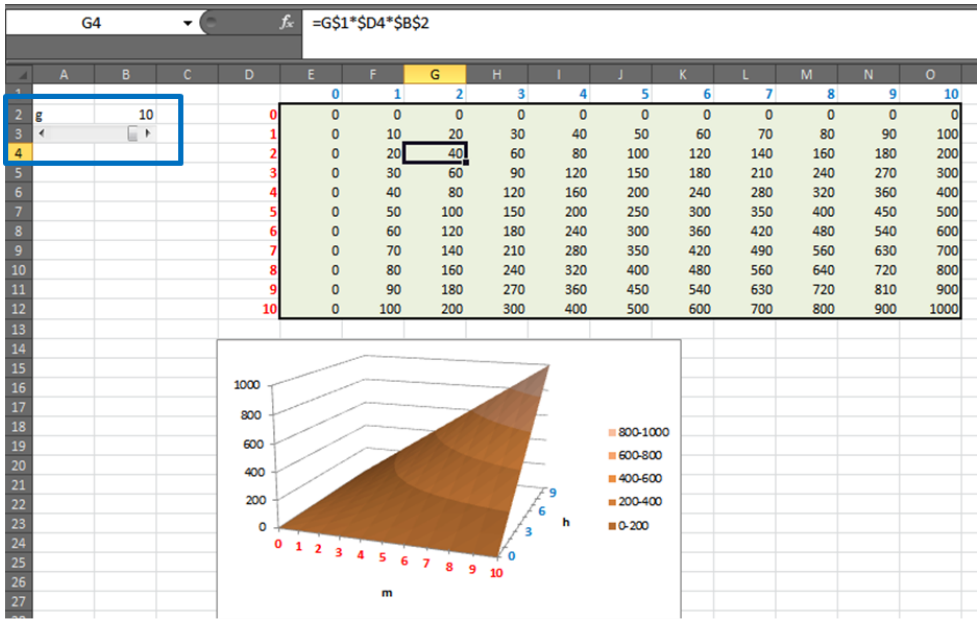

Figure 1: Binding a scrollbar with a surface chart of $\Delta E(m, g, h) = mgh$ requires linking the scrollbar to a cell value, then using the cell value inside each cell formula of the surface data we plan to change. Although one can survey different combinations of masses (row values in red) and heights (column values in blue) coupled with varying gravitational constants from 1 to 10 (scrollbar linked with cell B2, boxed in dark blue).

If the reader seeks more practice material for producing a dynamic surface chart based on cell formulae, an in-depth tutorial involving two scrollbars is provided [23].

2.2. Membrane Proteins in Medicine and Nanobiotechnology

Membrane proteins are very important molecules – they are groups of atoms arranged in some order and connected to be functional units in biology. They float dynamically within the cell membrane, a biological envelope that separates the contents of the cell from the rest of its surrounding environment. The membrane proteins we have studied perform functions such as transporting nutrients, transmitting signals from the outside of the cell to the inside (and vice versa), and attacking cells that are foreign and considered threatening to itself (this harmful attack is called virulence) [24]. These are the same proteins behind the deadly and virtually untreatable "superbug" called methicillin-resistant *Staphylococcus aureus*, also known as MRSA (pronounced "mersa") [25]. These membrane proteins are responsible for rejecting the drugs scientists have been developing.

While membrane proteins in MRSAs pose threats to humans, they are also exciting candidates in nanobiotechnology. The properties of these membrane proteins can be tweaked for other applications such as biofuel production and detection of chemicals present in bodies [26]. A necessary step is to identify amino acids in the sequence that can tolerate the mutation while preserving the structural integrity of the protein. For the membrane proteins found in MRSAs, functions that assess how well the protein incorporates into natural membranes have been developed [27, 28]. Depending on the type of amino acid in the protein sequence and its depth in the membrane, the function yields different energies. To derive the total energy of the protein, we simply calculate the sum of the amino acid energies. Section 6 is a technical appendix for these depth-dependent energy calculations.

2.3. Membrane Proteins in a Lipid Environment

When each amino acid in a membrane protein sequence has an energy that depends on its depth in the membrane, one can see that rigid motions of the entire protein (i.e. translations and rotations) yield different corresponding values of insertion energies.

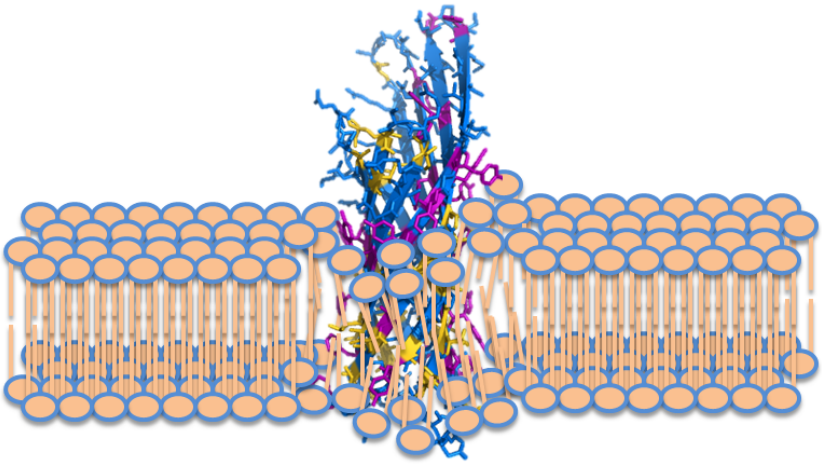

Figure 2: A schematic of a membrane protein situated in a lipid environment

The main component of the environment, the lipid molecule, is composed of water-loving and water-avoiding components. When placed together in water, a result of many lipid molecules is that the water-loving sides will still interface with water molecules while water-avoiding components will be shielded away from interaction with water. This leads to the formation of a bilayer. When a membrane protein is introduced to this environment, the lipids adjust their water-loving components called headgroup regions (blue ellipses) to match placement of amino acids that best interact with them (colored purple). These particular amino acids tend to interact with the headgroup regions of the lipid bilayer, thus readjusting the positions of certain lipid molecules and deforming the overall bilayer (Figure 2). Thus, it is not hard to imagine that the membrane protein has limited orientation that achieves the best energy given the lipid molecules do not escape too far as to break the bilayer itself. What are the favorable rigid motions of the protein with respect to the center of the lipid environment?

We have developed a way to visualize all insertion energy values from applying rigid motions (this visualization is called the energy landscape) by creating a dynamic Excel surface chart tool. Briefly, the energy landscape data is arranged in multiple worksheets in the order of the membrane protein's depth with respect to the center of membrane, and the data of energy values from rotating the protein at a fixed depth is located within that corresponding worksheet. Chapter 2 explains how one can set up a similar visualization for four-dimensional data that can be sliced along one dimension.

2.4. Visualizing the Rigid Motion of Membrane Proteins using Dynamic Surface Charts

One way to visualize the orientation of proteins in membranes is through surface charts, which convey the information of energy landscapes. The 3-d orientation of a membrane protein can be expressed as a combination of rotations by x- and y-axes. The resulting energy from that orientation is the value of that surface

chart corresponding to the orientation coordinates. A static surface chart cannot suffice in communicating 4-d data of the insertion energetics because we also need to visualize depth information. Hence, some sort of animation, with the depth as the fourth dimension, is needed. A solution in setting up such an animation involves indirect binding of a scrollbar to a property of the multiple sheets of scattered data (Fig. 3).

Here we provide a walkthrough and the thought process behind setting up this scrollbar-based solution. The code is provided in Excel files with the suffix "_withCode". The code (text only) is in the .bas file. Users with versions lower than 2003 will most likely not be able to implement or view this workbook properly.

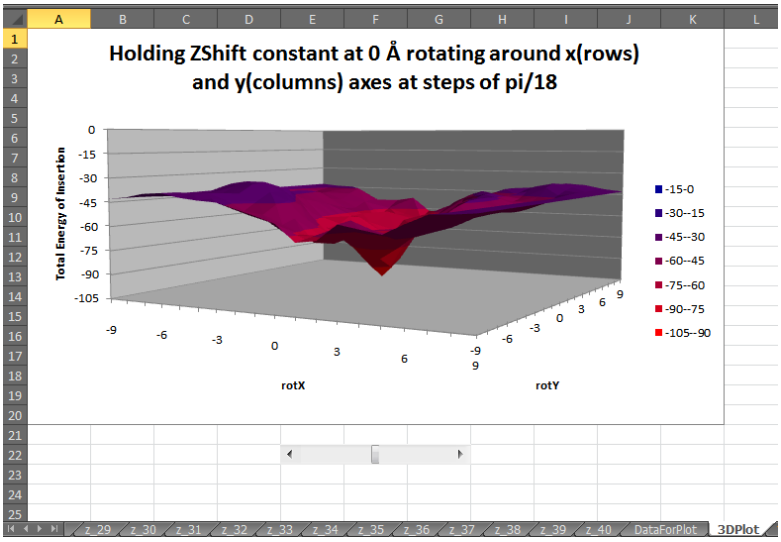

Figure 3: Energy landscape generated by sampling rotations about x- and y-axis of a protein at given depth in membrane. Here, a depth value of 0 Å corresponds with the protein center of mass situated in the center of the membrane. Binding a scrollbar allows one to visualize multiple landscapes along the fourth dimension of depth.

3. Setting up the Visualization Example

3.1. Prerequisites and Assumptions

This project contains VBA (Visual Basic for Applications) code that requires knowledge only to change the visualization, not run it. We assume the reader has access to either Microsoft Excel 2010, 2007, or 2003 for PC. If the reader has Office for Mac, versions 2011 should be able to run VBA, but not 2008, which omitted VBA as a feature [12]. Versions 2003 and 2004 will require a converter.

We will be using the terms "macro" and "VBA" frequently. Here is the relation between the two terms: a macro is computer-generated code in VBA, which is a special subset language of Visual Basic that manipulates Excel, Access, PowerPoint and other Office applications. Since the macro is computer-generated, it is less elegant to read longer macros. However, for those learning to program in VBA, recording a macro helps organize procedures into programming logic [2].

The terms "subroutine" and "function" will also be used frequently in this walkthrough. A subroutine can be procedural but does not return a value for other blocks of code to use. A function is a subroutine that does return a value, but cannot run on its own.

What we will be doing is discuss the big picture behind the VBA code of the provided example. For those interested in learning VBA at beginner to intermediate level, we recommend the "Power Programming with VBA" series for versions 2003 and 2010 by John Walkenbach [29, 30].

3.2. Enable Macros and Display Developer Tab (PC version)

We assume the reader has not yet set preferences with regards to macro security. By allowing macros in this worksheet, you have access to viewing, running and editing the code running behind a worksheet.

Open file "DynamicSurfaceChart_withCode.xlsm" and choose "Enable Macros". If you are using 2003, please open the .xls version of the file and proceed in a similar fashion. You may find more info on macro security settings on the Microsoft Office site [31, 32].

Next, we display the Developer Tab, which is a menu for easier access to the VBA Editor, macros and other programming/design tools.

Go to File→ Options→ Customize Ribbon→ Customize the Ribbon (on the right side)→ Choose Main Tab. Check the box that says "Developer" to reveal the Developer tab. For Excel 2007, go to File→ Options→ Popular. Check the box that says "Show Developer tab in the Ribbon".

Enable Macros and Display Developer Tab (Mac version):

In Excel 2011, choose the "Enable Macros" option upon opening the file "DynamicSurfaceChart_withCode.xlsm". For Excel 2004, you will need open the file "DynamicSurfaceChart_withCode.xls." and go through a similar process. If you choose the file with "_withCode", you may skip section 3.3.

For showing the Developer Tab in 2011, go to Excel→ Preferences→ Ribbon. Under "Show or Hide Tab or drag them into the order you prefer", scroll down the list and check "Developer".

3.3. Import VBA code into a VBA module

Once macros are enabled for Excel, we are ready to import the provided VBA code and test run the visualization. Excel allows its users to export VBA code to a ".bas" (BASIC) file type for ease of transferring to other Excel applications [33]. We did exactly that so that the reader can import the code from a .bas file into a VBA module. If the reader is unable to load the .bas file due to technical reasons, we have prepared a second version of the Excel workbook that comes with the VBA code preloaded called "DynamicSurfaceChart_withCode.xlsm". If you are using this particular version of this workbook, you may skip the remainder of this section and proceed to section 3.4.

We will need the two files for this exercise: "DynamicSurfaceChart.xlsx" or ".xls" (for 2003 and 2004 versions) and "DynamicSurfaceChart.bas". Open DynamicSurfaceChart.xlsx or .xls. If you have enabled macros, then you shouldn't see notifications regarding macro security. If you have not done so, you should check the option to allow macros for this file. You should only see sheets labeled "z_-40" to "z_40", a total of 81 sheets in numerical order (Fig. 4).

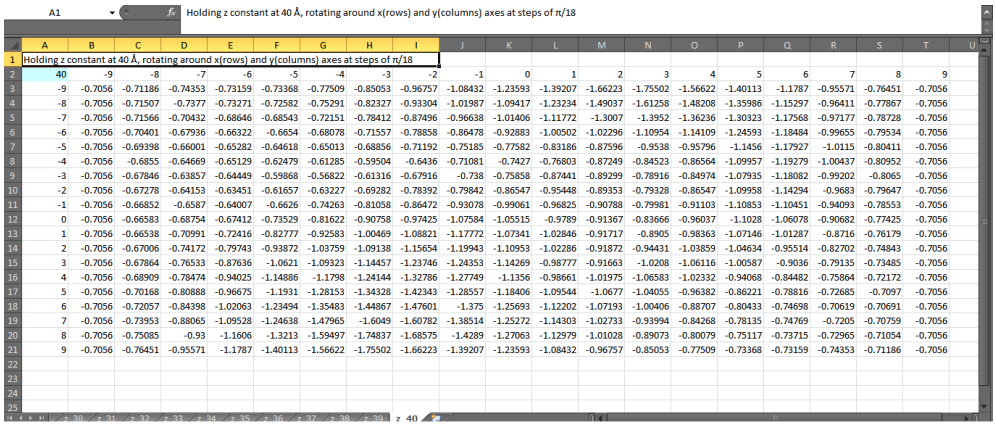

Figure 4: You will see only Excel data (of this format) when you first open DynamicSurfaceChart.xlsm

3.4. Running the VBA code in the Module "DynamicSurfaceChart"

When in Excel (make sure the application is the active window), press Alt+F11 to open up a new window called the Visual Basic Editor (For Mac users, Alt+Fn+F11). This editor will be the environment in which you code and test your VBA and macros. (Fig. 5) If you don't see the panels in Figure 8, go to View and select the Project Explorer, Properties Window, Immediate Window, and Locals Window. The Project Explorer gives you a view of all workbooks, worksheets and associated VBA code/macros in a hierarchical structure. The Properties Window allows you to view and modify properties of selected object. (workbook, worksheet, code) The Immediate Window is used to monitor outputs alphanumerical data from running test code. The Locals Window can help you track changes to variables when you run the code line by line.

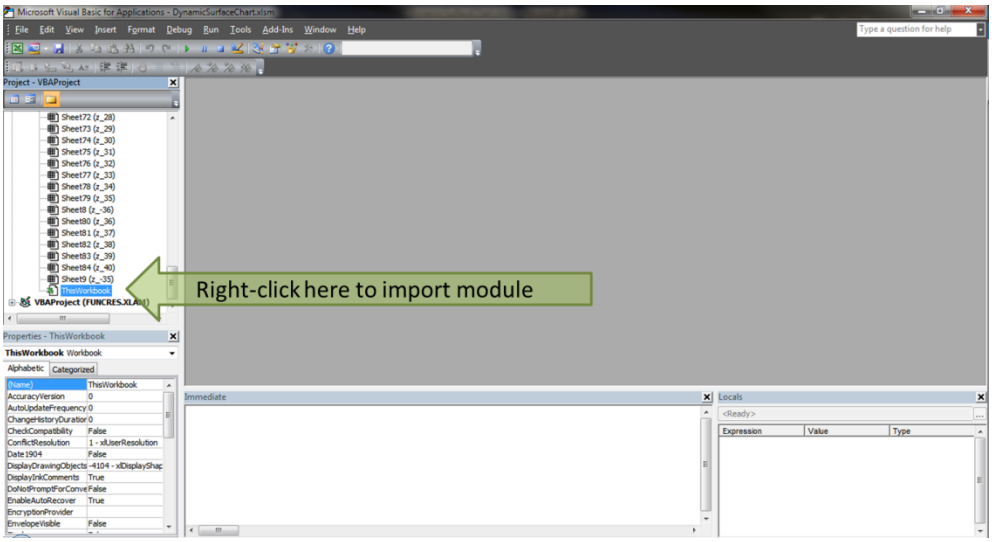

Figure 5: the Visual Basic Editor after opening up the four panels

In the Visual Basic Editor, right-click the "ThisWorkbook" icon (highlighted blue in Fig. 5) and select "Import File…" Locate and select the DynamicSurfaceChart.bas file for import. You should now have a new folder called Modules containing a VBA module called "DynamicSurfaceChart". If you do not immediately see the code, double-click on the DynamicSurfaceChart module. Click once inside the "test_S3DP()" subroutine as shown (Fig. 6):

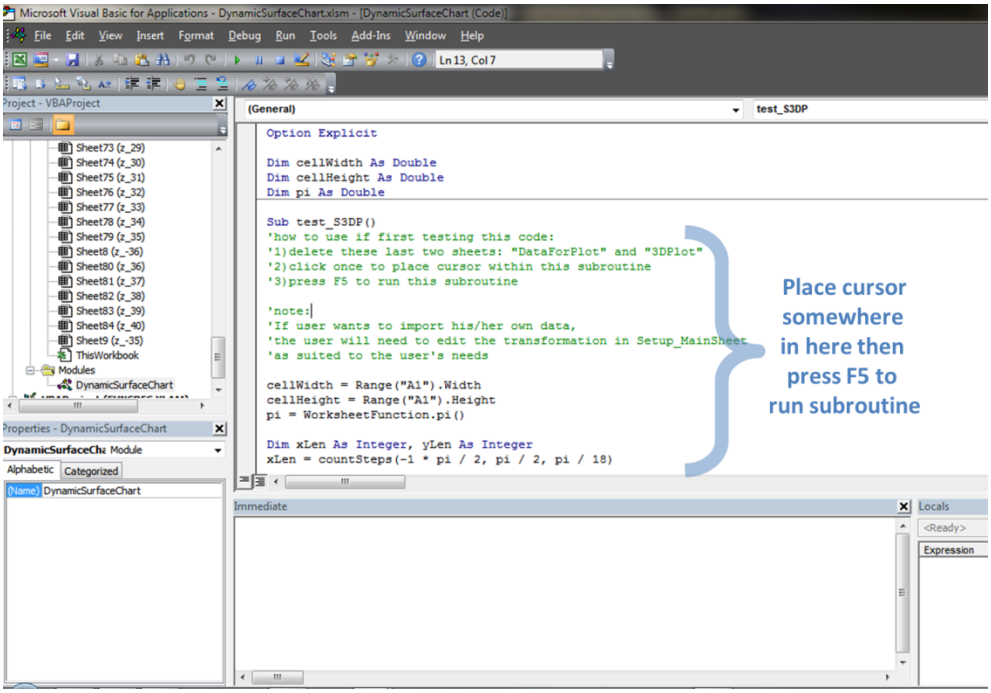

Figure 6: A view of the DynamicSurfaceChart code in the Visual Basic Editor.

Press F5 to run the "test_S3DP" subroutine.

For Mac users, go to Developer→ Macros→ run "test_S3DP" or Developer→ Editor→ place cursor in "test_S3DP" and press F5.

After the code is finished running, you should see two new sheets created called "DataForPlot" and "3DPlot". The worksheet "DataForPlot" contains the dynamic range dependent on the scrollbar value in "3DPlot". Go to the worksheet called "3DPlot" (as shown in Fig. 3) and manipulate the scrollbar around. Notice that when the scrollbar value is changed, the surface chart is updated and the data in "DataForPlot" change as expected. "DataForPlot" is pulling data from a specific "z_" data sheet using VBA generated cell formulae.

3.5. Underlying Logic of the Code You Just Ran

Let us formally state the problem in the context of dynamic surface charts: **We want to build a tool that consolidates multiple (3-D) surface charts into a single visualization tool.** In order to attain such a goal, we can divide the problem into more bite-sized chunks:

1. We will require worksheets containing data that is formatted for outputting a surface chart.
2. Since we will display multiple 3-D surface charts, we will need to know how to output at least one surface chart programmatically through VBA.
3. People can only see one surface chart at a time. Therefore user interaction is necessary to indicate their selection of a particular surface chart out of many. It would be useful to consider scrollbar binding to a changing cell value of a static worksheet containing data.

We will henceforth call this virtual data worksheet "DataForPlot", and the worksheet that displays the dynamic surface chart "3DPlot". The corresponding subroutine calls are "Setup_MainSheet" and "Setup_PlotSheet". The vision is to use the scrollbar to scroll through sheets of data as well as their corresponding surface charts.

The following sections will review the mechanics behind each major subroutine and function. We then review the test subroutine that calls these subroutines and functions in a procedural manner.

3.6. Preparing the Dynamic Range: The "Setup_MainSheet" Subroutine

Viewing the data using our dynamic surface chart to view individual 3-D slices of 4-D animation is like using a microfilm machine to view individual slides of archived newspaper articles. The machine is composed of a magnifying lens and a slide sorter consisting of multiple slides. The "Setup_Mainsheet" subroutine is responsible for loading a particular slide's data to be magnified and viewed. The worksheet responsible for preparing this view at the particular 3-D data is called "DataForPlot". As with the microfilm machine, the user manually scrolls to a certain slide, which is properly indexed. Similarly, our data is indexed in numerical order from -40 to 40 Å (for the biology example). "DataForPlot" also reports the index number of the slide for viewing. Since the scrollbar can only take numbers 1, 2, 3 and so on, the index number of the 3-D data needs to be translated into the scrollbar value via a simple mathematical formula. Depending on the scrollbar value, and therefore the index number of the 3-D data to "view",

the dynamic range reports the corresponding data values of the particular 3-D slice of the 4-D animation.

3.7. Displaying the Surface Chart Animation: The "Setup_PlotSheet" Subroutine

While the "DataForPlot" sheet is like the prepared slide for microfilm viewing, the surface chart animation is the viewer itself. The "Setup_PlotSheet" subroutine prepares both the surface chart visualization tool and the scrollbar. The visualization tool is the interface between the user and the data. To control the index number of the 3-D slice of the 4-D animation, the subroutine generates a scrollbar, whose value dictates which particular 3-D data to view. When the scrollbar value is changed, the value of index reported in "DataForPlot" updates accordingly, and vice versa.

3.8. Giving the Surface Chart Analytical Meaning: the "ColorGrad" Function

When a surface chart is created, Excel does not create meaningful legend colors. They start off as random colors. There exist useful websites demonstrating how to implement gradient-based color schemes [34] for heat maps (contour plots) [35, 36] and therefore surface charts by extension. "ColorGrad" paints the surface chart with a meaningful color gradient, so that visualization and analysis will be easier for those reading the surface chart.

3.9. Putting It All Together with the "test_S3DP" Subroutine

The test subroutine, here called "test_S3DP" (short for testing surface 3-D plot), is a subroutine without arguments used to call the two main subroutines: one that generates a sheet "Setup_MainSheet" with virtual data, and another "Setup_PlotSheet" that generates the surface plot. It is necessary to write at least one subroutine without arguments because subroutines with arguments cannot be directly run unless the arguments are indicated in a call. Therefore this type of subroutine call needs to be performed within a testing subroutine. The next page shows a schematic of the interplay between all the sheets created through the VBA code we have discussed (Fig. 7).

3.10. Optional: Replace Current Scrollbar with ActiveX Scrollbar for Fluid Animation

The animation of the dynamic surface chart could be continuous and can be viewed by holding down the arrow key on the scrollbar. However, this requires some manual replacement of the forms version of the scrollbar (the one implemented here) with an ActiveX version of the scrollbar. The following steps show the reader how to manually install the ActiveX scrollbar:

Mac users cannot proceed with this tutorial as Mac Offices are not ActiveX compatible.

1. Delete the scrollbar on the sheet "3DPlot" by right-clicking it once and pressing Cut.

2. In Developers tab, go to Insert → ActiveX Controls → Scroll Bar
3. Activate "Design Mode" in the Developers tab
4. Draw out a scrollbar underneath the surface chart
5. Right-click the scrollbar while in Design Mode → Properties
6. Change the field "Min" from 0 to 1
7. Change the field "Max" from 32767 to 81
8. Exit the Properties dialog
9. Right-click the scrollbar → View Code
10. Insert the following code:

```
Private Sub ScrollBar1_Change()
        Sheets("DataForPlot").Cells(2, 1) = ScrollBar1.Value
End Sub
Private Sub ScrollBar1_Scroll()
        Sheets("DataForPlot").Cells(2, 1) = ScrollBar1.Value
End Sub
```

11. Go back to "3DPlot" and exit the Design Mode.
12. You can now hold the right arrow on the scrollbar and watch the movie-like animation.

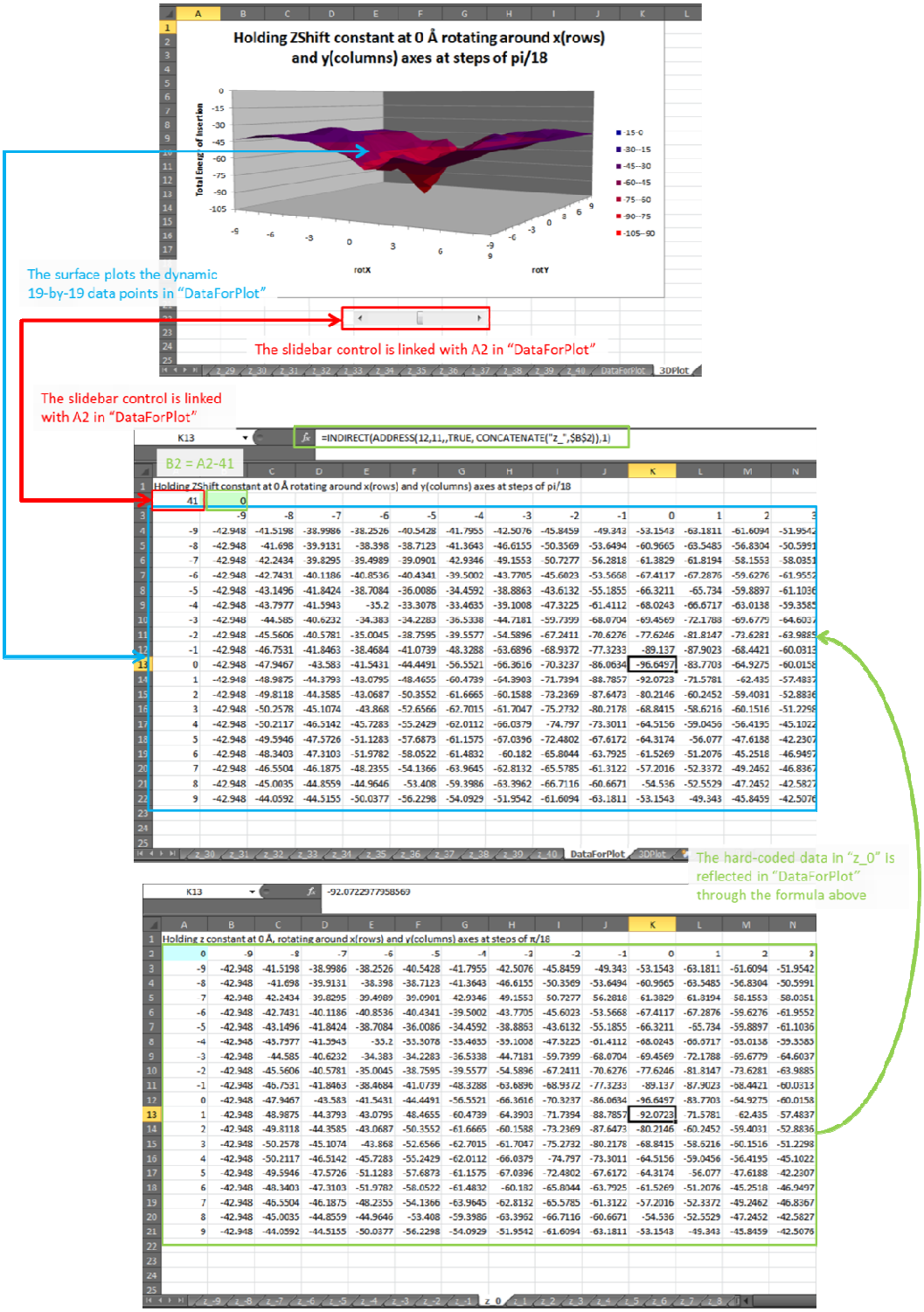

Figure 7: A schematic of sheets "z_-40" to "z_40",

4. Discussion: What our Visualization Tells Us

By linking our scrollbar values 1 to 81 with the biologically relevant energies at depths from the center of the bilayer, z = -40 to 40 Å, we can now visualize the landscape of all possible orientations of a single protein. We have prepared the surface chart animation along with the corresponding biophysical animation as one animated GIF file. Please download "insertionAnimation.gif" and view it in PowerPoint or your favorite animated GIF viewer program.

By using VBA to automate this visualization process for each protein, we found a pattern that verifies our potential: that every protein reaches an energy minimizing orientation when situated in the center of the membrane. Since we have surveyed all orientations, we see that this corresponds to an energy funnel (think lowest ground in a valley), where amino acids have reached their optimal placement of the entire corresponds with a stable orientation. This tool could be applied further to delve into the preference of insertion pathway for membrane proteins, i.e. from outside the cell (extracellular space) towards intracellular space versus the scenario vice versa.

5. Conclusion

We show that through programming a surface chart whose data points are indirectly bound to the name of sheets of interest that dynamic surface charts can be created in Excel. Although we developed a solution in the context of evaluating protein depth-dependent energy with respect to the center of the membrane and orientation in 3-D space, a dynamic surface chart could benefit all kinds of fields that need to visualize multidimensional data, including finance, economics, theoretical and applied sciences, etc.

6. Appendix: Energy Landscape Calculations for Each Depth z = -40 to 40 Å

Any membrane protein sequence is composed of tens to hundreds of amino acids. Each amino acid in these sequence is one of twenty amino acid types, each with their own energy function associated [27, 28]. The associated energy of insertion for each amino acid within the membrane protein sequence is contained in a worksheet called "Hybrid". (Fig. 8) This worksheet is then subjected to rigid motion transformations and other downstream calculations (VBA code not included). Since only the depth and amino acid type are the only relevant parameters in this model of insertion, the rigid motion transformations are applied only to the z-values (the depths) of the amino acids. The sum of the resulting energies are changed as well, and then summed. Each surface chart value in the sheets "z_-40", "z_-39" ... "z_39", and "z_40" is the result of an orientation and then vertical translation of the membrane protein. This is a far more complex calculation than the potential energy setup described in Section 2.1 and requires the 81 sheets as intermediate data output for the visualization.

Column D:
Amino acid type

Column H:
Depth (z-coordinate) of each amino acid

L12

	A	B	C	D	E	F	G	H	I	J
1	pdbID	Chain	ResIndex	3-Code	1-Code	Secondar	%SASA	zCoord of	GLY=1	vectProj
2	3EMN	X	1	MET	M		0.970588	-17.587	0	-0.17075
3	3EMN	X	2	ALA	A		0.245283	-16.826	0	1.082164
4	3EMN	X	3	VAL	V	S	0.256098	-13.455	0	-1.52529
5	3EMN	X	4	PRO	P		0.043716	-8.835	0	1.288047
6	3EMN	X	5	PRO	P		0.120219	-6.672	0	-1.44983
7	3EMN	X	6	THR	T		0.290323	-3.819	0	-1.48287
8	3EMN	X	7	TYR	Y	G	0.08	0.048	0	1.140824
9	3EMN	X	8	ALA	A	G	0.358491	0.836	0	-1.50511
10	3EMN	X	9	ASP	D	G	0.380952	-3.937	0	-0.79848
11	3EMN	X	10	LEU	L	T	0.04717	-1.83	0	1.417988
12	3EMN	X	11	GLY	G	T	0.057692	-1.76	1	-0.66159
13	3EMN	X	12	LYS	K	H	0.376623	-3.223	0	-1.35309
14	3EMN	X	13	SER	S	H	0.198864	-6.071	0	0.935633
15	3EMN	X	14	ALA	A	H	0.006289	-1.765	0	1.407821
16	3EMN	X	15	ARG	R	H	0.348921	0.857	0	-1.45752
17	3EMN	X	16	ASP	D	H	0.190476	-3.331	0	-0.61337
18	3EMN	X	17	VAL	V	H	0.02439	-2.625	0	1.495627
19	3EMN	X	18	PHE	F	H	0.139241	2.668	0	-1.17116
20	3EMN	X	19	THR	T	H	0.456989	2.339	0	-1.55538
21	3EMN	X	20	LYS	K	T	0.4329	-2.379	0	-0.42844
22	3EMN	X	21	GLY	G	T	0.182692	-1.028	1	1.528295
23	3EMN	X	22	TYR	Y		0.084	3.476	0	-1.33149
24	3EMN	X	23	GLY	G		0.105769	5.304	1	1.370345
25	3EMN	X	24	PHE	F	T	0.168776	9.157	0	-1.36462
26	3EMN	X	25	GLY	G	T	0.009615	11.236	1	-1.51587
27	3EMN	X	26	LEU	L	E	0.25	8.366	0	-1.39155
28	3EMN	X	27	ILE	I	E	0.405263	7.282	0	1.4978
29	3EMN	X	28	LYS	K	E	0.229437	3.734	0	-1.42626
30	3EMN	X	29	LEU	L	E	0.509434	2.334	0	1.423139

EzBetaParamArray | PreHybrid | Hybrid | z_-40 | z_-39 | z_-38 | z_-37 | z_-36

depth of each
o rigid motion
at associate each
demonstrate the
e workbooks.

7. References

[1] Clark, A.C. and Matthews, B. (2000). "Scientific and Technical Visualization: A New Course Offering that Integrates Mathematics, Science, and Technology". *J. Geom. and Graphics, 4* (1), 89-98.
Available at: http://www.heldermann-verlag.de/jgg/jgg01_05/jgg0407.pdf

[2] Arganbright, D. (2005). "Enhancing Mathematical Graphical Displays in Excel through Animation". *Spreadsheets in Education (eJSiE), 2* (1), Article 8.
Available at: http://epublications.bond.edu.au/ejsie/vol2/iss1/8

[3] Wischniewsky, Wilfried A.L. (2008). "Movie-like Animation with Excel's Single Step Iteration Exemplified by Lissajous Figures". *Spreadsheets in Education (eJSiE), 3* (1), Article 4.
Available at: http://epublications.bond.edu.au/ejsie/vol3/iss1/4

[4] Benacka, J. (2011). "Graphing Functions of Two Variables in Spreadsheets". *Spreadsheets in Education (eJSiE), 4* (3), Article 3.
Available at: http://epublications.bond.edu.au/ejsie/vol4/iss3/3

[5] Vieira, I. (2008, December 2). "Support for 3D Series". [Forum discussion].
Available at:
http://social.msdn.microsoft.com/Forums/en/MSWinWebChart/thread/67c586ae-26c7-4510-be22-7eb0c181d181

[6] Frith, S. (2005, May 12). "Excel Should Show 4 Dimensional Charts". [Forum discussion].
Available at: http://www.excelbanter.com/showthread.php?t=25824

[7] Nelson, S. (2009, September 24). "Is Excel Already a Multidimensional Spreadsheet?" [Blog post].
Available at: http://it.toolbox.com/blogs/tricks-of-the-trade/is-excel-already-a-multidimensional-spreadsheet-34345

[8] Peltier, J. (2010, February 23). "Contour and Surface Charts in Excel 2007". [Blog post].
Available at: http://peltiertech.com/WordPress/contour-and-surface-charts-in-excel-2007/

[9] Ferry, D. (2010, May 26). "Animated Business Chart". [Blog post].
Available at: http://www.excelhero.com/blog/2010/05/animated-business-chart.html

[10] Mehta, T. (n.d.) "3D Surface".
Available at: http://www.tushar-mehta.com/excel/charts/3d_surface/

[11] Pope, A.J. (n.d.) "3D XY Scatter Chart".
Available at: http://www.andypope.info/charts/3drotate.htm

[12] Doka, G. (n.d.) "3D Scatter Plot For MS Excel (VBA Macro)".
Available at: http://www.doka.ch/Excel3Dscatterplot.htm

[13] Microsoft Office. (n.d.) "Overview of Online Analytical Processing (OLAP)". Available at: http://office.microsoft.com/en-us/excel-help/overview-of-online-analytical-processing-olap-HP010177437.aspx

[14] Hoadley, P. (n.d.) "Implied Volatility Calculator (Including Hedging Optimizer)". Available at: http://www.hoadley.net/options/develtoolsimpliedvolcalc.htm

[15] Kimmel, P.T., Green J., Bullen, S., Bovey, R., Rosenberg, R., Patterson, Brian D., (2004), *Excel 2003 VBA: Programmer's Reference*, Wiley Publishing Inc.

[16] Microsoft DreamSpark. (n.d.) Available at: https://www.dreamspark.com/

[17] Microsoft Developer Network. (n.d.) ".NET Development". Available at: http://msdn.microsoft.com/library/aa139615

[18] Microsoft Developer Network (n.d.) "Learning Visual Studio". Available at: http://msdn.microsoft.com/en-us/vstudio/cc136611

[19] Meister, C. (2008, May 25). "Please Read First – What is VSTO and Non-VSTO Resources". Available at: http://social.msdn.microsoft.com/Forums/en-US/vsto/thread/063a23a6-1595-4c83-a25f-6c94658c4649/

[20] VSTOTeam. (2008, January 16). "The Reports of VBA's Demise Have Been Greatly Exaggerated". Available at: http://blogs.msdn.com/b/vsto/archive/2008/01/16/the-reports-of-vba-s-demise-have-been-greatly-exaggerated.aspx

[21] Mueller, J. (2012, July 26). "John's Random Thoughts and Discussions: VBA and Office 2013". Available at: http://blog.johnmuellerbooks.com/2012/07/26/vba-and-office-2013.aspx

[22] Microsoft Developer Network (n.d.) "Chart Controls". Available at: http://msdn.microsoft.com/en-us/library/dd456632.aspx

[23] Burns, D. (n.d.). "Three-Dimensional (Surface) Plots". Available at: http://faculty.pingry.k12.nj.us/dburns/AdvPhys/documents/3DPlotting-ScrollbarsHandout.pdf

[24] Walther, D.M., Rapaport, D., Tommassen, J. (2009 September). Biogenesis of beta-barrel membrane proteins in bacteria and eukaryotes: evolutionary conservation and divergence. *Cell Mol. Life Sci., 66* (17):2789-804.

[25] PubMed Health (n.d.) "MRSA – PubMed Health". Available at: http://www.ncbi.nlm.nih.gov/pubmedhealth/PMH0004520/

[26] Caruana, D.J. and Howorka, S. (2010). Biosensors and Biofuel Cells with Engineered Proteins. *Mol. BioSyst*., *6*, 1548-56.

[27] Senes, A., Chadi, D.C., Law, P.B., Walters, R.F.S, Nanda, V., DeGrado, W.F. (2007). E_z, a Depth-Dependent Potential for Assessing the Energies of Insertion of Amino Acid Side-chains into Membranes: Derivation and Applications to

Determining the Orientation of Transmembrane and Interfacial Helices. *J. Mol. Biol., 366*. 436-48.

[28] Hsieh, D., Davis, A., Nanda, V. (2012). A Knowledge-based Potential Highlights Unique Features of Membrane α-helical and β-barrel Protein Insertion and Folding. *Protein Science, 21*. 50-62.

[29] Walkenbach, J., (2004), *Excel 2003 Power Programming with VBA*, Wiley Publishing Inc.

[30] Walkenbach, J., (2010), *Excel 2010 Power Programming with VBA*, Wiley Publishing Inc.

[31] Microsoft Office (n.d.) "Change Macro Settings in Excel". Available at: http://office.microsoft.com/en-us/excel-help/change-macro-security-settings-in-excel-HP010096919.aspx

[32] Microsoft Office (n.d.) "Macro Security Levels in Office 2003". Available at: http://office.microsoft.com/en-us/office-2003-resource-kit/macro-security-levels-in-office-2003-HA001140307.aspx

[33] Richardson, M.A. (2007, October 30). "Export Your VBA Code for Use in Another Excel Application". Available at: http://www.techrepublic.com/blog/msoffice/export-your-vba-code-for-use-in-another-excel-application/295

[34] Pope, A. (2007, April 28). "XY Scatter Colouration Plot". Available at: http://www.andypope.info/charts/spectrum.htm

[35] Blumenstock, J. (2003, April 24). "Heatmap Tool (VBA)". Available at: http://senorjosh.jblumenstock.com/archives/2003/04/heatmap_tool_vba.shtml

[36] The "How" Blog. (2009, April). "How To Create a Heat Map in Excel". Available at: http://how.best-free-information.com/2009/04/how-to-create-a-heat-map-in-excel/