SAND REPORT

SAND2002-1881
Unlimited Release
Printed June 2002

A Pattern Recognition Feature Optimization Tool Using the Visual Empirical Region of Influence Algorithm

Rubel F. Martinez

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/ordering.htm

SAND2002-1881
Unlimited Release
Printed June 2002

A Pattern Recognition Feature Optimization Tool Using the Visual Empirical Region of Influence Algorithm

Rubel F. Martinez
Laser Optics and Remote Sensing Dept.
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-1423

Abstract

This document is the second in a series that describe graphical user interface tools developed to control the Visual Empirical Region of Influence (VERI) algorithm. In this paper we describe a user interface designed to optimize the VERI algorithm results. The optimization mode uses a brute force method of searching through the combinations of features in a data set for features that produce the best pattern recognition results. With a small number of features in a data set an exact solution can be determined. However, the number of possible combinations increases exponentially with the number of features and an alternate means of finding a solution must be found. We developed and implemented a technique for finding solutions in data sets with both small and large numbers of features.

This document illustrates step-by-step examples of how to use the interface and how to interpret the results. It is written in two parts, part I deals with using the interface to find the best combination from all possible sets of features, part II describes how to use the tool to find a good solution in data sets with a large number of features.

The VERI Optimization Interface Tool was written using the Tcl/Tk Graphical User Interface (GUI) programming language, version 8.1. Although the Tcl/Tk packages are designed to run on multiple computer platforms, we have concentrated our efforts to develop a user interface for the ubiquitous DOS environment. The VERI algorithms are compiled, executable programs.

The optimization interface executes theVERI algorithm in Leave-One-Out mode using the Euclidean metric. For a thorough description of the type of data analysis we perform, and for a general Pattern Recognition tutorial, refer to our website at: http://www.sandia.gov/imrl/XVisionScience/Xusers.htm.

Contents

Figures

Part I.
Exact Optimization Solution

This part of the paper discusses and exemplifies the section of the interface designed to optimize the feature set in multi-feature data sets. In this part of the paper we discuss using the interface to find the combination of features that produces the best Pattern Recognition (PR) results from all possible combinations.

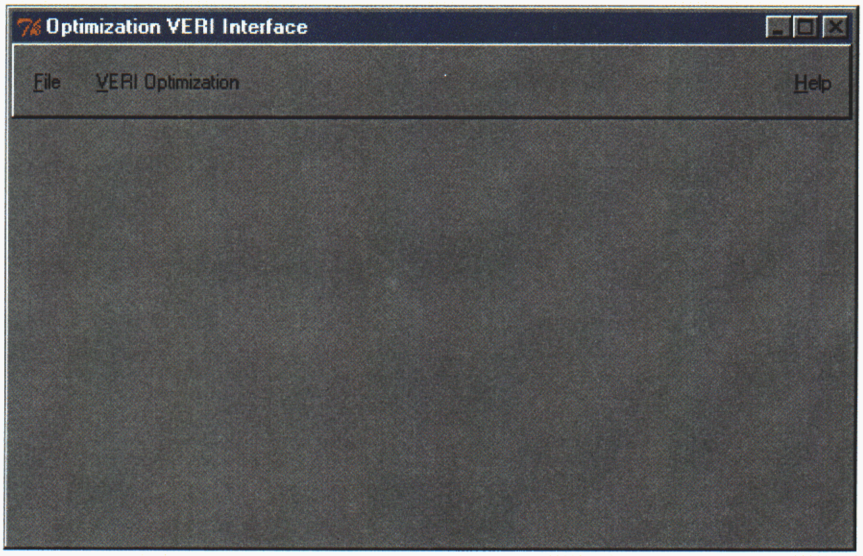


Figure 1. Top Level Window of Optimization Interface

Figure 1 shows the window that is displayed each time theVERI Optimization Interface Tool is executed. There are two pull-down menus on the menu bar, the "File" menu, and the "VERI Optimization" menu. The "File" menu is used to locate data files and to designate locations in storage where results will be stored. The "VERI Optimization" menu is used to set parameters for the optimization analysis. A "Help" menu also exists. In the Help menu, users are directed to read documents supplied with the interface.

The VERI Optimization Interface Tool is written with the Tcl/Tk[1] Graphical User Interface (GUI) software packages. This is the same GUI package that controls our single pass Leave-One-Out data analysis and visualization tool[2]. The two interfaces have the same look and feel, thus making it intuitive to use both interfaces. A key difference in the two interfaces lies in the method of selecting features for which PR results are generated. In the single pass version, the user manually selects a set of features, and PR results are generated for the selected feature set, whereas in the optimization version, results using all feasible combinations of the features are examined and the combinations that produce the best PR results are reported. The single pass version has a tool for data visualization. This tool aids the user to visually examine characteristics (such as shape and volume of the classes of data) in the data examined by the Optimization algorithm.

There is no charge to acquire the VERI Optimization interface. For information on how to obtain this package, contact the author at rfmarti@sandia.gov.

11.4296875	**0.005347252**	**1.80006E-05**	**8.234523**	**0.0799179**	**1**
10.4375	**0.019611359**	**0.001790166**	**18.89843**	**0.0670433**	**1**
9.3359375	**0.029211998**	**0.001452088**	**11.9687**	**0.0100764**	**1**
9.0703125	**0.0508461**	**0.00472486**	**8.398430**	**0.0768080**	**1**
5.23046875	**0.01540947**	**0.000308871**	**8.460468**	**0.0184094**	**2**
13.6796875	**0.035961151**	**0.002737999**	**12.8675**	**0.0136113**	**2**
4.3984375	**0.036808014**	**0.003687263**	**14.12968**	**0.0634725**	**2**
18.8984375	**0.000670433**	**6.69956E-05**	**4.898434**	**0.0670433**	**3**
18.4609375	**0.000799179**	**0.000872612**	**8.660156**	**0.0134520**	**3**
17.96875	**0.001009941**	**0.000922203**	**9.41625**	**0.0185271**	**3**
17.8671875	**0.000728607**	**0.001067042**	**12.93968**	**0.0318618**	**3**
-5.66015625	**-0.013422012**	**-0.000935555**	**18.46093**	**0.0799172**	**5**
-5.33984375	**-0.018762589**	**-0.001707077**	**17.7177**	**0.0728607**	**5**
-5.390625	**-0.018527031**	**-0.00110054**	**6.2103835**	**0.0804411**	**5**

Figure 2. Sample contents of VERI spread sheet file named "voc2.ss"

Data Format

The VERI Optimization Interface Tool reads the data it will analyze from a computer file. The user must manipulate their data to create files with a specific format. The interface is designed to read data from files saved in a spreadsheet format (hence the .ss in the file names). Figure 2 shows an example of what the contents of a properly formatted spreadsheet file named voc2.ss may look like. Spreadsheet format requires that the data on each row be separated by white space such as character spaces or tabs, and ended with a carriage return. Each row of data in a spreadsheet file is a spectrum of N measures of data acquired in a test setting, followed by an integer class value for that data. Each row contains all N dimensions (features) of one data point. In Figure 2, there are fourteen data points, each point has five dimensions. There are four classes of data (1,2,3,5). All the data in a class must be stored contiguously.

A class of data refers to all measurements acquired when the same types of inputs are being measured by the system. For example, to measure gas compounds, a test bed of with N sensors may be developed whose signal is proportional to the amount and type of gas accumulated by the sensors over a fixed period of time. Each time a set of N measurements is made at the end of the accumulation period, the sensors must be flushed and prepared for the next set of measurements. In this example, all data measured (with varying concentrations) when the sensors accumulate acetone vapors are in one class, all data measured when the sensors accumulate methanol vapors are in another class. Each row of data must have the same number (N) of measured values or features.

The number of features and number of points per class in figure 2 are only for illustrating proper file formatting. In most cases this would be considered a sparse data set. Not enough data has been measured (too few concentrations) to properly represent each class of data.

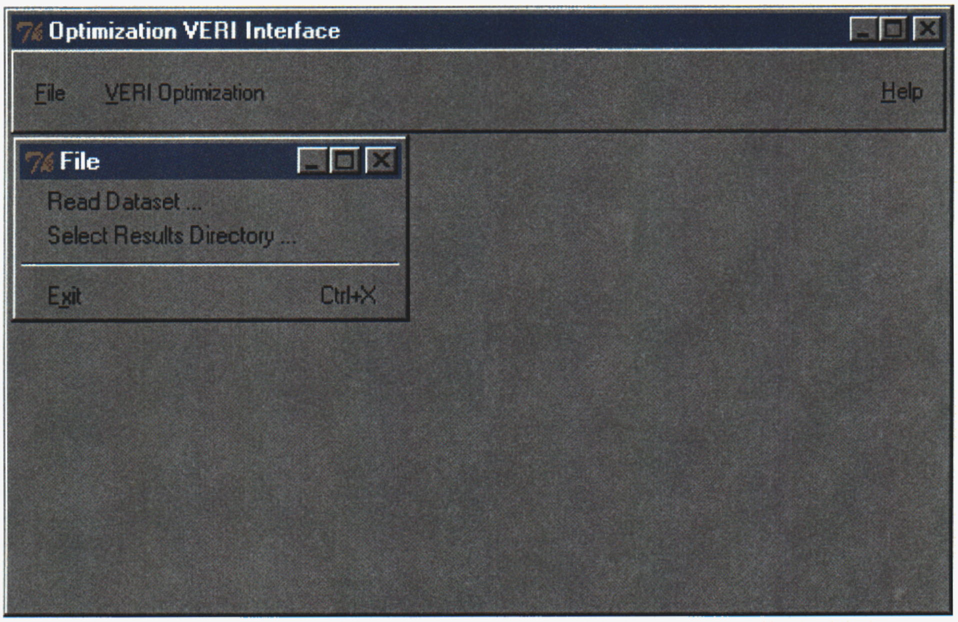


Figure 3. "File" Menu Contents

"File" Menu

Figure 3 shows the commands available in the "File" menu. The main functions of this menu are to select which data to analyze and where to store the analysis results. The "File" window can be opened and posted. Posting a window causes it to remain open until closed by the user. The "File" window is posted by selecting the menu name "File" in the main menu bar with the left mouse button and releasing the button when the dashed line at the top of the File pull down window is highlighted. The dashed line is not illustrated in figure 3. Clicking on the standard Windows close icon (X) in the upper right corner of the window will close the posted "File" menu.

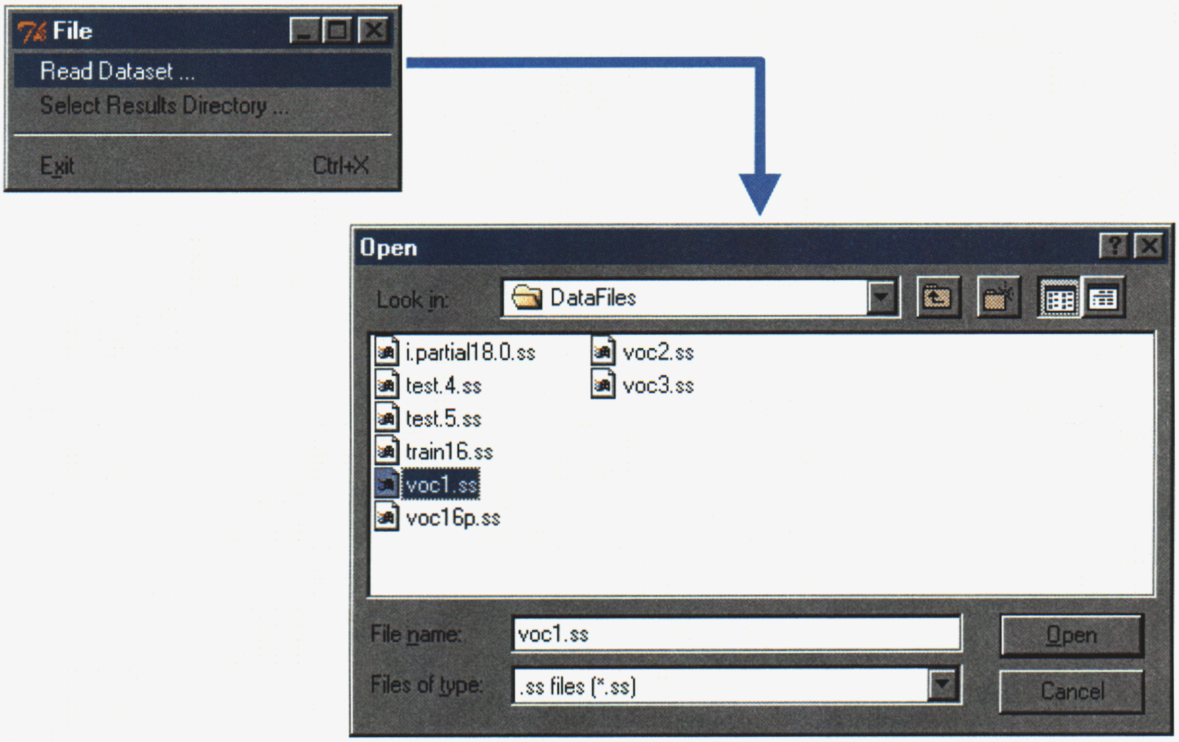


Figure 4. Reading data with the "Read Dataset" sub menu in "File" menu.

The "File" menu is where the interface accesses the spreadsheet data files. Figure 4 shows an example of opening the "File" menu and selecting the "Read Dataset" sub-menu. This figure depicts a user browsing a computer hard disk to find a file to analyze. In this illustration the file voc1.ss has been selected. The Tcl/Tk packages interface with built-in Windows functions and dialog boxes to read in file names. When "Read Dataset" is selected, a Windows "Open" dialog box is used to interface with the data files saved on the computer. Double clicking on a file name or single clicking and pressing the "Open" button will read in a file name and close the dialog box. The interface defaults to list names of files of type ".ss files (*.ss)". These are user formatted spreadsheet files whose file names end with the characters ".ss".

Figure 5. Selecting a directory to store optimization results in.

The Optimization algorithm generates result files for each subset of dimensions it analyzes. A location for the result files must be selected or created on a system hard disk. Figure 5 shows the process of selecting a directory in which to store optimization results. When the user selects the "Select Results Directory" sub menu from the "File" menu, a Windows "Save As" dialog window opens allowing the user to select from an existing directory or to create a new directory. In this example a directory named "voc1" has been selected. A concatenated list of the names of the directories leading to where the optimization results are stored is known as a directory path. For example, H:\docs\optim\results\voc1 is a directory path. **Directory paths where input data files are stored or where results files will be stored must not contain white space. The optimization algorithms will not operate correctly if spaces or tabs are included in the directory paths.**

A directory named "veri-matrix" is created under the parent directory selected by the user. Figure 5 shows the name "veri-matrix" in the "File name" entry line. This name cannot be changed.

The final selection available in the "File" menu is the "Exit" command. Selecting the "Exit" sub-menu will exit and quit the Optimization interface. It can be selected at any time when then interface is running.

VERI Command and Control

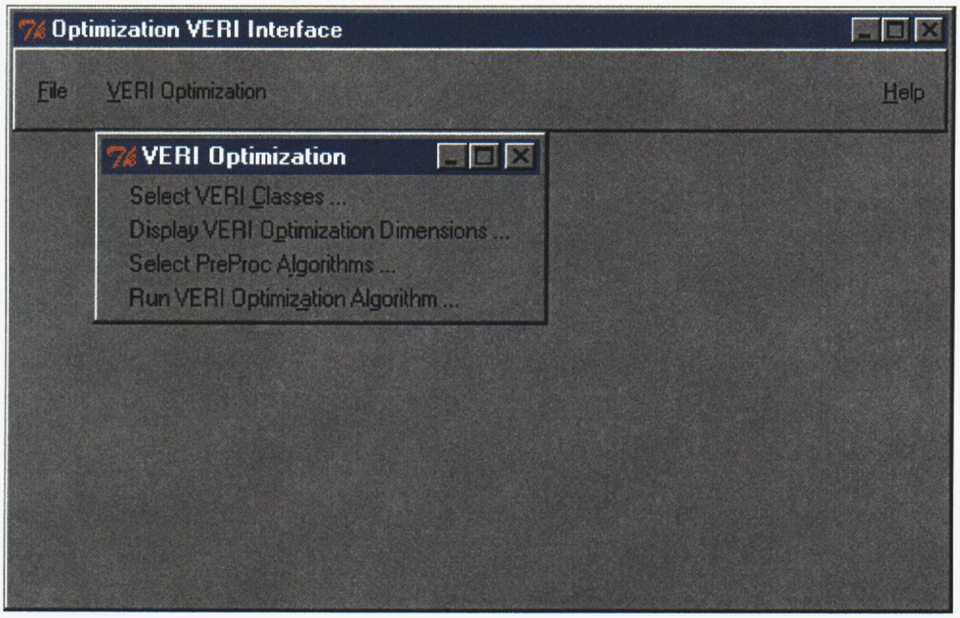


Figure 6. VERI Optimization menu – Select Parameters and Optimize

"VERI Optimization" Menu

Figure 6 shows the options available in the "VERI Optimization" pull down menu. The "VERI Optimization" window can be opened and posted in a manner similar to the way the "File" menu is posted. The "VERI Optimization" menu interfaces to sub-windows where the user can select the remaining parameters for the optimization program command line. There is also a sub-window to run the optimization algorithm with the supplied parameters. Each sub-window and the effects it has on the optimization results will be described next.

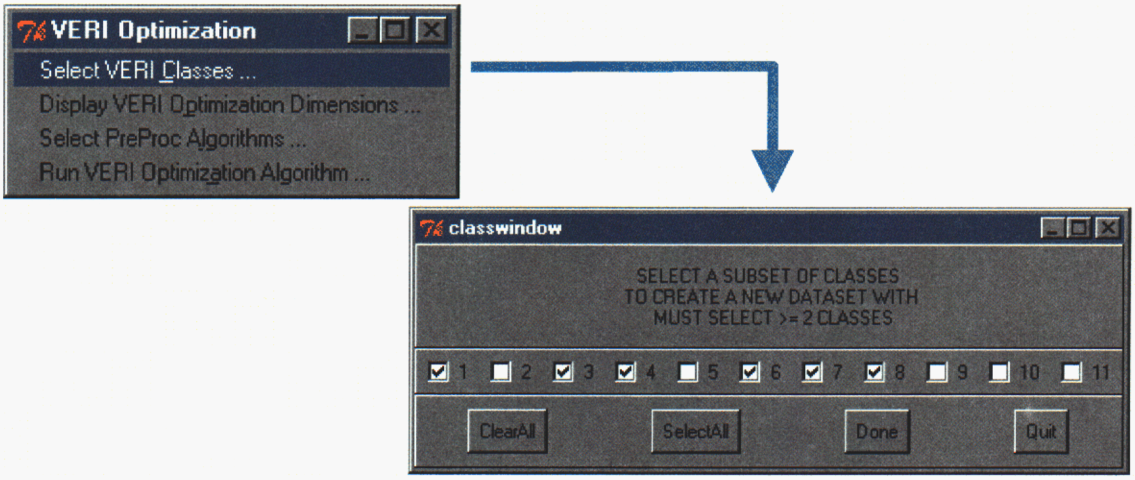


Figure 7. VERI Optimization Classwindow -- Select Classes of data to optimize

The first parameter on the "VERI Optimization" menu allows the user to select which classes of data to include in their analysis. All the classes of data stored in a file do not need to be analyzed together. It is possible to select only the classes of interest while containing all classes in one file. At a minimum, two classes must be selected. Figure 7 illustrates selecting classes of data from a file with eleven classes. In this data set the classes are numbered 1-11 sequentially, and classes 1,3,4,6,7,8 have been selected. It is not necessary to number classes sequentially in an input data file, but classes must be numbered with values greater than or equal to zero.

The classes can be individually selected or deselected for use in the optimization analysis. This is accomplished by pointing the cursor in the box to the left of a class number and clicking on the left mouse button. Buttons to affect all the classes in the class list are also available. The operations performed by the buttons are self explanatory, but will be explained for the sake of thoroughness:

ClearAll – clears all currently selected classes
SelectAll – selects all available classes
Done – Saves selected class values and exits Classwindow
Quit – Does not save class values and exits Classwindow

After the classes of interest have been selected, press "Done" to exit this sub-window.

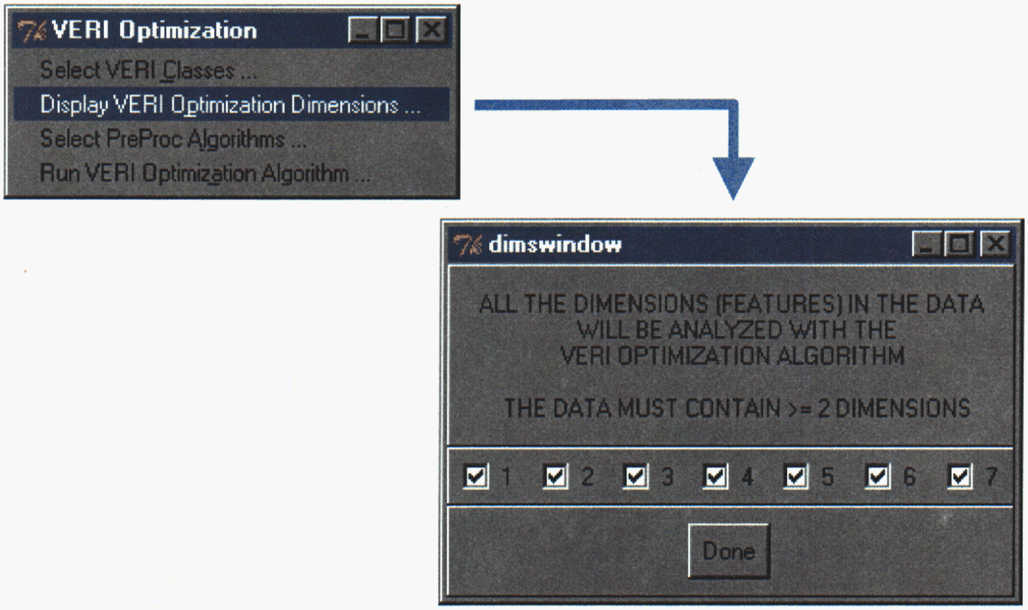


Figure 8. VERI Optimization Dimswindow – display dimensions in data set.

Unlike the VERI single pass tool where the user manually selects which dimensions to analyze, the VERI Optimization Interface Tool uses all of the dimensions of the data in its analysis. Using all dimensions is a fundamental aspect of the optimization algorithm since it searches for the combination of dimensions that gives the best PR results. Figure 8 shows the window displayed for a data set that has seven dimensions of data per measurement cycle (N=7). The dimensions are always numbered 1…N. Data must contain at least two dimensions, and all classes must have the same number of dimensions.

While the Optimization algorithm attempts to use all dimensions to generate its results, it sometimes is not feasible to use all dimensions as the number of dimensions increases. For example, to compute results for a data set with six dimensions requires 57 combinations of the feature set, a data set with ten dimensions requires 1,013 combinations, and a data set with twenty dimensions would require that a total of 1,048,555 combinations be examined. The first two sums of combinations are easily achievable, but the latter will require substantial computing power to complete the project in a reasonable amount of time. We have developed a technique to handle computing optimization results for data sets with a large number of dimensions, and we will discuss it in the next section.

Figure 9. VERI Optimization Preprocwindow – processes to spatially separate data.

We have included data processing algorithms as part of the interface to try and improve data spatial separation and therefore the optimization results. The best results are achieved when classes of data are spatially well separated from each other and when the points in a class uniformly fill a volume. These algorithms attempt to accomplish these goals, but their use is optional. Figure 9 shows the algorithms available under the window named "Preprocwindow". These algorithms have proven useful in spatially separating classes of data. Two algorithms are available, but only one can be applied per run of the Optimization algorithm. It is best to think of the data points as multi-dimensional vectors with a magnitude and direction when using and describing these functions.

The Normalize function scales the magnitude of the data points in the direction of a unit sphere. As examples, a 25% normalize function scales each vector to a magnitude of 25% between the original magnitude and a magnitude of 1.0, a 50% normalize function scales each vector to a magnitude of 50% between the original magnitude and a magnitude of 1.0, and a 100% normalize function scales each vector to a magnitude of unit length 1.0.

The Radialize function attempts to introduce inter-class data point separation by displacing the points radially away from the origin. After the Radialize function is applied, the original direction of each vector is maintained, but their magnitude is increased. The data points are further away from the origin after the Radialize function is applied. If you have points in several classes that spatially occur close to the origin, the Radialize function can serve to separate the points and classes.

The N dimensions of data that are selected for analysis are always equalized. Data that have been equalized occur in the range $-1.0 <= Eq <= 1.0$. Equalization eliminates sensor signal scaling and helps to spatially distribute the data in a uniform volume where the maximum distance from the origin to a data point is $\sqrt{N}$.

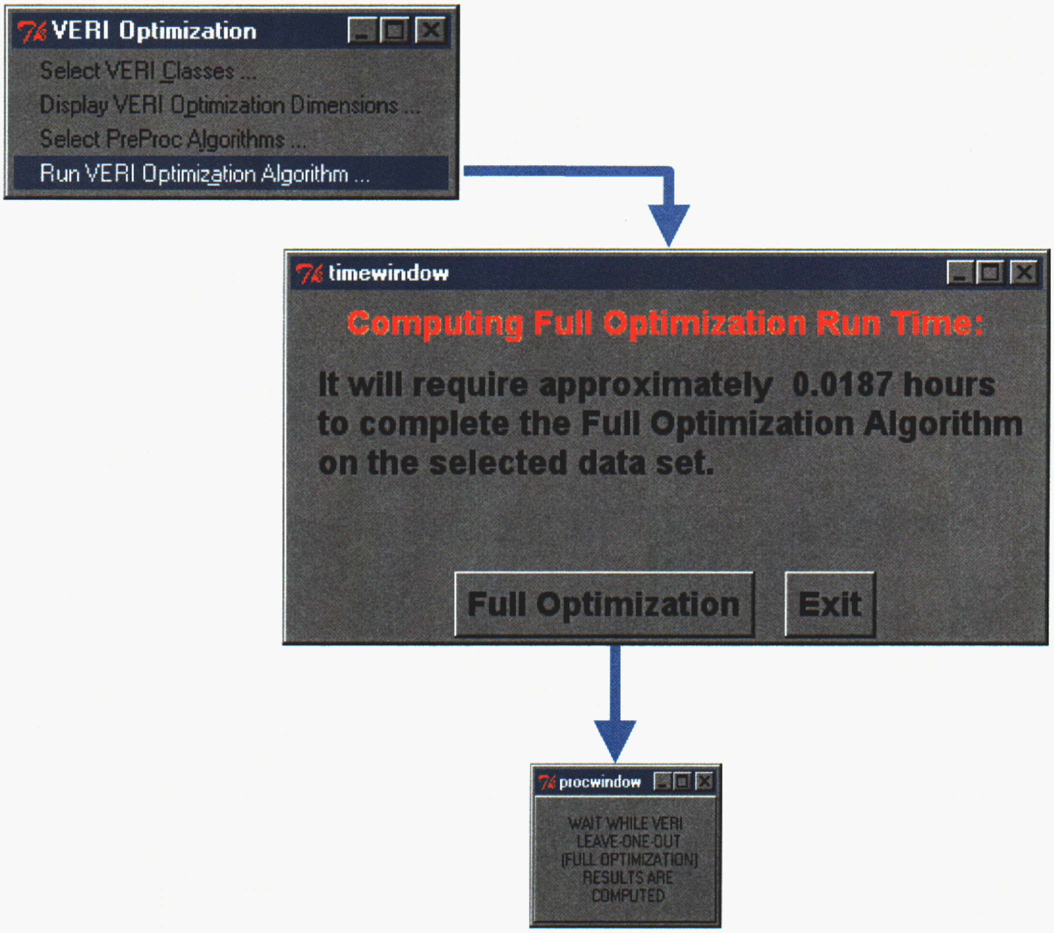
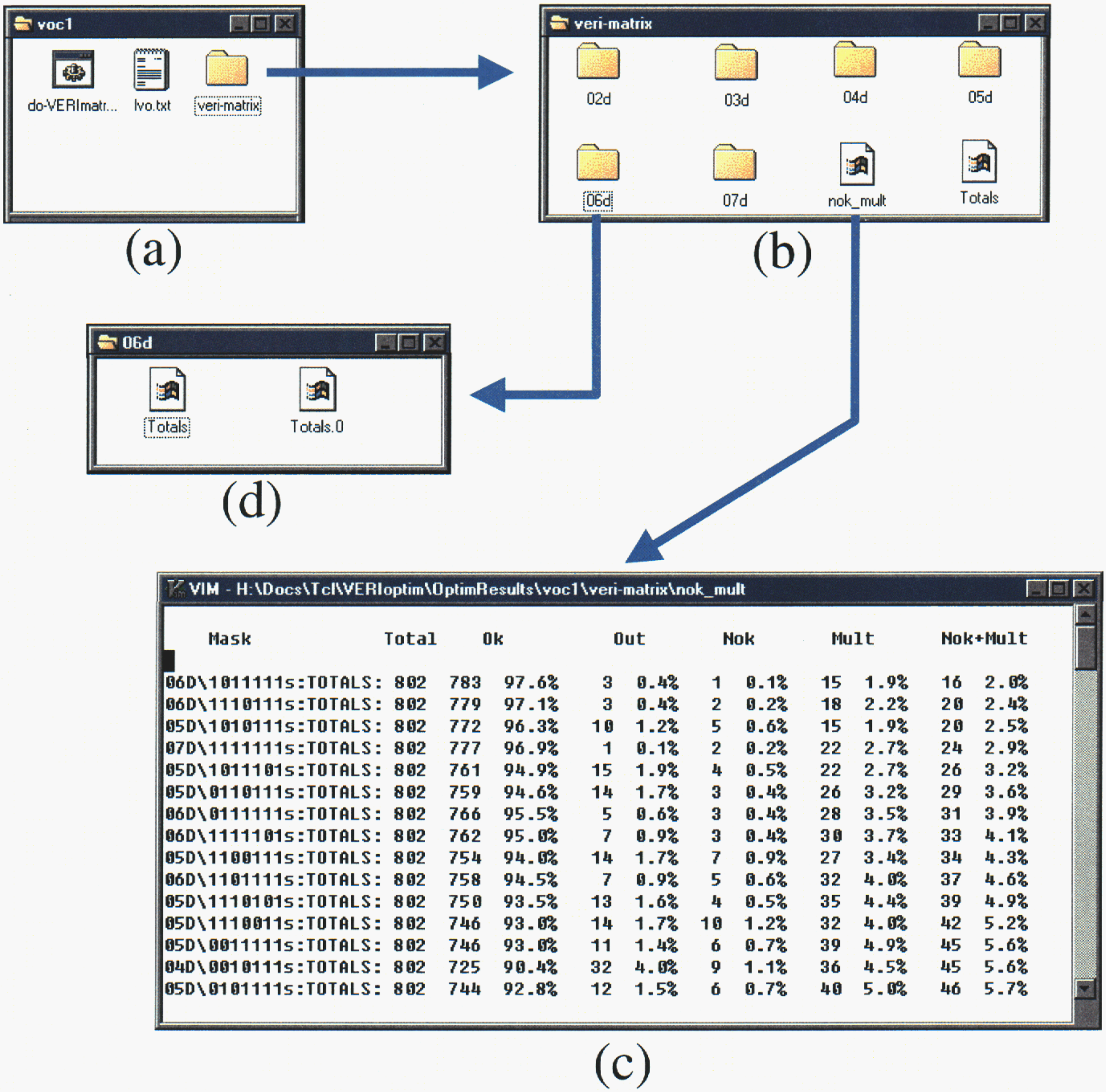


Figure 10. VERI Optimization Timewindow -- Computing Full Optimization

The user can run the optimization program when all the necessary parameters have been selected. Figure 10 shows the steps the interface goes through to allow the user to calculate the full optimization results. In "Timewindow", an estimate is made of how much time will be required to run all combinations of the data dimensions on the system platform. This is a system dependent timing estimate. Selecting the "Full Optimization" button will execute the optimization algorithm. The window named "Procwindow" is displayed while the results are generated. Selecting "Exit" will stop the interface from proceeding with the optimization run.



Mask	Total	Ok		Out		Nok		Mult		Nok+Mult	
06D\1011111s:TOTALS:	802	783	97.6%	3	0.4%	1	0.1%	15	1.9%	16	2.0%
06D\1110111s:TOTALS:	802	779	97.1%	3	0.4%	2	0.2%	18	2.2%	20	2.4%
05D\1010111s:TOTALS:	802	772	96.3%	10	1.2%	5	0.6%	15	1.9%	20	2.5%
07D\1111111s:TOTALS:	802	777	96.9%	1	0.1%	2	0.2%	22	2.7%	24	2.9%
05D\1011101s:TOTALS:	802	761	94.9%	15	1.9%	4	0.5%	22	2.7%	26	3.2%
05D\0110111s:TOTALS:	802	759	94.6%	14	1.7%	3	0.4%	26	3.2%	29	3.6%
06D\0111111s:TOTALS:	802	766	95.5%	5	0.6%	3	0.4%	28	3.5%	31	3.9%
06D\1111101s:TOTALS:	802	762	95.0%	7	0.9%	3	0.4%	30	3.7%	33	4.1%
05D\1100111s:TOTALS:	802	754	94.0%	14	1.7%	7	0.9%	27	3.4%	34	4.3%
06D\1101111s:TOTALS:	802	758	94.5%	7	0.9%	5	0.6%	32	4.0%	37	4.6%
05D\1110101s:TOTALS:	802	750	93.5%	13	1.6%	4	0.5%	35	4.4%	39	4.9%
05D\1110011s:TOTALS:	802	746	93.0%	14	1.7%	10	1.2%	32	4.0%	42	5.2%
05D\0011111s:TOTALS:	802	746	93.0%	11	1.4%	6	0.7%	39	4.9%	45	5.6%
04D\0010111s:TOTALS:	802	725	90.4%	32	4.0%	9	1.1%	36	4.5%	45	5.6%
05D\0101111s:TOTALS:	802	744	92.8%	12	1.5%	6	0.7%	40	5.0%	46	5.7%

Figure 11. Files generated by optimization algorithm

Comprehending Optimization Results

The optimization algorithm creates a storage directory structure and saves the results as they are computed. The directory structure where the results are stored is created one directory level under the directory that was selected in the "File" menu under the "Select Results Directory" option. Figure 11 shows an example of the directories created and the PR result files that are generated when the optimization algorithm is executed.

Figure 11(a) shows the upper level directory where the results are stored. In this example the user selected a directory named "voc1". Two files and another directory are created in this directory. The file named "do-VERImatrix.bat" is a batch file that is created from the information the users supply to the interface. It is executed by the interface to run the optimization code. The file named "lvo.txt" is a command file. Its contents are the input parameter strings read by the optimization algorithm. Neither of these two files should be needed by the user, they remain in the results directory for historical reasons. The directory named "veri-matrix" is where the final optimization results are stored.

Figure 11(b) shows the contents of the "veri-matrix" directory for a data set with seven dimensions. Separate directories are created within "veri-matrix" to store the results from examining all the combinations of features in each subset of data dimensions. For an N-dimensional data set, directories named 02D, 03D… ND are created. Two important result files exist in the "veri-matrix" directory. They are the files named "nok_mult" and "Totals".

Figure 11(c) illustrates a sampling of the optimization results file named "nok_mult". This file is organized with the same general structure as all the other result files. The explanation of the structure in "nok_mult" applies to the "Totals" file as well. In this sample figure there is a heading line describing the columns of data that exist under each name. This line is for illustration purposes only, it does not exist in the actual data files that are generated. The "nok_mult" file is a combination of all the output files from all the result subdirectories. In the result files there is one line of statistics for each combination of the features. The "Mask" column shows a directory name and binary bit string patterns. A "1" in the bit strings indicate which features of a data set were used to generate the statistics that exist on that line. A "0" indicates which features were not used. For example, the first line of data begins with the string "06D\1011111s:TOTALS". This string indicates that a result from the 06D directory using six features was used to generate the statistics on this line. The second feature was not used. There were a total of seven features in this data set. The "s" and "TOTALS" in the string are historical relics. Some versions of the result files may not have the directory listing (06D) in them. The "Total" column is the total number of points in the data set that is being reported. All the numbers should be the same in this column for each data set. In this example there were 802 data points examined.

The results generated by the VERI algorithm and saved in this file are based on a Leave-One-Out type of pattern recognition analysis. If the reader wants to learn more about this type of PR analysis and the results we generate, they can refer the VERI Users Guide at our website:
http://www.sandia.gov/imrl/XVisionScience/Xusers.htm. There are four types of classification results that are reported in the result file: points that grouped or classified correctly (Ok), outlier points that did not classify with other points (Out), points that classified with the wrong class (Nok), and points that classify with multiple classes (Mult). We report the number of points that occur with each of these classifications and its percentage with respect to the total number of points in the data set.

For example, continuing with parsing the first line of statistics in figure 11(c) we see that 783 (97.6%) of the points classified Ok (Ok), 3 (0.4%) of the points were classified as outliers (Out), 1 (0.1%) of the points classified incorrectly (Nok), and 15 (1.9%) of the points classified with multiple classes of data (Mult). The last two columns of results are the sum (and percentage) of the points that classified incorrectly plus the points that classified with multiple classes of data (Nok+Mult). In our example on the first line of results those values are 16 (2.0%).

Since the sum of the classification statistics is 100% on each row, small values in the Nok+Mult column indicate that Ok and Out classifications will dominate the statistics generated for a set of features. Data points that classified with the Out class are not incorrect, they did not group with any other points.

The result file "nok_mult" is sorted on a Nok+Mult column in increasing order. Sorting on this column also orders the data with the sum Ok + Out sorted in decreasing order. The number of Out classifications tends to increase proportionately to the sum Nok+Mult, while the number of Ok classifications is inversely proportional to the sum Nok+Mult. Therefore, qualitatively the best results are the ones that have the lowest values of Nok+Mult.

Figure 11(d) shows the result files that are created in one of the result subdirectories. The "Totals" files are the same format as the "nok_mult" files, the "Totals.0" files are the same format except that the Mask information is abbreviated and they do not contain the Nok+Mult column of information.

The "Totals" file in the "veri-matrix" subdirectory is a simple concatenation of the "Totals" files from all the result subdirectories 02D, 03D… ND.

While each result file can be analyzed, the preferred file to examine is the file named "nok_mult". The sorted results from all the feature combinations are stored in this file.

Part II.
Approximating Mid-Range Optimization With Upper Band Results

This section describes the option in this interface that allows the user to compute optimization results for data sets with a large number of features. Large data sets may require a prohibitive amount of computing time if all possible combinations are examined. As mentioned in part I, a data set with twenty dimensions has 1,048,555 possible feature combinations to examine. If one second of time were required to execute VERI and store the results for each combination, it would require over twelve days to complete the full set of optimization calculations. Data storage limitations become a factor as well. Computation time and data storage requirements increase exponentially as a function of the number of features in the data set.

We developed a method to find optimization solutions for problems with large numbers of dimensions [4]. The solution is an approximation, but it is possible that our technique will find the optimal feature subset as its best solution. Our research has shown that good feature subsets reoccur frequently in the best subsets of any size. Based on this, we developed an algorithm that works by finding exact (best) optimization results for subsets of dimensions in ranges where it is computationally feasible, such as where a small number of features (currently <= 5) and a large number of features (currently N-5) are examined. Using the best ranking results generated, we generate subsets of features for the intermediate dimensions using the best exact results in combination with other features. For example, in the lower range of features we use a fixed number of the best (upper band) five-feature results to generate good candidate feature subsets for the six-feature cases. In the high range of features, the upper band of the N-m feature subsets is used to generate good candidates for the N-m-1 feature subsets. This process continues until all feature subsets are examined that is, until
$I+5 = N-I-5$, where $I>=1$ and N is the total number of features or dimensions in a data set.

This section describes how to use the interface to generate upper band optimization results in data sets with a large number of features.

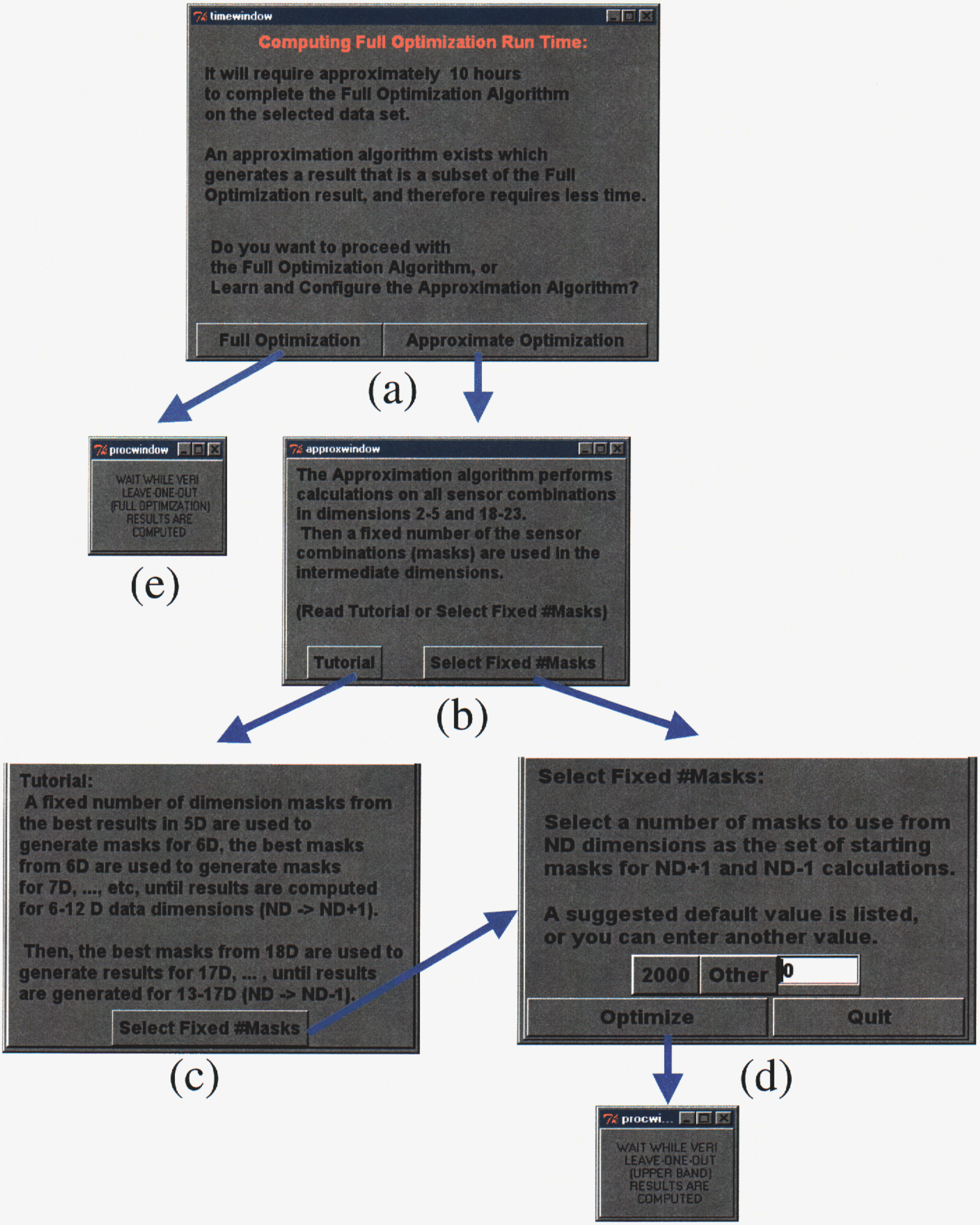


Figure 12. Configuring Upper Band Optimization

Figure 12 shows the windows that are displayed and used when the upper band mode of computing optimization results is utilized. Some windows in this group are mandatory, others exist to instruct or remind the user of how the analysis works.

Figure 12(a) shows the window from the upper band interface named "Timewindow". This window is displayed when the system estimates that the run time for the full optimization will be longer than a set threshold. The threshold is set to a time at which we estimate a user may want to consider using the upper band algorithm. In this figure the system has estimated it will require ten hours to complete the run. The time that is computed to run all the feature combinations is an approximation. It is computed by executing the VERI algorithm with one set of features and using the computed time to estimate how much run time all possible combinations of the features would require. From this window the user can proceed to either the "Full Optimization" or the upper band "Approximate Optimization" window.

Figure 12(b) illustrates the window named Approxwindow. It is displayed when the user has chosen to execute the approximate optimization algorithm. In this window mention is made of how the analysis will proceed, for example, which feature subsets will have all combinations examined. It briefly explains how the upper band of results is used to calculate solutions for the intermediate feature subsets. From this window the user can select to read more information in the "Tutorial" window or proceed to the "Select Fixed #Masks" window.

A brief tutorial sub-window is displayed when the user selects the "Tutorial" button. This is shown in figure 12(c). Here the upper band technique is explained with more case specific details. The parameters in the text are data dependent. When the user has finished reading the tutorial she can proceed by clicking on the "Select Fixed #Masks" button.

The "Select Fixed #Masks" sub-window shown in figure 12(d) allows the user to select how many of the best ranking upper band masks (feature subsets) to use when generating mask patterns to represent the intermediate feature sets. A default value of 2000 has been established as a sufficient value, but the user can enter a different value. This window can be entered from the "Approxwindow" in figure 12(b) or from the "Tutorial" sub-window shown in figure 12(c). The figure shows the entry window open and ready for the user to enter a new value. Selecting the "Optimize" button will execute the optimization algorithm, selecting "Quit" will return the interface to the main menu.

Finally, if the user selects "Full Optimization" in figure 12(a), the window named "Procwindow" shown in figure 12(e) will display on the screen until the full optimization is completed.

The results generated by the upper band algorithm are stored with the same directory and file formats as are used by the full optimization algorithm.

Conclusions

This paper describes how to use the Visual Empirical Region of Influence (VERI) Optimization Interface Tool. The interface operates in two modes, full optimization and upper band optimization. Both modes of operation are described and illustrated. The interface package was developed using the Tcl/Tk GUI programming language version 8.1. Screen shots of the interface's operations are used to describe its modes of use. Examples and advice on how to use this tool to analyze data are presented throughout the text. Examples of how to interpret the PR results are presented.

It is up to each user's discretion to determine their time constraints and whether it is more practical to user the full optimization algorithm or the upper band optimization algorithm.

A copy of the VERI Optimization Interface package can be obtained without charge by contacting the author at rfmarti@sandia.gov.

References

1 Tcl and the Tk Toolkit, John K. Ousterhout, Addison-Wesley (1994)

2 G.C. Osbourn and R.F. Martinez, EMPIRICALLY DEFINED REGIONS OF INFLUENCE FOR CLUSTERING ANALYSIS, Pattern Recognition, Vol. 28, No. 11, 1793-1806(1995)

3 R.F. Martinez, A Visual Empirical Region of Influence Pattern Recognition Tool For Leave-One-Out Data Analysis, SAND2002-0702, Sandia National Laboratories (2002)

4 G. C. Osbourn, R. F. Martinez, J. W. Bartholomew, W. G. Yelton and A. J. Ricco, Optimizing Chemical Sensor Array Sizes, Proceedings ElectroChemical Society Mtg., ECS 99-23, p. 127 (1999).

Distribution:

	MS	Name, Org.
15	1423	R. F. Martinez, 1118
1	1423	G. C. Osbourn, 1001
1	1423	J.W. Bartholomew, 1118
1	1423	G. N. Hays, 1118
1	1425	R. C. Hughes, 1744
1	0892	R. W. Cernosek, 1764
1	1425	W.G. Yelton, 1743
1	1425	C. E. Davis, 1744
1	9018	Central Tech Files, 8945-1
2	0899	Technical Library, 9616
1	0612	Review and Approval Desk, 9612