Getting Started

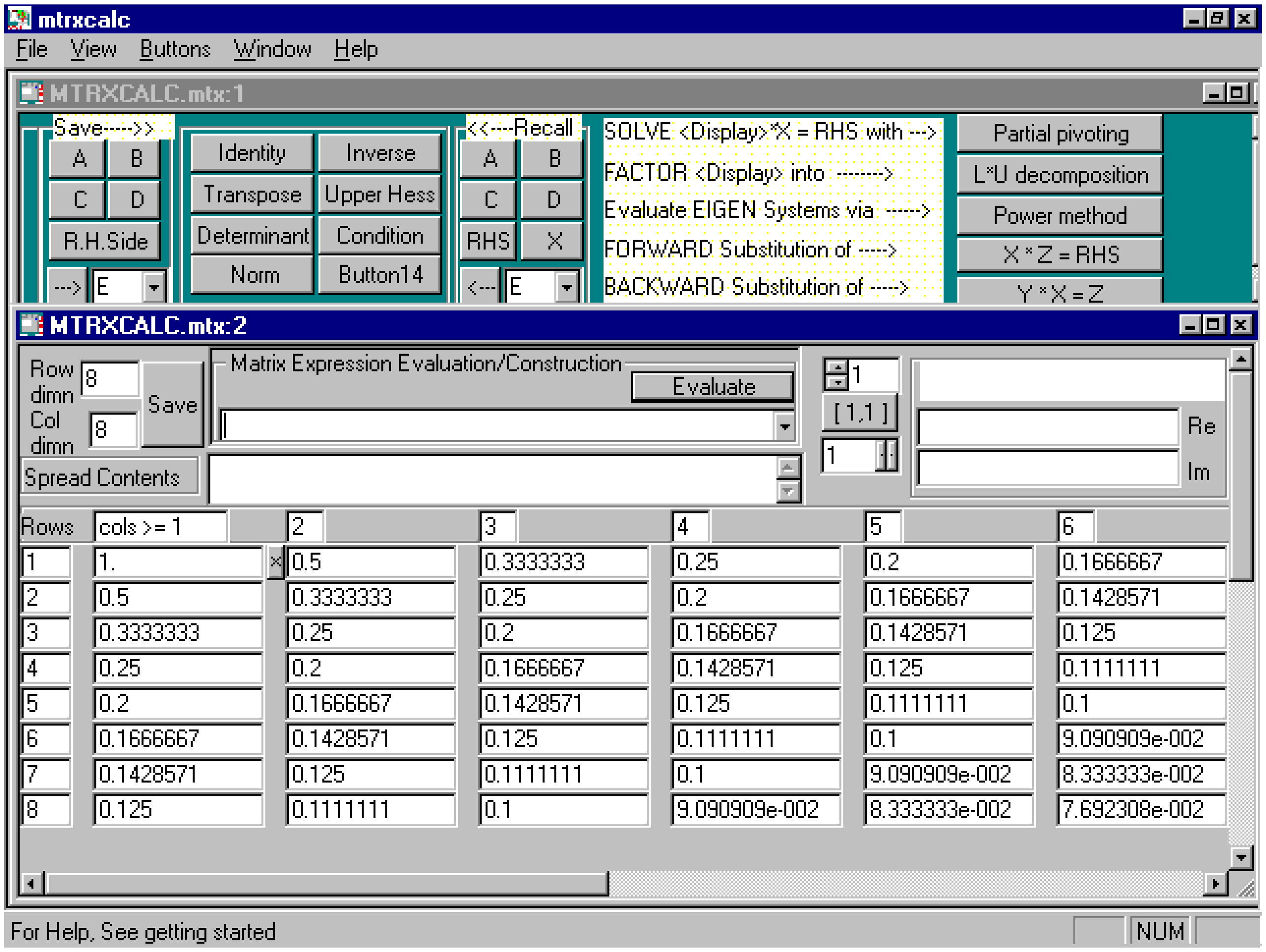
Any of the standard methods of starting an application in Windows 95™ or Windows NT™ should work with the appropriate **MtrxCalc V2** but with the following exceptions: A file with filename **MtrxCalc.dll** or **MtrxCalc.mtx** must reside in the same directory as **MtrxCalc.exe** (else you will have to locate it for **MtrxCalc** each time you start.) One of the above files is required for start up and is provided with the setup files.

MtrxCalc opens with two separate windows. The two windows provide the user interface to the computing features of this program. The starting positions of the windows are in a tile mode (actually slightly overlapped) designed to facilitate the interaction of the two. The upper window contains the calculator emulating buttons and is best left in this position until you are very familiar with the workings of the program. The lower window contains the display view (often referred to herein as <Display>) which performs as the method of input, output and display. The two windows should look much like the. bitmap below if your resolution is 800 by 600.

If the position of your windows misses the mark you should be able to drag and resize the windows to the position best suiting you. You may on many occasions wish to maximize the lower window to view a larger display area and then restore it to the starting position when convenient. ***We repeat,*** for most applications you should not have to move the upper calculator window once properly positioned. However, once you are well acquainted with this program you can do without the upper window entirely. All the functionality of the buttons (in the upper window) can also be done from the lower window.

This topic contains jumps to almost the entire help file contents. If you are just getting started then you may wish to leave this open for quick reference to the MtrxCalc application.

Click on various areas of the bitmap below to see the features explained here.


mtrxcalc
File View Buttons Window Help
MTRXCALC.mtx:1
Save---->>
A B C D R.H.Side ---> E
Identity Inverse Transpose Upper Hess Determinant Condition Norm Button14
<<---Recall
A B C D RHS X <--- E
SOLVE <Display>*X = RHS with --->
FACTOR <Display> into -------->
Evaluate EIGEN Systems via ------>
FORWARD Substitution of ----->
BACKWARD Substitution of ----->
Partial pivoting
L*U decomposition
Power method
X * Z = RHS
Y * X = Z
MTRXCALC.mtx:2
Row dimn 8
Col dimn 8
Save
Spread Contents
Matrix Expression Evaluation/Construction
Evaluate
1
[1,1]
1
Re
Im
For Help, See getting started
NUM

Rows	cols >= 1	2	3	4	5	6
1	1.	0.5	0.3333333	0.25	0.2	0.1666667
2	0.5	0.3333333	0.25	0.2	0.1666667	0.1428571
3	0.3333333	0.25	0.2	0.1666667	0.1428571	0.125
4	0.25	0.2	0.1666667	0.1428571	0.125	0.1111111
5	0.2	0.1666667	0.1428571	0.125	0.1111111	0.1
6	0.1666667	0.1428571	0.125	0.1111111	0.1	9.090909e-002
7	0.1428571	0.125	0.1111111	0.1	9.090909e-002	8.333333e-002
8	0.125	0.1111111	0.1	9.090909e-002	8.333333e-002	7.692308e-002

For smaller resolutions or zoomed views the cells in the <display> should adjust for the new font size automatically. Since all large matrices are viewed by partitions one can also change the number of columns in each partition to suit that view. We hope this feature will allow this application to be useful to those requiring/desiring zoomed views.

Entering a matrix

How To Use The Calculator '<display>' Form...

A matrix can be created by entering the data in the display array, entering the dimensions in 'row dimn' and 'col dimn', depressing the Save button to register the changes ***and then*** transferring the data to the chosen matrix name (A,B,C,D,E,...Z).
For example, to enter an element in row 2 column 3 go to the row labeled 2 (row labels are on far left hand side) then go across to column labeled 3 (column labels are across the top). (You may use the tap key to move down, the shift-tab key to go up, the **alt +** arrow keys to go up, down, left or right. or one may merely click the appropriate cell with the mouse cursor.) At this point in the <display> type the value to be recorded in row 2 and column 3 of the edited (or new) matrix. ***If the element is complex type the real part of the element then. if in complex display, drop down in the same column to the row labeled '+i' immediately below this entry, and here type in the imaginary part of the element.*** Once you have finished editing the matrix entries you will need to depress the Save button for the changes to be implemented.
Warning! The Save button in the <display> window only saves the <display> to a temporary matrix named **U** and is not in safe memory. While the buttons in the save group save the matrix **U** into the matrix of your choice. Please read further.

If you are entering a matrix with at least one nonzero imaginary part into the <display> of a real matrix, then you need to **force a complex display view**. To force the complex display go the View menu item and choose Force Complex Display. Except for the matrix you may be currently entering the matrices are smart enough to know if they have a non zero imaginary part.

If you need to enter a value such as $\sqrt{5}$ you must use the special notation of the built-in parser. For example typing sqrt(5) in the real part of row x and column y will evaluate to 2.23606797xxx once the SAVE button is depressed. Typing sqrt(i(2)) will load **1+i** in the appropriate parts of that row & column location. One may enter expressions like sin(pi/4+i(4)) for
$\sin(\pi/4+4i)$ and so forth. You may also use the set function

Please note that the next matrix entered in the <display> will destroy the old matrix there. To avoid this save to a letter like A,B,C,...
(See Matrix Expression Evaluator , the function setSET, the Save--> group of buttons, and the **init** Examples **)**

Save Button

If you enter values directly into the <display> area   targeted for the <display> matrix **U** you will need to depress the Save Button to effect the registering of the new data.   Do not use this button unless you want to overwrite the current **U** matrix with the values in the current display.   This registration is automatic for some buttons.   This not the same as saving a file.   See Saving a File

Transfer Buttons
The buttons in the ***Save --->>*** and ***<<---Recall*** groups.
Buttons under the ***Save -->>*** group transfer the contents of the spread sheet (<display>) into memory with the name of the button depressed. For example depressing button A will save the contents of <display> into ***'A'***. The Dimensions of A, B and so on will be determined by the dimensions in the edit boxes for ***row dimn*** and ***col dimn*** at the time of transfer.
The buttons in the ***<<---Recall*** group work in the reverse mode transferring the contents of A, B, etc. into the spread sheet.
Both groups of transfer buttons have a **drop-down button** with name ***E*** showing at start up. One can simply type X,Y or Z etc. into the drop-down box (white square) and cause transfers to or from ***X,Y or Z*** depending on the transfer group.
Both transfer groups have a button named ***RHS*** this button is needed to set (or recall) the Right Hand Side of the linear equation ***(<display>)*X=RHS***. You will note as you use this program the matrix in <display> is named U. See Solving Linear Systems

One can also use the Matrix Expression Evaluator for transfers. Typing >>Y , will transfer from <display> to Y. Typing just Y , will recall the matrix saved under name Y. Do not type <<Y it will not parse correctly.
For more on buttons and controls see any of the following:

Identity displays the identity of the matrix in the display (if row and column dimensions are equal).

Transpose computes the conjugate transpose of the matrix in <display>. Namely U

Determinant computes or retrieves the determinant.

Norm a button to compute the induced norm of the 1_norm metric.

Inverse computes the inverse if it exists.

Upper Hess for finding the upper Hessenberg form of a matrix.

Condition a button to compute the condition number of a matrix.

Solving linear systems used to solve linear systems by various methods.

Factoring for LU, QR and Cholesky factorizations.

Eigen Systems for algorithms to find approximate eigenvalues.

Row and Col Dimn to set the dimensions of the matrix in the spreadsheet

Other Buttons

Identity displays the identity of the matrix in the display (if row and column dimensions are equal).

Transpose computes the conjugate transpose of the matrix in <display>.   Namely U

<u>Determinant</u> computes or retrieves the determinant.

Norm a button to compute the induced norm of the 1_norm metric.

Inverse   computes the inverse if it exists.

Upper Hess   for finding the upper Hessenberg form of a matrix.

Condition a button to compute the condition number of a matrix.

Solving linear systems used to solve linear systems by various methods.

Factoring for LU, QR and Cholesky factorizations.

Eigen Systems for algorithms to find approximate eigenvalues.

Row and Col Dimn to set the dimensions of the matrix in the spreadsheet.

Partition View   For viewing larger matrices.

init Examples
Typing the following in the matrix evaluation edit control

init("sin(x)*ln(y)",10,10)

will construct a 10 by 10 matrix whose element in row x and column y is sin(x)*ln(y)

Any expression of the form

init("expression in x,y",m,n)

can be used to construct an m by n matrix.

An example of the more general form of initializations follows:

init("expression in x,y",8,8,"x","8")

which constructs an upper triangular matrix of order 8x8.

Note if you need to save one of these init results you must use one of the transfer buttons,

or otherwise it is only saved in the temporary matrix **U**.

Other examples are:

A **Hilbert** matrix of order 10x10 can be constructed by entering:

init("1/(x+y-1)",10,10)

A **tri-diagonal** matrix of order 8 can be constructed by entering:

init("1+abs(x-y)",8,8,"x-1",x+1")

The general form of init is: **init**("*expression_1 in x and y*", *integer value for row order*, *integer for column order*, "*expression_2 in x* ", "*expression_3 in x* "). Expression_1 is evaluated and placed in row x and column y. Expression_2 is evaluated and used to determine in which column in row x does the first possible non-zero value occur in this row, expression_3 gives the column after which all values are zero in this row. This allows us to construct banded, triangular and diagonal matrices.

Several examples are already entered in the drop-down list box control for Matrix Expression Evaluation

Solving linear systems <display>*X=RHS
Gauss elimination

Solving linear systems

The calculator provides buttons to perform automatically some standard methods of solving linear systems of equations such as gaussian elimination with complete pivoting (**Complete pivoting**) or partial pivoting(**Partial pivoting**) and the QR factorization method(**Q*R factors**).   Each of these requires a matrix in **<display>** and a matrix in **RHS**. **Each will solve <display>*X=RHS** destroying what may be stored in **X**.

Less automatic methods, methods much like those on some calculators, can also be performed.

See gauss elimination for the step by step approach.

See Example: Automatic methods for the more automatic methods.

Factors Button
The button face Cholesky factors yields the matrices G and trans(G) the Cholesky factorization (as in <display> = G*trans(G)) of a positive definite symmetric/Hermite matrix. At termination <display>=G, X=G and Y=trans(G); where G is lower triangular. One can solve the system <display>*X=RHS by using the Forward Sub button to solve X*Z=RHS and then using the Backward Sub button to solve trans(G)*X=Z.

The button L*U decomposition yields the LU factorization of the <DISPLAY> matrix. Where U is an upper triangular matrix and L is a unit lower triangular matrix. At termination <display>=U, Y=U and X=L. You may solve LU*X=RHS by placing an appropriate right hand side in RHS, use the Forward Sub button to solve X*Z=RHS, then the Backward Sub button to solve Y*X=Z. The solution will now be in X so that *L will be lost* if not saved elsewhere.

The button Q*R factors yields the QR factorization of the <display> matrix. Q is orthogonal and R is upper triangular. At termination <display>=R, Y=R, X=Q, and Z=trans(Q)*RHS. Backward Sub will solve R*X=trans(Q)*RHS but *Q will be lost.*

See Solving linear systems

Eigen systems
The Power method button performs a sequence of high level applications of the inverse power method on the matrix in <display> using the matrix in RHS as the 'initial vector'.   The default RHS is a column of ones.   The 'vector' in RHS is updated each time Power is depressed while <display> is unchanged. Thus you may repeat this operation several times to improve the estimate.   The RHS yields the eigenvector estimate, and each element of the column matrix in **X** is an estimate of the eigenvalue with largest magnitude.

The Upper Hess button performs Householder reflections on **<display>** until it is upper Hessenberg.

The QR Step button performs the qr algorithm on **<display>**,   The results after a number of steps proportional to the dimension of <display> are presented in the <display> .   The approximate eigenvalues are on the main diagonal of <display> and the corresponding approximate eigenvectors are in **Y**.   Please see QR Step

QR Step button
The button QR_Step activates a sequence of QR factorizations of the matrix in the Dialog Box 'Dlg'. The number in the sequence of steps is proportional to the order of the matrix. The full number of steps may not be performed if convergence criterion is met along the way. This algorithm can be re-applied to the previous result if convergence is not adequate. Transformation to Hessenberg form is automatic so using the **HESS** button is not required.

This is a high level method in that the algorithm uses modules at the many steps along the way. Thus it is relatively slow. The accuracy is respectable for real matrices and marginal for complex. Approximate eigenvectors are computed as well making the procedure even slower.

The method for determining the eigenvectors is the unstable method often seen in undergraduate courses of solving

(A-sI)v=0

in which s is an approximate eigenvalue and thus the system is very ill conditioned. This method is not reliable and especially so in the complex case and is offered "AS IS" only for limited use. The first updated version will have a more stable method. Having warned you about this I will now say that it is not in many cases as poor a performer as it sounds for use within the reals.

As a short **example** execute the init statement in the dropdown list of the matrix evaluator that

reads as follows: **init("x+y^x",5,5)**. Save this in say D and then depress the **QR Step** button to obtain an approximate upper triangular matrix whose diagonal elements are the approximate eigenvalues. In this example 3351.--- the largest eigenvalue should appear in the 1,1 position and 0.641--- the smallest should appear in the 5,5 position of the resulting matrix. In the title edit box the statement '<display>=last matrix in QR sequence, approx eigenvalues are in **Y**' should appear. So transfer Y to <display> (typing Y in the matrix evaluator and depressing enter will do it). The matrix displayed should contain 5 independent columns corresponding to the respective eigenvalues found earlier. To check on the independence evaluate the determinant. Also evaluate D*Y. Does the expected result develop?

One can perform a step by step QR algorithm by applying the following steps:

0) Enter a matrix and save into say A,
1) Depress the **QR Factor** button (note that Q is in X and R is in Y),
2) Type Y*X in the **Expression evaluator** and depress the **evaluate** button (call this A_2)
3) Go back to 1) and repeat as often as needed.

See Eigensystems

The matrix expression evaluator
Matrix operations such as **inv(A)**, **abs(B)** or **A*B+inv(trans(X)*X)*trans(X)*Y** can be entered by typing them directly in the **matrix expression** edit box and then depressing the button **Evaluate.** The functions inv, abs and trans(pose) must be typed in lower case. The matrices A, B, & etc. must be typed in upper case. Please look at the drop-down list in this control for some examples. You may highlight one of these, say the first one on the list, and use it as a pattern to make your own version. If you have not visited the init function definition and examples now may be a good time to do so. The init function examples

Note if you need to use the matrix **<display>** currently in the spread sheet use **U** in the expression as **inv(U)*U** and if you need **RHS** use **R**.

The operators {**+,-,*,%+**} are supported with the following definitions:

'**+**' matrix or scalar addition,

'**-**' matrix or scalar subtraction,

'*' matrix multiplication or scalar multiplication a matrix,

'**%+**' column-wise matrix concatenation -augmenting,

'$+' row-wise matrix concatenation - stacking,

In addition to these basic operators there are many other matrix and linear algebra functions. See Matrix Functions

And other basic functions- Elementary functions init

The **matrix expression** evaluator can also be used to initialize a matrix as in

init("sqrt(2)",5,5,"x","x")

which will construct a diagonal matrix in the calculator form with root 2 on the main diagonal.   See Entering a Matrix

The general form of init is**: init("expression_1** in x and y**", integer value** for row order**, integer for** column order, **"expression_2** in x **", "expression_3** in x **").**

Expression_1 is evaluated and placed in row x and column y.   **Expression_2** is evaluated and used to determine in which column in row x does the first possible non-zero value occur in this row, **expression_3** gives the column after which all values are zero in this row.   This allows us to construct banded, triangular and diagonal matrices as you see above.

set

Typing the following into the evaluation control

set("ln(x)",8.7)

will initialize the element in row 8 and column 7 to ln((x).

The general form of set is:

set(“expression”, x,y)

Where ‘expression’ is an algebraic expression in x, y, and the usual calculator functions such as sin, cos, log, exp,acos,atan,...etc.   While ‘x’ is an integer denoting the row index and ’y’ the column index.

@

The matrix expression evaluator can also be used as a **complex scalar calculator** to evaluate elementary expressions.   To do so, prefix the expression by **@** as in **@”sin(pi/4)^4”**.   The results will appear in the **Calculator register**.   If it is more convenient you can type expressions directly in the real or imaginary part of the scalar calculator registers, such as sqrt(5) for example.

See Elementary functions

rows

By typing rows(1,3..5,8,9) in the Matrix Expression Evaluation control and then depressing execute, you construct a new <display> matrix comprised of the following rows of the original <display>:
    1,3,4,5,8,9.

In general you must type row or rows with an argument list of integers separated by commas or double periods. If an integer is preceded by a comma or is the first integer in the list, then that row will be included in the rows of the new <display>. If an integer k is preceded by a pair of periods then all the rows since the last added row up to and including row k will be added. The row dimension of the new matrix will changed appropriately

cols

The function cols is analogous to the function rows. By typing cols(1,3..5,8,9) in the Matrix Expression Evaluation control and then depressing execute, you construct a new <display> matrix comprised of the following columns of the original <display>:   1,3,4,5,8,9.

In general you must type col or cols with an argument list of integers separated by commas or double periods. If an integer is preceded by a comma or is the first integer in the list, then that column will be included in the columns of the new <display>. If an integer k is preceded by a pair of periods then all the columns since the last added column up to and including column k will be added. The column dimension of the new matrix will changed appropriately.

If one would like to find a basic solution corresponding to a basis chosen form any, say m, columns of a given matrix as is common in *linear programming* then the cols operation is useful. To be more specific given the system Ax=b with A of order m by n with m<n and rank(A)=m, use cols to construct a matrix B using any m columns of A and then solve Bx=b by any method appropriate. The same applies to constructing a linear model X=Y for solution by *least squares*. Here A is of order n by m with n>m, select p columns of A to describe your model place in X and insure that rank(X)=p, then type 'inv(transpose(X)*X)*transpose(X)*Y' in the matrix expression evaluator matrix expression evaluator to obtain the solution.

exrows

You can exchange any two rows in the <display> matrix using the function exrows( ).   For example exrows(3,5) will exchange row 3 and row 5.

The function excols( ) works in the same way.

elmrowop

The function elmrowop performs the elementary row operation of replacing one row ,say row 3, with row 3 + (1/3)* row 5 elmrowop(3,5,1/3). Or in general elmrowop(j,k,c) will replace row j by rowj+(c)*rowk. See Gaussian elimination

Gaussian Elimination
You can perform gaussian elimination and similar methods based on elementary row operations using step by step calls of exrows andelmrowop functions. The augmented matrix can be constructed with the **%+** operator.

As an example of this tedious but sometimes informative alternative to the automatic methods provided here, start with the matrix given by typing init("x+y^x",8,8) in the **Evaluator** execute and load in A. Our intention here is to find the inverse of the submatrix of A formed from the first three rows and columns. You can use the functions rows and cols to do this. Typing 'rows(1..3)' in the Matrix Expression Evaluation control (or finding it in the drop down list box) will create a submatrix of 3 rows by 4 columns using the first three rows of A. Save this to **A** <display>-->A and then type (or find in the list box) 'cols(1..3)' . Save this to **A** also so that now you have the desired 3 by 3 in **A**.

To construct the 3 by 3 **I**dentity matrix we can use the Identity button or the init function and type init("1",3,3,"x","x") in the evaluator control. Save this result to **B**.

To build the augmented matrix A **|** **I** we will use the %+ operator and simply type A%+B in the same evaluator control. Save this in **C** ,in case you need to start again.

As a first step in a *follow along mode* we will divide the first row by 2, to do so type elmrowop(1,1/2) in the evaluator control hit enter and this will multiply row 1 by 1/5. So now the element in cell row-1 and column-1 is 1. The next step is performed by typing elmrowop(2,1,-3) followed by enter hereby subtracting 3 times row-1 from row-2 and putting the result in row-2. Next type elmrowop(3,1,-4) and enter so that col-1 now consists of all zeros except for the 1 in row-1 col-1. This finishes our work with column one.

For column two the sequence is

elmrowop(2,1/1.5)
elmrowop(3,2,-5)
elmrowop(1,2,-1.5)

For column three:

elmrowop(3,3/16)
elmrowop(2,3,-10/3)
elmrowop((1,3,3)

Now the last three columns hold the inverse of the original first three columns, A. You can finish this example by typing cols(3..5) and storing this inverse somewhere. I am sure this will not be your favorite method. However I would like to note one advantage this tedious approach offers. In the event that a near-zero value occurs at some step which you know should be replaced by an exact zero, then you can do so; something much harder to do in compiled code. Another use is a step by step approach to the Simplex algorithm when applied to (m x n) matrices of rank m, m<n, (occurring in small linear programming applications)

LU Factors
There are several variations provided of the LU factorization of a matrix say A.
The **LU decomp** button decomposes as in (A=<display>)=L*U where L is a lower triangular matrix with units on the main diagonal. The matrix U here is upper triangular. One can use the keyboard for this and type **lu_decomp(U)** or **lu_decomp(A)**. (Recall <display> is same as the matrix U)
The **Partial pivot** button decomposes as B=L*U where B=A except for possible *row* exchanges. However the partial pivot, and also the complete pivot, procedures do more than decomposing into factors. Each of these algorithms solve the equation **(matrix_1)*(solution matrix)=martix_2** and require two matrices. To make use of the preprogrammed buttons for these methods matrix_1 must be in the <display> and matrix_2 must be in the matrix named RHS. To execute by keyboard type **p_pivot(A,B)**, or **p_pivot(U,R)** or in general **p_pivot(matrix_1, matrix_2)**
The **Complete pivot** button solves by decomposing B as B=L*U where B=<display> except for possible *row* and/or *column* exchanges. Or by keyboard **c_pivot(matrix_1,matrix_2)**.

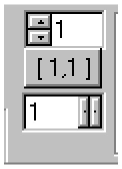
See Examples
See Init() function Examples for examples on initializing or constructing matrices.
See Gauss elimination for a step by step row reduction of a matrix.
See Automatic Solutions for the automatic methods of solving linear systems
See QR Step button for an example of the QR algorithm for finding eigenvalues.
See Elementary Statistics Example

Partition View
In MtrxCalc one will view larger matrices by partitions as opposed to scrolling. For an example, to view a real matrix of 35 rows and 35 columns beyond the 20 rows by 8 columns we see in the original <display>. We imagine a partitioning of the 35 rows into two rows of 20 and 15 rows each; and the 35 columns into five columns of 8, 8, 8, 8 and 3 columns each. Looking something like the below..

$$\begin{bmatrix} [11] & [12] & [13] & [14] & [15] \\ [21] & [22] & [23] & [24] & [25] \end{bmatrix}$$

The submatrix in the [11] partition will consist of rows 1 through 20 and columns 1 through 8, the submatrix denoted by [12] will have the same 20 rows but columns 9 through 16 and so forth with submatrix [15] consisting of rows 1 through 20 and columns 33 through 35. While submatrix [21] will contain rows 21 through 35 and columns 1 through 8.
The original display will always be the partition [11] . We decide which part of the matrix not in view we wish to see next and then depress the appropriate **[X]** button to the right of the corresponding cell. For example to view the submatrix [2,3] depress the **[X]** on the right end on the cell in **row 2** and **column 3**.

One may also use the spin controls to navigate the partition views. The upper control navigates the rows while the lower changes the column index of the partition.



For the larger matrices the partition viewing gives much better control than does scrolling.
See Building Larger Matrices

Building Larger Matrices

This application is designed as a small matrix application but can be extended to larger (but still limited) matrices by building your matrix by partitions. Suppose you need a matrix with twice the number of columns allowed in the display and twice the number or rows. You can build this matrix in several ways. One way is to set the total dimensions of the desired matrix into Row and Col Dimn. hit the Save button and a matrix of that size will be automatically entered into the display. All of the previously unused cells well be filled with zeros. It will now be your task to change these to the values you require. Be sure to save the resulting matrix under some name before it is lost. You may even want to save it to a file if it is too large and tedious to enter again.

One could also use the row and column concatenation to build large matrices from smaller ones.

To view this larger matrix see Partition View . For future reference **%+ 'augments'** columns and $+ 'stacks' rows.

Also don't forget the init function explained in init() function examples allows one to build larger matrices all in one stroke i.e. init("1/(x+y-1)",18,18) will construct a m=18 by n=18 matrix and store it in U while displaying the northwest partition in the <display>.

Elementary functions

You may type expressions involving elementary functions into cells and have the expression parser and evaluator replace the expression by its numeric value.   For example typing sqrt(2)/2 or sin(pi/4) will both result in the value 0.707106..... when the display is saved to some storage space.

The following algebraic operators can be used in the expressions mentioned above:

+, -, *, /, ^

as well as:

- absolute value--abs()
- square root--sqrt()
- imaginary part--i()
- imaginary part--imag()
- real part--real()
- complex argument--arg()

Trigonometry:

- sine--sin()
- cosine--cos()
- tangent--tan()
- secant--sec()
- cosecant--csc()
- cotangent--cot()
- inverse sine--asin()
- inverse cosine--acos()
- inverse tangent--atan()

Other transcendental functions;

- natural logarithm--ln()
- common logarithm--log()
- exponential--exp()
- hyperbolic sine--sinh()
- hyperbolic cosine--cosh()
- hyperbolic tangent--tanh()

Example 1

Example1. For this example start with the command init(“x+y^x”,10,10). Type this command into the Matrix Expression Evaluation/Construction control box. (We will call this control the **evaluator** or **parser**). Depress the evaluate button or the **enter key**. Next save this matrix into A by using the **Save --> group** button **A** or by typing **>>A** into the evaluator. Now type **A*A** in the evaluator and save the result into RHS. and also save into B. (If you use the evaluator to save into RHS type >>R not >>RHS). Now we have a matrix named A and a matrix named B with B=A*A. Notice that only 8 columns of A are displayed to see columns 9 through 10 click on the partition button (a button marked with X) to the right of cell in row 1 and column 2. This will give a view of the [1,2] partition - See the partition buttons). Then when ready to come back to the first view click button with face [X] located to the right of the cell in row 1 and column 1.

Wenow solve the linear system A*X=RHS by using the button partial pivoting located to the right of the Solve <Display>*X=RHS caption. To do this we will be using the gaussian elimination method with partial pivoting. The button referenced by Solve <display> * RHS with ---> is capable of showing several faces to allow several choices of the method to apply. In this case make sure the button displays the face Partial pivoting. If you need to, refer to the Changing buttons. To use this button method we must have A displayed in the spread sheet (<display>) because the ‘button’ expects to solve <display>*X=RHS and we want to solve A*X=RHS.

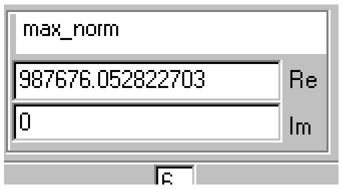
Depress the Partial pivoting button now and the solution will show in the display. The contents title will tell you this and also inform you that the solution is also saved into the matrix named X. So if you want to see it again a little later just recall X to the display. However matrices named X through Z are not safe storage areas so while **X** is in the <display> type **>>C** or otherwise save to **C**.

While the buttons are convenient and fast they are not nearly as flexible as the evaluator/parser.

So as an alternative to the ‘button-use’ above one could type p_pivot(A,B) into the evaluator and the result will be the same as before. In general p_pivot(A,B) will solve the linear system A*X=B.

Likewise you may use of the buttons Complete pivot, Partial pivot or QR Solve. (Actually for this matrix you can also use several of the factorization methods as well).

As you can see the solution is not accurate to the precision of the machine. The exact solution to the intended system is original matrix A. So the matrix **A-C** will give the error in our solution. Type A-C in the evaluator and inspect the result. For a single-number (scalar) representation of this error let us use the matrix norm called the max.-norm or infinity norm. To evaluate this type norm_max(A-C) into the evaluator. the value obtained should be approximately 0.99e-06. You should see the value reported in the real part of the scalar calculator control


```
max_norm
987676.052822703   Re
0                  Im
```

Now let us use the complete pivot method. See the changing buttons jump above to change the face and operations of this button used in partial pivoting. Alternately we can just use the keyboard and evaluator as in the below sequence:

c_pivot(A,B)	calls the complete pivoting algorithm to solve **A*X=B**
>>D	saves the result to **D**.
norm_max(A-D)	computes the infinity norm as above

In this case complete pivoting gives a dramatic improvement to partial pivoting. This is not always the case and often the improvement is very slight. Compare this to the solution given by using the LU decomposition.

More About

Ombudsman

This program is produced by a member of the Association of Shareware Professionals(ASP). ASP wants to make sure that the shareware principle works for you. If you are unable to resolve a shareware-related problem with an ASP member by contacting the member directly, ASP may be able to help. The ASP Ombudsman can help you resolve a dispute or problem with an ASP member, but does not provide technical support for members products. Please write to the ASP Ombudsman at 545 Grover Road, Muskegon, MI 49442-9427 USA, FAX 616-788-2765 or send a CompuServe message via CompuServe Mail to ASP Ombudsman 70007,3536.

Support for this program.

MTRXCALC is distributed as *shareware* and if this application is useful to you please let us know how to improve it. We do hope that you find it easy to use. If you have some trouble running this program or you would like to see this application expanded in some area please drop us a line via one of the addresses below. Be sure to include your own return address in the case that further communication is necessary to clearly define your suggestion.

Merritt_S@msn.com	***on internet***
GO wyesoft on MSN	***support area on Microsoft Network***
http://www/augusta.net/dsugg/wyesoft.htm	for ***World Wide Web page***

(If you are running Windows 95 Click this for MSN support
and/or
Wye Software
19 Vista Drive
Little Rock AR 72110-1714 via US Mail

Support for your questions on how to use this application is provided by the means above. Questions and comments are welcome. Please do not register this application if you have hardware incompatibilities. On the other hand if there is some small adjustment that can be made to this application to make it work for you please let us know.

Save to memory These buttons will save the contents of the matrix in the spread sheet to a memory location with name **A**, **B**, **C** and etc.. That is, if you depress the button in this group labeled **A** then the matrix will be saved to memory and can be recalled with name **A.** See Transfer Buttons for more.

Recall Buttons These buttons perform the opposite operation to the Save Buttons. Depressing the button in this group labeled A will transfer the matrix stored in memory under the name A back to the spread sheet. See Transfer Buttons for more

Save To drop down. This drop-down list box allows you to type in or select other letters to use for saving your matrix.   The choices are limited to single capital letters **A,B**,...through **T**.   See Save to memory

Recall drop down. Type the name in the drop-down list box of the matrix you wish to recall back to the spread sheet, and the depress the **<<---** button.   For example type **Z** in the box and depress **<<--**- and the matrix previously saved in **Z** will be transferred back into the spread sheet.

Identity matrix. This button will clear the spread sheet and insert an identity matrix. The dimensions will be that of the current value of the row dimn and col dimn so make sure they define a square matrix.

Transpose matrix. This button will take the transpose of the matrix in the spread sheet, or the conjugate transpose if complex. The original matrix will be lost if not saved previously.

Typing trans(U) in the evaluator box will do the same, typing trans(C) will present the transpose (conjugate transpose if complex) in the <display> (**U**), but will not destroy **C**.

Inverse matrix. This button will find the inverse of the matrix in the <display> if possible.   That is if the matrix is U then U*inv(U)=inv(U)*U=**I** the identity matrix. The original matrix will not be saved, so save it first if you will need it later.

Typing inv(A)   or inv(B) or inv(E) in the evaluator box will evaluate the inverse of **A,B** or **E**, without destroying the original matrix.

The current method of finding the determinant and inverse is by L U factorization

Determinant. This button evaluates the determinant of the matrix in the spread sheet display and displays the result in the scalar register.   One can also do this by typing det(U) in the evaluator.   Also one may type det(A) or det(D) to evaluate the determinant of A or D and have that determinant saved to A or D respectively.

The current method of finding the determinant and inverse is by L U factorization

Upper Hessenberg form. The HESS button will transform the matrix in the spread sheet into an upper Hessenberg form.

| x x x x x |

| 0 x x x x | ***The resulting matrix will have***

| 0 0 x x x | ***the same eigen values***

| 0 0 0 x x | ***as the original***.

Typing hess(A) will display the upper hessenberg transform but will not change **A.**

Norm of the matrix. This button will determine the 1_norm, approximate 2_norm and the max_norm of the matrix currently in the spread sheet. The results will be displayed in the scalar register. The button will cycle through the alternatives above. If you do not want to cycle through the alternatives simply type norm_1(U) or norm_2(U) or norm_max(U) which ever is appropriate.

Condition. This button determines the condition number of the matrix in the spread sheet using the 1_norm.   The condition number for a nonsingular matrix is the norm of the matrix times the norm of the inverse of that matrix.   Typing cond(A) or cond(matrix name) will give the same results.

Solving linear systems

This button is used to solve linear systems by various methods.   Depressing this button will automatically solve the linear system U*X=RHS where U is the matrix in the spread sheet and RHS refers to the matrix stored under that name.   To solve A*X=B you must first recall A into the spread sheet (U) and then transfer B into RHS. The button face can be changed to perform various methods. See Changing Button Faces. (One can also A*X=B more directly, without buttons, by using the matrix expression evaluator

Factoring matrices.   These buttons perform matrix factorizations.   In particular the LU, QR and Cholesky factorizations.   The button faces can be changed to perform any of these methods. See Changing Button Faces.

Eigen System buttons.   This button is used to approximate the eigenvalues and eigenvectors of a matrix, say U.   To use these buttons the system must first be setup up where U is the matrix in the spread sheet. Also one may select the method to be applied by changing the button face.   See Changing ButtonFaces

Also see Eigen systems

(Any method applied by buttons can also be applied by using the matrix expression evaluator

Changing Button Faces.   Several of the buttons will allow the user to change their face so that one can select the function performed.   The menu it Buttons is used.   For example, to solve the linear system U*X=RHS by partial pivoting one may go to the menu item Buttons/Set Solve buttons/Partial pivoting and the button face will be changed immediately.   The button will remain set until changed again.   All the dynamical button faces will be under the menu item Buttons.   The choices are only active when the upper window containing these buttons has the focus.   If it does not have the focus simply click on it.

Evaluator

. This is the input control for the parser. Any command the buttons will perform (and more) can be performed here. For example inv(trans(A)*A)*trans(A)*B is one way of obtaining the least squares solution of **A*U=B**.

Startup and Saved Files

In addition to the startup files such as MTRXCALC.EXE, * DLL , *.CNT and *.HLP This application will be loaded automatically when any *.MTX file (generated by a file save) is opened. By default the program will look for a file named mtrxcalc.dll, however opening any file with extension .mtx will also open the program. To save a file see the menu item Save under the File menu

See Save

Row and Column dimensioning

The dimensions of the matrix may be entered here.   If the matrix is being entered via the <display> and not the **Matrix Expression Evaluator** then the Save button next to Row Dimn must be used to register the changes.

Spread Sheet Display

The mock spread sheet display is where matrices are entered or displayed. One may enter a value into row 2 and column 3 of the matrix in the display by finding the cell in row 2 and column 3 and simply typing the value there. If the value must be computed such as square root of 5 then the syntax of the included parser must be used. In this case sqrt(5) will work. Other computations such as sin(pi/4), 1/3 and etc. can be made as well.(Elementary functions When all the information is entered you must depress the Save button to assure that all will be computed and saved into the display. If the matrix to be entered is larger than the display area use the Partition buttons

Evaluator button

This button activates the parser and calls the library functions to execute the expression contained in the Evaluator control.   One may simple hit the ENTER key as well.

Scalar calculator register

This register is used to display various scalar computations. For examples the determinant and the various norms. One may use the @ function

For example @sin(sqrt(pi)) will evaluate ***sin()*** and show the results here

Forward substitution

Forward substitution is an algorithm for solving lower triangular systems.   Allowing us to solve for the first unknown directly, then solve for the second unknown given the now known first, and so on.

Backward substitution

Backward substitution is an algorithm for solving upper triangular systems.   Allowing us to solve for the last unknown directly, then solve for the next to last given the last (now known) and so on.

Contents title

This area is used to display the matrix titles of the matrix currently in the display as well as a running history of the operations that have been done up to date. Your may use a right mouse click allowing you to copy the contents to the clipboard and then you may save to a text file as desired.

Often the title gives information about the algorithm used to obtain the matrix. Also one may use this area to set the print title for the matrix in the display. To do this type <@”your title” in the evaluator control, evaluate and then look at the print preview.

Elementary statistics example

Suppose the statistical data of our interest has already been collected and is now in the form of a 200 rows by 6 columns matrix. Each column of this matrix contains the experimental data (observations) on the variable represented by that column. Also suppose we want the average value of each column. To compute this by matrix algebra we may use the expression (1/200)*trans(J)*X where X is the matrix containing the collected data and J is a 200 row by 1 column matrix of ones. i.e. J = init("1",200,1). However since sums and averages are needed frequently there is a built-in function to accomplish this namely c_sum() and c_avg().which computes the sum or average of the columns of a matrix. So simply entering c_avg(X) will get us our averages and the result will be a 1 row by 6 column matrix with the average of column one in the first column of the result, and so on.

In this instance we do not have a data matrix X at hand so we will generate one. Let us suppose that column one of the data is to contain 200 elements distributed about a mean of 1 with a variance of 1. Column two will have a mean of two and a variance of 1 and so on for columns 3 through 6. We may use the init() function and the u_rand() function to accomplish this. The u_rand() function generates approximate uniformly distributed random numbers.

Let us enter init("u_rand(y,0.289)",200,6) to generate this X matrix. In general u_rand(a,b) will generate approximate uniformly distributed random numbers with mean a and variance 12b*b. Save this to X and then enter c_avg(X) and we have in the display the averages we wanted.

We might also be interested in the variance covariance matrix of this data matrix X. The covariance matrix of X can be computed by the expression

trans(X-Xbar)*(X-Xbar)=(1/200)*trans(X)*(200*I-J*trans(J))*X

This might be OK for the dedicated but the built in function c_cov(X) will compute all this with less chance of keystroke error. The diagonal elements of the covariance matrix are the variances of their respective columns. You should find that all the variances are close to '1' and the covariances in the off diagonal cells should be less than 1/2 in magnitude..

c_sum
To obtain the sum of all columns in the matrix **A** type c_sum(A).

Also see c_avg

c_avg
To obtain the average of all columns in the matrix **A** type c_avg(A).

Also see c_cov

c_cov
To obtain the variances and covariance's of column data.   Typing c_cov(A) will place into the <display> a matrix whose main diagonal elements are the variances and whose off-diagonal elements are the co-variances.   That is the element in row 3 and column 5
is the covariance of column 3 with column 5.

Typing c_cov(A) will return the variances and co-variances of the columns of A.   If A is say a 200 rows by 6 columns matrix then a 6 rows by 6 columns matrix will result in which the main diagonal elements are the variances (the variance of column 1 is in the (1,1) position and so on) while the covariance of column 1 with column 4 is in cells (1,4) and (4,1).

Letting s^2_{ij} denote the (i,j) element of the result,
then $s^2_{ij} = (1/m) * \Sigma_1^{200} (a_{i,k} - \overline{a}_i) * (a_{j,k} - \overline{a}_j)$

wherein $\overline{a}_j$ is the mean of column j.

apply
The matrix operation ***apply*** will do pretty much as it says, it will apply a function to the given matrix. For example apply(ln,A) will replace the matrix named A with a matrix obtained from A by taking the natural logarithm of each element of A.
Try this if you please. Type init(“exp(x+y) “,3,4) in the expression evaluator and then save the result into A. Then type apply((ln(x)), A) and the result should be a matrix whose element in **row x** and **column y** is **x+y**.

The general form is ***apply((fn()) , X)*** , and when used the scalar function ***fn()*** is applied to each element of matrix ***X*** and the resulting matrix is displayed in the <display> and does not replace the matrix ***X*** unless that matrix name is ***U***. One can apply one’s own function as in apply((3*x*x-2*x+1) B) or apply ((sin(x)^2+cos(x)^2) A).

Display

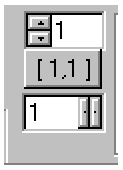
The mock spread sheet display is where matrices are entered or displayed. One may enter a value into row 2 and column 3 of the matrix in the display by finding the cell in row 2 and column 3 and simply typing the value there. If the value must be computed such as square root of 5 then the syntax of the included parser must be used. In this case sqrt(5) will work. Other computations such as sin(pi/4), 1/3 and etc. can be made as well.(Elementary functions When all the information is entered you must depress the Save button to assure that all will be computed and saved into the display. If the matrix to be entered is larger than the display area use the Partition buttons

Save buttons
These buttons save the matrix currently in the display to memory under the name chosen. For example if the button with face **A** is depressed the matrix in the display will be saved in temporary memory under the name **A**. This collection of buttons is called the Transfer buttons

Recall buttons

These buttons are also part of the Transfer Buttons group.   Depressing the button with face A will recall the matrix saved into temporary memory under the name A.   When recalled this matrix will be displayed automatically.   See the Transfer buttons

Partition spin controls
The partition view **spin controls** are used to go from submatrix to submatrix.



The upper spin control controls the submatrix row index

The button face shows the row and column index of the submatrix currently in the view.

The lower spin control varies the submatrix column index.

Calculator buttons
This group of buttons perform the elementary matrix operations indicated on their faces.

**Inverse   Identity   Transpose   Determinant   matrix N
orm   Upper Hessenberg form   Condition number**

Matrix Functions
The supported functions include the following fully programmable functions:

A abs() apply,

B backward_sub() ,
C c_avg , c_cov , c_pivot() , c_sum,
cholesky() , cond() ,
D det() , diag(),
F forward_sub() ,
H hess() ,
I inv() , init(),
L lu_decomp() ,
N norm_1() , norm_2() , norm_max() ,
P p_pivot() , power_method() ,
Q qr_alg() , qr_factor() ,
T trans() .

As well as the special functions:
@, cols, elmrowop, rows, set, exrows, and excols, ,

abs()
Typing abs(A) in the evaluator control will result in a display of a matrix whose elements are the absolute value of the corresponding elements of **A**.   However the matrix **A** will not be changed.

c_pivot()
Typing c_pivot(A,B) in the evaluator control will result in a display of a matrix which is the solution to the equation A*X=B obtained by the complete pivoting algorithm. This algorithm is essentially a **LU** decomposition with possible exchanges of rows and or columns. At each stage of the method a search is made for the largest absolute value in the part of the matrix yet-to-be-factored. Row and column interchanges are done to place that element in the [1,1] position of the yet-to-be-factored part of the matrix.

cholesky()
Typing cholesky(A) yields the matrices G and transpose(G) the Cholesky factorization
(as in A = G*transpose(G)) of a positive definite symmetric/Hermite matrix. At termination
<display>=G, X=G and Y=transpose(G); where G is lower triangular. One can solve the system A*X=B
by using the forward_sub(X, B) button to solve X*U=B and then use backward_sub(Y*X=U). Please note
that the matrix referred to as G in this discussion is **not** saved in the matrix named **G**

cond()
Typing cond(A) determines the condition number of the matrix A using the 1_norm.   The condition number for a nonsingular matrix is the norm of the matrix times the norm of the inverse of that matrix.

p_pivot()
Typing p_pivot(A,B) in the evaluator control will result in a display of a matrix which is the solution to the equation A*X=B obtained by the complete pivoting algorithm. This algorithm is essentially a **LU** decomposition with possible exchanges of rows. At each stage of the method a search is made for the largest absolute value in the first column of the matrix yet-to-be-factored. Row interchanges are done to place that element in the [1,1] position of the yet-to-be-factored part of the matrix.

lu_decomp()
Typing lu_decomp(A) or lu_decomp(some matrix name) to call for the **A=L*U** factorization, will result in the display of the composite **L\U** matrix in which the elements lij below the main diagonal are elements of **L** and the elments uij on and above the main diagonal are the elements of **U**. The matrix **L** is the lower triangular matrix comprised of ones on the main diagonal and the elements lij. **U** is the upper triangular matrix formed from the elements uij mentioned above.

diag()
The command diag(A) will display the a diagonal matrix constructed from the diagonal elements of A. The display matrix can be saved into a some name say D.

power_method
The command power_method(A) will result in the display of the last in several iterations of a **power method** algorithm. The elements of the column matrix displayed should all be approximately the same. If so then this common value is the approximate eigenvalue of A with greatest magnitude. The execution of this algorithm saves in matrix **Y** an approximation to the eigen vector corresponding to the largest absolute eigenvalue. So if the approximation is not satisfactory you may execute power_method(A,Y) and see an improvement.

qr_alg()
The command qr_alg(A) will execute a series of steps of the qr algorithm designed to drive the matrix **A** to a similar upper triangular matrix. Upper triangular matrices have their eigenvalues on the main diagonal. Two similar matrices have the same eigenvalues. Thus the qr algorithm is often used to obtain the approximate eigenvalues of a matrix. This is algorithm as coded here is very high level and will be slow for large matrices and it’s results should be verified.

qr_factor
The command qr_factor(A) will execute the computation of two factors of A, i.e. A = Q*R, where R is an upper triangular matrix and Q is an Unitary matrix.

Registration & License---
This product is shareware and you are licensed to freely distribute the files provided herewith provided you do so free-of-charge. You are licensed to use this product for evaluation of it's usefulness. After 60 days of trial you should either discontinue use or pay $25.00 for a registered copy. Registration entitles you to a full feature copy of this application. The registered copy will be free of any feature restricting code which may have limited the print, file-save and matrix size capability. The full version has more functions, more features, larger capacity, elementary programming techniques and offers the next upgrade and minor custom changes for just the shipping and handling charge.

Two order forms follow:

Print Now ? *If so select the appropriate form, hi-lite the form body, & use the print command on the menu*.

```
----------------------------------------------------------------------
|              Wye Software / Pik A Program Order Form                 |
----------------------------------------------------------------------
                    THANK YOU FOR REGISTERING YOUR SHAREWARE!
```

Important: If you are ordering by mail, e-mail, or fax, please completely
fill out this form and send it in. If you are sending a check or money
order, please make sure that payment is made in US funds drawn on a US bank.

Technical support is not available from Pik A Program. For any technical
help, please contact Wye Software directly online:
MicroSoft Network (msn) GO WYESOFT
Internet: Merritt_S@msn.com

For registered users of any program which Pik A Program distributes, who do
not already have access to Compuserve, we can supply a FREE Compuserve
membership with a $15.00 usage credit and access software. This is much more
than enough online time to have your technical support questions answered,
and still have some fun!

You may mail, fax, e-mail, or phone in your order. Please send the completed
registration form, along with payment to Pik A Program, Inc.

```
----------------------------------------------------------------------
|              Wye Software / Pik A Program ORDER FORM                 |
----------------------------------------------------------------------
  Pik A Program (tm)           TOLL FREE (ORDERS ONLY) 1-800-TOREGISTER
  13 Saint Marks Place                                  (867-3447)
  NY, NY   10003                 Telephone (212) 598-4939
  USA                            Fax       (212) 228-5879
                               Compuserve: 74777,3233
                               Internet: 74777.3233@compuserve.com
```

____
(Name)

____
(Company)

____
(Street1)

____
(Street2)

____
(Town) (State) (Zip)

____
(Telephone) (Fax) (Country if outside USA)

____
(e-mail address)

Shipping $4.00 for your entire order, whether you order 1 or 100 items.

QUANTITY	ITEM	PRICE EACH	TOTAL
	MtrxCalc V2.20	US $25.00	US$
	Shipping	US $ 4.00	US$ 4.00
	TAX (NYS residents only)	___ %	US$
		TOTAL:	US$

================

Orders are sent on 3.5" diskettes unless otherwise requested. Prices good through 12/96, subject to change thereafter.
I am paying by:
( ) CHECK ( ) VISACARD ( ) MASTERCARD ( ) AMERICAN EXPRESS
( ) CASH (by registered mail only, please) ( ) DISCOVER CARD

Number ______-______-______-______ Exp. date ___-___

____ ______-______-______
(Signature) (Date)

We would appreciate it if you would tell us where did you first heard about this program. Thanks for your cooperation.
[] BBS/Online service - which one?____________________________________
[] Internet - which site?____________________________________
[] Friend [] retail store package [] CD-ROM ______________________
[] magazine - which one?_________________________________
[] other - ____

Ordering DIRECTLY from Wye Software

NOTE! Please use this form if you wish to have your registered copy branded with your name.

MtrxCalc Registration Order Form

Payments must be in US dollars drawn on a US bank, or you can send international postal money orders in US dollars.
Send the payment and this order form to:

Wye Software, 19 Vista Drive,
Little Rock, AR 72210-1714.

Name: ________________________________ Date: _______________

Name as branded ________________________________

Company: ________________________________(Optional)

Street Address ________________________________

____

City ________________________________

State ________________________________ Zip _______________

Country: ________________________________

Single Copy	$25.00 each	$___________
S & H	$ 5.00(USA)	$___________
or	$ 7.00(Elsewhere)	$___________
Total payment		$___________

Version(Select one) WindowsNT____ Windows95____

Electronic Mail address: ________________________________(Optional)

Matrix Calculator Help Contents

The Matrix Calculator Version 1 was described as a ‘calculator’ interface to a numerical linear algebra library. This new version does not depend on the calculator button simulation and can be operated without that interface entirely (however, that interface is preserved as an option for it’s convenience and ease). Any method applied by depressing a button can also be applied by keyboard. For example pushing the button labeled ‘Partial pivot’ will apply that method of solution to the system U*X=R; while entering p_pivot(U,R) from the keyboard will do the same. In addition, using a button ‘teaches’ the use of the keyboard. That is, in the example above the script ‘p_pivot(U,R)’ will be displayed when that button is pushed. Even when used, much less screen space is required now for the button interface since several related methods can be done by choosing the appropriate button face for the one button that applies all those related methods. If one does not like the calculator button interface it can be closed or minimized and the data entry window can be maximized. See Getting Started for a detailed image of these windows.

Mtrxcalc provides a sophisticated parser/interpreter for the evaluation of expressions such as A*B or A*(B+C) or inv(A*trans(A))*X and etc.. A diary of the latest twenty operations is kept and any item from that list can be re-applied at any time, edited or as is. The matrices keep a title of their origins and can be saved to file or printed.

MtrxCalc now reads and saves in the ‘CSV’ format and other text file formats so that the output can be read by other applications. See the File Save As… option.

THIS APPLICATION IS PRESENTED AS IS AND WITHOUT WARRANTY, EXPLICIT OR IMPLIED. It is *intended to make day-to-day use of matrix algebra easier*.

Help is available on the following topics

Menu Commands

Edit menu commands

The Edit menu offers the following commands:

Cut	Deletes data from the document and moves it to the clipboard.
Copy	Copies data from the document to the clipboard.
Paste	Pastes data from the clipboard into the document.
Paste Link	Pastes from the clipboard a link to data in another application.
Insert New Object	Inserts and embeds an object, such as a chart or an equation in a document.
Links	List and edit links to embedded documents.

View menu commands

The View menu offers the following commands:

Toolbar	Shows or hides the toolbar.
Status Bar	Shows or hides the status bar.

Window menu commands

The Window menu offers the following commands, which enable you to arrange multiple views of multiple documents in the application window:

New Window	Creates a new window that views the same document.
Cascade	Arranges windows in an overlapped fashion.
Tile	Arranges windows in non-overlapped tiles.
Arrange Icons	Arranges icons of closed windows.
Window 1, 2, ...	Goes to specified window.

Help menu commands

The Help menu offers the following commands, which provide you assistance with this application:

Index	Offers you an index to topics on which you can get help.
Using Help	Provides general instructions on using help.
About	Displays the version number of this application.

New command (File menu)

Use this command to open a new empty document in MtrxCalc. You may use the SAVE command to write and save your matrix files. The default extensions is TXT

Shortcuts

Toolbar:

Keys: CTRL+N

File New dialog box

Open a new document with it’s accompanying window pair:
At present only file type *.MTX is supported here

Open command (File menu)

Use this command to open an existing file/document in a new window.   You can open multiple documents at once, but you must remember that each document has a pair of views.   The upper view of buttons and the lower view of data.   The buttons of one document will not operate on data of another.   Use the Window menu to switch among the multiple open documents.   See Window 1, 2, ... command.

Shortcuts

Toolbar:
Keys: CTRL+O

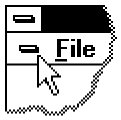
Import Data File

Use this command to import data from a file created in Comma Separated Value (CSV) format, Tab Separated Value (TSV) and other text file formats. One may construct such a file with most any text editor. You may also use Microsoft's Excel to create a CSV file. Make certain that the CSV file contains only the matrix elements separated by commas, and written first row to last. Save a matrix file using MtrxCalc and use Notepad or some other text editor to view the format.

Close command (File menu)

Use this command to close all windows containing the active document.   MTRXCALC **does not** suggest that you save changes to your document before you close it, since in most cases of for seen use, that is not desired.   If you do wish to save a matrix result be sure the result is in the display **Dlg** and use the Save As dialog box

You can also close a document by using the Close icon on the document's window, as shown below:



Save command (File menu)

In M

In MtrxCalc your must use the ‘Save ...’ command to save your files. Enter a name, say, *filename* into the dialog control and depress the ***save** button.* This action will save the matrix in the current display of MtrxCalc into two files, *filename.mtx* and *filename.txt.* The filename.txt file can be read by many other applications, including Microsoft’s Excel and Notepad. **Please note this is the save button under the file menu not the Save button in the <display>**

If you need Tab Separated Values or Semi-Colon (Seperated) Values , type <TSV> or <SCV> in the Evaluator in the MtrxCalc <display> and hit enter before saving the file.

Save As command (File menu)

Use this command to save and name the active document. MtrxCalc displays the Save As dialog box so you can name your document files ???????.MTX and ????????.TXT. Please note that the files saved in the *.MTX extension are in a binary format that is meaningless to other applications, while those saved in the *.txt extension are text files with the Comma Separated Values format. The latter files can be read by many other applications including Microsoft Excel and Matlab.

If you need Tab Separated Values or Semi-Colon (Seperated) Values , type <TSV> or <SCV> in the Evaluator in the MtrxCalc <display> and hit enter before saving the file.

File Save As dialog box

The following options allow you to specify the name and location of the file you're about to save:

File Name

Type a new filename to save a document with a different name.   A filename in the Win32S subsystem can contain up to eight characters and an extension of up to three characters. In Windows NT or Windows 95 long file names (up to 256 charaters) are possible.   MtrxCalc adds the extension .MTX.

Select the drive in which you want to store the document.

Directories

Select the directory in which you want to store the document.

Network...

Choose this button to connect to a network location, assigning it a new drive letter.

1, 2, 3, 4 command (File menu)

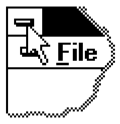
Use the numbers and filenames listed at the bottom of the File menu to open the last four documents you closed.   Choose the number that corresponds with the document you want to open.

Exit command (File menu)

Use this command to end your MTRXCALC session.   You can also use the Close command on the application Control menu. MTRXCALC **does not prompt** you to save documents with unsaved changes.

Shortcuts

Mouse: Double-click the application's Control menu button.



Keys: ALT+F4

Cut command (Edit menu)

Use this command to remove the currently selected data from the document and put it on the clipboard.   This command is unavailable if there is no data currently selected.

Cutting data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar:
Keys: CTRL+X

Copy command (Edit menu)

Use this command to copy selected data onto the clipboard.   This command is unavailable if there is no data currently selected.

Copying data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar:

Keys: CTRL+C

Paste command (Edit menu)

Use this command to insert a copy of the clipboard contents at the insertion point.   This command is unavailable if the clipboard is empty.

Shortcuts

Toolbar:

Keys: CTRL+V

Toolbar command (View menu)

Use this command to display and hide the Toolbar, which includes buttons for some of the most common commands in MTRXCALC, such as File Open.   A check mark appears next to the menu item when the Toolbar is displayed.

See Toolbar for help on using the toolbar.

Toolbar

🗋 📂 💾

To hide or display the Toolbar, choose Toolbar from the View menu (ALT, V, T).

Click	**To**
🗋	Open a new document.
🗋	Open an existing document. MTRXCALC displays the Open dialog box, in which you can locate and open the desired file.
💾	Save the active document or template with its current name. If you have not named the document, MTRXCALC displays the Save As dialog box.
🖨	Print the active document.
🗋	Remove selected data from the document and stores it on the clipboard.
🗋	Copy the selection to the clipboard.
🗋	Insert the contents of the clipboard at the insertion point.
▨	Reverse the last editing. Note: You cannot undo some actions.
⏮	Go to the first record in the current selection.
◀	Go to the previous record in the current selection.
▶	Go to the next record in the current selection.
⏭	Go to the last record in the current selection.

Status Bar command (View menu)

Use this command to display and hide the Status Bar, which describes the action to be executed by the selected menu item or depressed toolbar button, and keyboard latch state. A check mark appears next to the menu item when the Status Bar is displayed.

See Status Bar for help on using the status bar.

Status Bar

The status bar is displayed at the bottom of the MTRXCALC window.   To display or hide the status bar, use the Status Bar command in the View menu.

The left area of the status bar describes actions of menu items as you use the arrow keys to navigate through menus.   This area similarly shows messages that describe the actions of toolbar buttons as you depress them, before releasing them.   If after viewing the description of the toolbar button command you wish not to execute the command, then release the mouse button while the pointer is off the toolbar button.

The right areas of the status bar indicate which of the following keys are latched down:

Indicator	**Description**
CAP	The Caps Lock key is latched down.
NUM	The Num Lock key is latched down.
SCRL	The Scroll Lock key is latched down.

New command (Window menu)

Use this command to open a new window with the same contents as the active window. You can open multiple document windows to display different parts or views of a document at the same time. If you change the contents in one window, all other windows containing the same document reflect those changes. When you open a new window, it becomes the active window and is displayed on top of all other open windows.

Cascade command (Window menu)

Use this command to arrange multiple opened windows in an overlapped fashion.

Tile command (Window menu)

Use this command to arrange multiple opened windows in a non-overlapped fashion.

Tile Horizontal command (Window menu)

Use this command to vertically arrange multiple opened windows in a non-overlapped fashion.

Tile Vertical command (Window menu)

Use this command to arrange multiple opened windows side by side.

Window Arrange Icons Command

Use this command to arrange the icons for minimized windows at the bottom of the main window.   If there is an open document window at the bottom of the main window, then some or all of the icons may not be visible because they will be underneath this document window.

Split Command (Window menu)

Use this command to split the active window into panes.   You may then use the mouse or the keyboard arrows to move the splitter bars.   When you are finished, press the mouse button or return to leave the splitter bars in their new location.   Pressing escape keeps the splitter bars in their original location.

1, 2, ... command (Window menu)

MTRXCALC displays a list of currently open document windows at the bottom of the Window menu.   A check mark appears in front of the document name of the active window.   Choose a document from this list to make its window active.

Index command (Help menu)

Use this command to display the opening screen of Help.   From the opening screen, you can jump to step-by-step instructions for using MTRXCALC and various types of reference information.

Once you open Help, you can click the Contents button whenever you want to return to the opening screen.

Using Help command (Help menu)

Use this command for instructions about using Help.

About command (Help menu)

Use this command to display the copyright notice and version number of your copy of MTRXCALC.

Context Help command

Use the Context Help command to obtain help on some portion of MTRXCALC.   When you choose the Toolbar's Context Help button, the mouse pointer will change to an arrow and question mark.   Then click somewhere in the MTRXCALC window, such as another Toolbar button.   The Help topic will be shown for the item you clicked.

Shortcut

Keys:	SHIFT+F1

Title Bar

<< Show your application's title bar here. >>

The title bar is located along the top of a window. It contains the name of the application and document.

To move the window, drag the title bar. Note: You can also move dialog boxes by dragging their title bars.

A title bar may contain the following elements:

- Application Control-menu button
- Document Control-menu button
- Maximize button
- Minimize button
- Name of the application
- Name of the document
- Restore button

Scroll bars

Displayed at the right and bottom edges of the document window.   The scroll boxes inside the scroll bars indicate your vertical and horizontal location in the document.   You can use the mouse to scroll to other parts of the document.

<< Describe the actions of the various parts of the scrollbar, according to how they behave in your application. >>

Size command (System menu)

Use this command to display a four-headed arrow so you can size the active window with the arrow keys.

After the pointer changes to the four-headed arrow:

1. Press one of the DIRECTION keys (left, right, up, or down arrow key) to move the pointer to the border you want to move.
2. Press a DIRECTION key to move the border.
3. Press ENTER when the window is the size you want.

Note: This command is unavailable if you maximize the window.

Shortcut

Mouse: Drag the size bars at the corners or edges of the window.

Move command (Control menu)

Use this command to display a four-headed arrow so you can move the active window or dialog box with the arrow keys.

Note: This command is unavailable if you maximize the window.

Shortcut

Keys:	CTRL+F7

Minimize command (application Control menu)

Use this command to reduce the <<YourApp>> window to an icon.

Shortcut

Mouse: Click the minimize icon on the title bar.
Keys: ALT+F9

Maximize command (System menu)

Use this command to enlarge the active window to fill the available space.

Shortcut

Mouse: Click the maximize icon ▲ on the title bar; or double-click the title bar.
Keys: CTRL+F10 enlarges a document window.

Next Window command (document Control menu)

Use this command to switch to the next open document window. <<YourApp>> determines which window is next according to the order in which you opened the windows.

Shortcut

Keys:	CTRL+F6

Previous Window command (document Control menu)

Use this command to switch to the previous open document window. <<YourApp>> determines which window is previous according to the order in which you opened the windows.

Shortcut

Keys:	SHIFT+CTRL+F6

Close command (Control menus)

Use this command to close the active window or dialog box.

Double-clicking a Control-menu box is the same as choosing the Close command.

Note: If you have multiple windows open for a single document, the Close command on the document Control menu closes only one window at a time.   You can close all windows at once with the Close command on the File menu.

Shortcuts

Keys:	CTRL+F4 closes a document window
	ALT+F4 closes the <<YourType>> window or dialog box

Restore command (Control menu)

Use this command to return the active window to its size and position before you chose the Maximize or Minimize command.

Switch to command (application Control menu)

Use this command to display a list of all open applications. Use this "Task List" to switch to or close an application on the list.

Shortcut

Keys: CTRL+ESC

Dialog Box Options

When you choose the Switch To command, you will be presented with a dialog box with the following options:

Task List

Select the application you want to switch to or close.

Switch To

Makes the selected application active.

End Task

Closes the selected application.

Cancel

Closes the Task List box.

Cascade

Arranges open applications so they overlap and you can see each title bar. This option does not affect applications reduced to icons.

Tile

Arranges open applications into windows that do not overlap. This option does not affect applications reduced to icons.

Arrange Icons

Arranges the icons of all minimized applications across the bottom of the screen.

Choose Font dialog box

<< Write application-specific help here. >>

Choose Color dialog box

<< Write application-specific help here. >>

Clear command (Edit menu)

<< Write application-specific help here. >>

Clear All command (Edit menu)

<< Write application-specific help here. >>

Next Pane

<< Write application-specific help here. >>

Prev Pane

<< Write application-specific help here. >>

Modifying the Document

<< Write application-specific help here that provides an overview of how the user should modify a document using your application.

If your application supports multiple document types and you want to have a distinct help topic for each, then use the help context i.d. generated by running the MAKEHELP.BAT file produced by AppWizard.   Alternatively, run MAKEHM as follows:

```
makehm IDR_HIDR_,0x2000 resource.h
```

If the IDR_ symbol for one of your document types is, for example, IDR_CHARTTYPE, then the help context i.d. generated by MAKEHM will be HIDR_CHARTTYPE.

Note, AppWizard defines the HIDR_DOC1TYPE help context i.d. used by this help topic for the first document type supported by your application.   AppWizard produces an alias in the .HPJ file for your application, mapping HIDR_DOC1TYPE to the HIDR_ produced by MAKEHM for that document type. >>

No Help Available

No help is available for this area of the window.

No Help Available

No help is available for this message box.

<< If you wish to author help specific to each message box prompt, then remove the AFX_HIDP_xxx values from the [ALIAS] section of your .HPJ file, and author a topic for each AFX_HIDP_xxx value.   For example, AFX_HIDP_INVALID_FILENAME is the help topic for the Invalid Filename message box. >>

File_Menu_Commands

The File menu offers the following commands:

New	Creates a new document.
Open	Opens an existing document.
Import	Imports an existing data file.
Close	Closes an opened document.
Save As	Saves an opened document to a specified file name.
Print	Prints a document.
Print Preview	Displays the document on the screen as it would appear printed.
Print Setup	Selects a printer and printer connection.
Exit	Exits MTRXCALC

Print command (File menu)

Use this command to print the current matrix in **Dlg**.   If you wish to print a matrix stored in A, B or etc, first transfer it to Dlg.   This command presents a Print dialog box, where you may specify the number of copies, the destination printer, and other printer setup options.   Please NOTE this application prints in Win NT but does not print on all versions for WIN32s

Shortcuts

Toolbar:

Keys: CTRL+P

Print dialog box

The following options allow you to specify how the document should be printed:

Printer
This is the active printer and printer connection. Choose the Setup option to change the printer and printer connection.

Setup
Displays a Print Setup dialog box, so you can select a printer and printer connection.

Print Range
Specify the pages you want to print:

All	Prints the entire document.
Selection	Prints the currently selected text.
Pages	Prints the range of pages you specify in the From and To boxes.

Copies
Specify the number of copies you want to print for the above page range.

Collate Copies
Prints copies in page number order, instead of separated multiple copies of each page.

Print Quality
Select the quality of the printing. Generally, lower quality printing takes less time to produce.

Print Progress Dialog

The Printing dialog box is shown during the time that <<YourApp>> is sending output to the printer.   The page number indicates the progress of the printing.

To abort printing, choose Cancel.

Print Preview command (File menu)

Use this command to display the active document as it would appear when printed.   When you choose this command, the main window will be replaced with a print preview window in which one or two pages will be displayed in their printed format.   The print preview toolbar offers you options to view either one or two pages at a time; move back and forth through the document; zoom in and out of pages; and initiate a print job.

Print Preview toolbar

The print preview toolbar offers you the following options:

Print
Bring up the print dialog box, to start a print job.

Next Page
Preview the next printed page.

Prev Page
Preview the previous printed page.

One Page / Two Page
Preview one or two printed pages at a time.

Zoom In
Take a closer look at the printed page.

Zoom Out
Take a larger look at the printed page.

Close
Return from print preview to the editing window.

Print Setup command (File menu)

Use this command to select a printer and a printer connection.   This command presents a Print Setup dialog box, where you specify the printer and its connection.

Print Setup dialog box

The following options allow you to select the destination printer and its connection.

Printer
Select the printer you want to use. Choose the Default Printer; or choose the Specific Printer option and select one of the current installed printers shown in the box. You install printers and configure ports using the Windows Control Panel.

Orientation
Choose Portrait or Landscape.

Paper Size
Select the size of paper that the document is to be printed on.

Paper Source
Some printers offer multiple trays for different paper sources. Specify the tray here.

Options
Displays a dialog box where you can make additional choices about printing, specific to the type of printer you have selected.

Network...
Choose this button to connect to a network location, assigning it a new drive letter.

Page Setup command (File menu)

<< Write application-specific help here. >>