M_Upper is an upper triangular (or trapezoidal) matrix.

W_PIV is 1D wave containing pivot indices.
See code example below for implementation details.

If /FM is omitted the output of the operation consists of five 1D waves:

W_Diagonal is the main diagonal of matrixU.

W_UDiagonal is the first upper diagonal of M_Upper.

W_U2Diagonal is the second diagonal of M_Upper.

W_LDiagonal is the first lower diagonal of M_Lower.

W_PIV is a vector of pivot indices.

In this case M_Lower can be constructed (see below) from W_LDiagonal and the pivot index wave W_PIV.

If you are working with tridiagonal matrices you can take advantage of MatrixOp functionality to reconstruct your outputs. For example:

```
MatrixOp/O M_Upper=Diagonal(W_diagonal)
MatrixOp/O M_Upper=setOffDiag(M_Upper,1,W_UDiagonal)
MatrixOp/O M_Upper=setOffDiag(M_Upper,2,W_U2Diagonal)
```

These commands can be combined into a single command line.

The construction of M_Lower is a bit more complicated and can be accomplished for real data using the following code:

```
Function MakeLTMatrix(W_diagonal,W_LDiagonal,W_PIV)
    Wave W_diagonal,W_LDiagonal,W_PIV

    Variable i,N=DimSize(W_diagonal,0)
    MatrixOp/O M_Lower=setOffDiag(ZeroMat(N,N,4),-1,W_LDiagonal)
    M_Lower=p==q ? 1:M_Lower[p][q]       // Set the main diagonal to 1's
    MatrixOp/O index=W_PIV-1             // Convert from 1-based array
    for(i=1;i<=N-2;i+=1)
        if(index[i]!=i)
            variable j,tmp
            for(j=0;j<=i-1;j+=1)
                tmp=M_Lower[i][j]
                M_Lower[i][j]=M_Lower[i+1][j]
                M_Lower[i+1][j]=tmp
            endfor
        endif
    endfor
End
```

This code is provided for illustration only. In practice you could use the /FM flag so that the operation creates the full lower and upper matrices for you.

The variable V_flag is set to zero if the operation succeeds and to 1 otherwise (e.g., if the input is singular). The variables V_Sum and V_min are also set by some of the flag options above.

See Also

MatrixLUD, **MatrixOp**, **Matrix Math Operations** on page III-138 for more about Igor's matrix routines.

MatrixMultiply

MatrixMultiply *matrixA* [**/T**], ***matrixB*** [**/T**] [, ***additional matrices***]

The MatrixMultiply operation calculates matrix expression *matrixA*matrixB* and puts the result in a matrix wave named M_product generated in the current data folder. The /T flag can be included to indicate that the transpose of the specified matrix should be used.

If any of the source matrices are complex, then the result is complex.

Parameters

If *matrixA* is an NxP matrix then *matrixB* must be a PxM matrix and the product is an NxM matrix. Up to 10 matrices can be specified although it is unlikely you will need more than three. The inner dimensions must be the same. Multiplication is performed from right to left.

It is legal for M_product to be one of the input matrices. Thus `MatrixMultiply A,B,C` could also be done as: