EXPLORATIONS

CHAPTER 7

Matrices and Vectors

In this chapter we will consider a variety of problems from the areas of matrix theory and linear algebra. One of the first things to be encountered will be the MATRX editor features of the TI-86, which are significantly different from those of the TI-85. This chapter relies heavily on matrix and vector information, which can be found in the *TI-86 Guidebook*. In fact, the reader of Chapter 7 may need to consult the *Guidebook* more frequently than in previous chapters.

Preliminaries

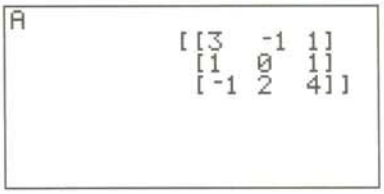
Since we will be creating several matrices in this chapter, it may be worthwhile to delete all existing matrices from your calculator in order to name the matrices created in this chapter the same as we have named them. Recall that to delete existing matrices, you press [2nd] [MEM] and select ⟨**DELET**⟩. Then press [MORE], select ⟨**MATRX**⟩, and then press [ENTER] as many times as is necessary to delete the existing matrices from your calculator.

§1 – Introduction to the Matrix Editor

1. Press [2nd] [MATRX] and select ⟨**EDIT**⟩ to access the MATRX editor. Enter **A** as the matrix name, and press [ENTER]. We will create a 3×3 matrix, so press [3] [ENTER], then press [3] [ENTER] again. The TI-86 now prompts us for the entries of the matrix as in (7.1.1). Note that the matrix is displayed in the form of an array and that initially all entries of the matrix **A** are zero.

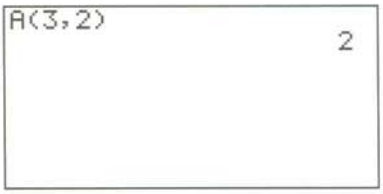
(7.1.1)

2. Press [3] to make the 1,1 entry equal to 3. Press [ENTER]. In similar manner, make the other entries in row one –1 and 1. Make row two 1, 0, 1 and row three –1, 2, 4. Press [EXIT] to return to the home screen. Type **A** and press [ENTER] to see (7.1.2).



(7.1.2)

3. Note that the beginning and end of the matrix are indicated by double square brackets, whereas individual rows are indicated by single square brackets. Each row of a matrix is really in the form of a vector. We can access a particular entry of **A** by typing **A**(row #, column #) and pressing [ENTER] as in (7.1.3).



(7.1.3)

4. We can change this entry from the home screen using the STO▸ key as in (7.1.4).

```
A(3,2)
                              2
3→A(3,2)
                              3
```

(7.1.4)

5. Type **A** and press ENTER to see the change in the 3, 2 entry of **A** as in (7.1.5).

```
A
                [[3   -1  1]
                 [1    0  1]
                 [-1   3  4]]
```

(7.1.5)

§2 – Creating a Matrix from the Home Screen

Matrices can also be created from the home screen as in (7.2.1). We create a 2×2 matrix by entering the matrix as indicated in (7.2.1). Note again that the beginning and end of the matrix are indicated by double square brackets, whereas individual rows are indicated by single square brackets. Note when entering a matrix (or a vector) from the home screen, it is necessary to put commas between entries but not between rows of a matrix. In (7.2.1) we also store the matrix in the variable **B** by pressing STO▸, then typing in **B**, and pressing ENTER.

```
[[3,1][4,5]]
                          [[3 1]
                           [4 5]]
Ans→B
                          [[3 1]
                           [4 5]]
```

(7.2.1)

§3 – Matrix-Vector Multiplication

We have noted that the TI-86 uses double square brackets to indicate a matrix. Single square brackets indicate a vector.

1. Figure (7.3.1) shows that the TI-86 can form the product of a matrix and a vector, giving a vector as a result. Note that the calculator knows to transpose the vector in order to do this product.

```
A*[1,5,4]
                          [2 5 30]
```

(7.3.1)

2. The equivalent computations using matrix-matrix multiplication would require the transpose operation found in the MATH submenu of the MATRX menu. Figure (7.3.2) shows that the 3×3 matrix **A** times the 3×1 matrix $[[1\ \ 5\ \ 4]]^T$ will produce the appropriate 3×1 matrix.

```
A*[[1,5,4]]ᵀ
                          [[2 ]
                           [5 ]
                           [30]]
NAMES EDIT MATH OPS CPLX
det   ᵀ    norm ei3V1 ei3Vc▸
```

(7.3.2)

§4 – Submatrices and Matrix Arithmetic

1. Of course, the product **B**∗**A** is undefined since **B** is 2×2 and **A** is 3×3. But the product of **B** and any 2×2 submatrix of **A** can be formed as in (7.4.1), (7.4.2), and (7.4.3). The first two arguments of the statement **A**(#, #, #, #) specify the upper left entry of the desired submatrix, and the second two arguments specify the lower right entry of the submatrix. Consult the *TI-86 Guidebook* for further details on accessing submatrices of a matrix.

```
B*A(1,1,2,2)
              [[10  -3]
               [17  -4]]
```
(7.4.1)

```
B*A(2,2,3,3)
              [[3   7 ]
               [15 24]]
```
(7.4.2)

```
B*A(1,2,2,3)
              [[-3  4]
               [-4  9]]
```
(7.4.3)

2. With the TI-86 we can also redimension a matrix. If rows or columns are added, then the new entries will have value zero. We make **B** a 3×3 matrix by executing the command shown in (7.4.4). The **dim** command is found in the OPS submenu of the MATRX menu, and the set brackets { } are found in the LIST menu.

```
{3,3}→dim B
                        {3 3}
NAMES EDIT MATH OPS CPLX
 dim  Fill ident ref rref ▸
```
(7.4.4)

3. Also check to see that **B** is now a 3×3 matrix with zeros filling in the new row and column as in (7.4.5).

```
B
              [[3 1 0]
               [4 5 0]
               [0 0 0]]
```
(7.4.5)

4. Now the product **B**∗**A** is defined and produces a 3×3 matrix as in (7.4.6).

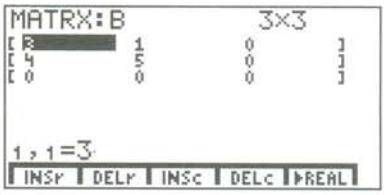
```
B*A
            [[10  -3  4]
             [17  -4  9]
             [0   0   0]]
```
(7.4.6)

5. Similarly, in (7.4.7) we see the computation of $\mathbf{A} + 5\mathbf{B}$.

```
A+5*B
            [[18  4   1]
             [21  25  1]
             [-1  3   4]]
```
(7.4.7)

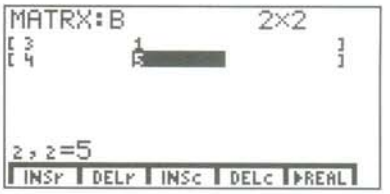
§5 – Editing Using the Matrix Editor

1. We have edited the matrices **A** and **B** from the home screen, but it is usually easier to use the MATRX editor to create and/or edit matrices on the TI-86. For example, press [2nd] [MATRX] ⟨**EDIT**⟩. Now select **B** from the menu as the matrix to be edited and press [ENTER]. Press the down arrow twice so that we get (7.5.1).



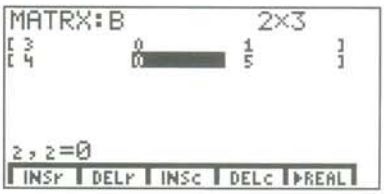
(7.5.1)

2. We will first convert **B** back to a 2×2 matrix by deleting row 3 and column 3. Press the right arrow twice to move to column 3. Now select ⟨**DELc**⟩ to delete the column the cursor is currently on. Press the down arrow twice, then select ⟨**DELr**⟩. Note that **DELr** deleted the row the cursor was on, and **B** is now a 2×2 matrix as in (7.5.2).



(7.5.2)

3. We will make **B** into a 2×3 matrix by inserting a column between the two columns currently in **B**. Select ⟨**INSc**⟩. Note that **INSc** inserted a new column to the left of the column the cursor was on, and **B** is now a 2×3 matrix with zeros in the new second column as in (7.5.3).

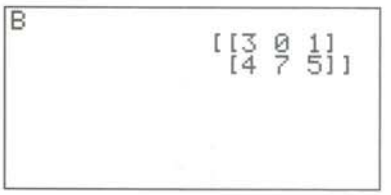


(7.5.3)

4. In (7.5.4) we are preparing to change the (2, 2) entry of this new matrix to 7. Pressing [ENTER] will record the change.

(7.5.4)

5. Finally, press [EXIT] to return to the home screen and verify that **B** is indeed a 2×3 matrix as in (7.5.5).

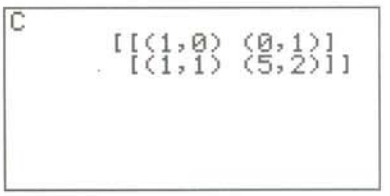


(7.5.5)

§6 – Matrices with Complex Entries

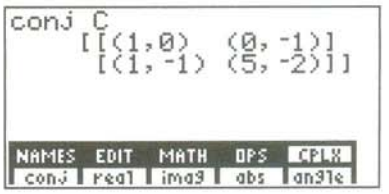
1. The TI-86 can do arithmetic using complex numbers, and it can handle matrices with complex entries. Access the MATRX editor and create the 2×2 matrix **C** given by

$$\mathbf{C} = \begin{bmatrix} 1 & i \\ 1+i & 5+2i \end{bmatrix}. \qquad (7.6.1)$$



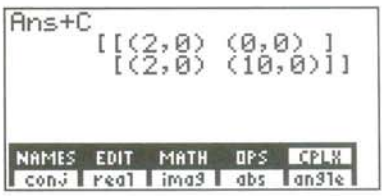
Recall that the TI-86 form of the complex number $a+bi$ is (a,b). Exit to the home screen and check that **C** is as in (7.6.1).

2. Using the CPLX submenu of the MATRX menu, form the conjugate of **C** as in (7.6.2).



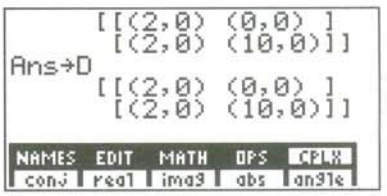
(7.6.2)

3. Add **C** to this result by pressing [+], typing **C**, and pressing [ENTER] to obtain (7.6.3). Note that the result is a 2×2 matrix with complex entries, even though the entries are real since they all have zero imaginary parts.



(7.6.3)

4. Store this result in **D** as indicated in (7.6.4). Now we will convert **D** to a real matrix by editing **D**. Access the MATRX editor and select **D** as the matrix to edit.

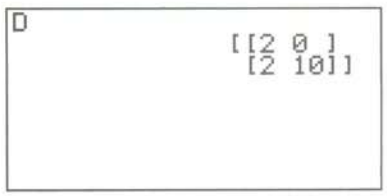


(7.6.4)

5. Press the down arrow twice to obtain (7.6.5).

(7.6.5)

6. Now select ⟨▸**REAL**⟩. Note that **D** becomes a 2×2 matrix with real entries. Exit to the home screen to verify this as in (7.6.6).



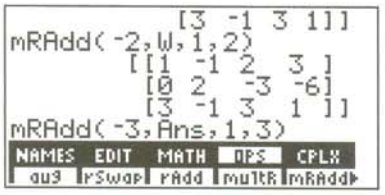
(7.6.6)

§7 – Using Elementary Row Operations

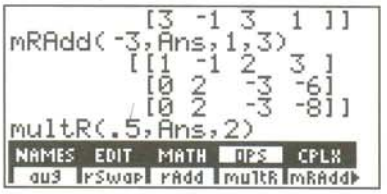
The OPS submenu of the MATRX menu contains the commands which are used to do elementary row operations on a matrix. In particular, we will use these commands to convert the matrix

$$\mathbf{W} = \begin{bmatrix} 1 & -1 & 2 & 3 \\ 2 & 0 & 1 & 0 \\ 3 & -1 & 3 & 1 \end{bmatrix}$$

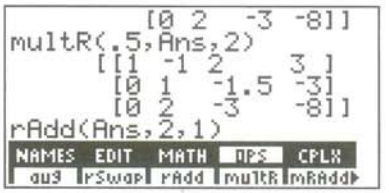
to row-reduced echelon form. First create **W**, either from the home screen or from the MATRX editor. Figures (7.7.1) through (7.7.7) show the steps used in putting **W** in row-reduced echelon form from the home screen. Each operation is in the OPS submenu. Working through these row operations one step at a time is a worthwhile exercise.



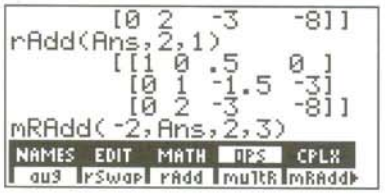
(7.7.1)



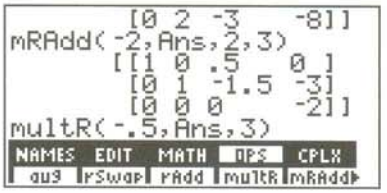
(7.7.2)



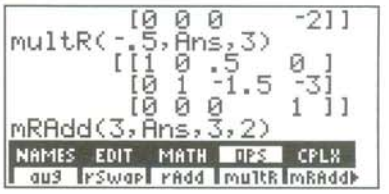
(7.7.3)



(7.7.4)



(7.7.5)



(7.7.6)

The syntax for the three row operations used here is:

mRAdd(*multiplier, matrix, row to be multiplied by multiplier, add to row*)

multR(*multiplier, matrix, row to be multiplied by multiplier*)

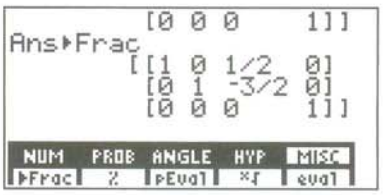
rAdd(*matrix, row to add, add to row*)

(7.7.7)

Note, however, that the process can be greatly shortened by selecting ⟨**rref**⟩ as in (7.7.8).

(7.7.8)

We could also now convert the entries in this row-reduced echelon form matrix to fraction form by using ▸**Frac** from the MISC submenu of the MATH menu as in (7.7.9).



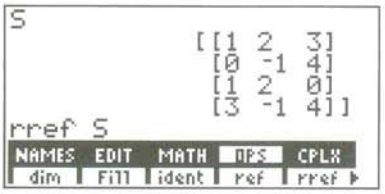
(7.7.9)

§8 – Using rref on a Matrix With More Rows Than Columns

1. Create the matrix

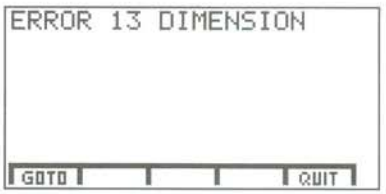
$$S=\begin{bmatrix}1 & 2 & 3\\ 0 & -1 & 4\\ 1 & 2 & 0\\ 3 & -1 & 4\end{bmatrix}.$$

If we try to use **rref** on **S** as in (7.8.1), we get the result given in (7.8.2).



(7.8.1)

2. Select ⟨**QUIT**⟩. The error is caused by the fact that the **rref** function requires that the number of rows in the matrix be less than or equal to the number of columns. We can get around this difficulty by adjoining a column of zeros to **S**, then using **rref**, and then deleting the last column of this result.

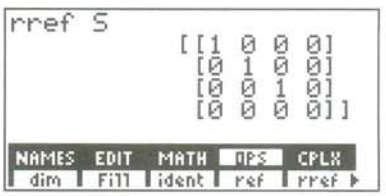


(7.8.2)

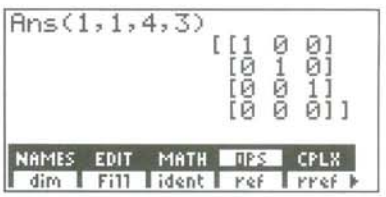
3. Figure (7.8.3) shows the command that we can use to redimension **S** using the **dim** command. Since a new column was created in **S**, it was automatically filled in with zeros by the TI-86.

(7.8.3)

4. Figures (7.8.4) and (7.8.5) show the reduction of the new **S** and the selection of the appropriate submatrix of the result. This last matrix is the row-reduced echelon form of the original matrix **S**.



(7.8.4)



(7.8.5)

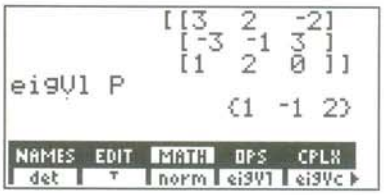
§9 – Finding Eigenvalues and Eigenvectors

1. The TI-86 will also find eigenvalues and eigenvectors for square matrices. Recall that the eigenvalues of a real matrix can be complex numbers. Create the matrix

$$\mathbf{P} = \begin{bmatrix} 3 & 2 & -2 \\ -3 & -1 & 3 \\ 1 & 2 & 0 \end{bmatrix}.$$

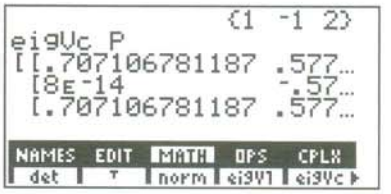
Then use **eigVl** found in the MATH submenu of the MATRX menu to find the eigenvalues of **P** as in (7.9.1).

(7.9.1)



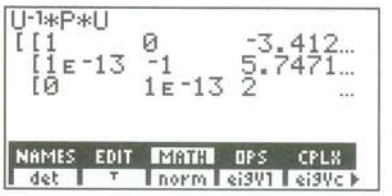
2. Note in this case, the 3×3 matrix **P** has three distinct eigenvalues. Now use **eigVc** to find some eigenvectors for **P**. See (7.9.2). Each column of the 3×3 matrix in (7.9.2) is supposed to be an eigenvector for **P**. Various important results from matrix theory and linear algebra insure that if we let **U** be the 3×3 matrix whose columns are eigenvectors for **P**, then $\mathbf{U}^{-1}\mathbf{PU}$ will be a diagonal matrix with the eigenvalues of **P** appearing as the diagonal entries.

(7.9.2)



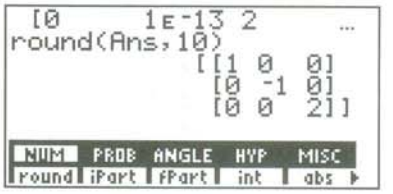
3. We will test this result for the matrix **P**. Working from the screen as in (7.9.2), press [STO▸] and type **U** and press [ENTER] to store the eigenvector matrix in **U**. Then compute $\mathbf{U}^{-1}*\mathbf{P}*\mathbf{U}$ and get the 3×3 matrix indicated in (7.9.3). The negative one exponent in (7.9.3) was obtained by pressing [2nd] [x^{-1}].

(7.9.3)



4. Use **round** in the NUM submenu of the MATH menu as in (7.9.4) to see that the nondiagonal entries are most likely exactly zero.

(7.9.4)



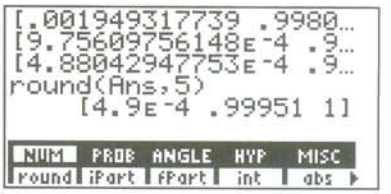
§10 – Using the Power Method

The Power Method is an algorithm which can be used to find the dominant eigenvalue and a corresponding eigenvector for a square matrix. We will use the matrix **P** of §9 and a small amount of the programming capabilities of the TI-86 to estimate the dominant eigenvalue for **P** using the Power Method. In general, under the proper conditions, if x_0 is a starting vector, then the sequence $\{a_n\} = \{P^n x_0\}$ will converge to a dominant eigenvector of **P**. Computationally, it is usually better to normalize the vector $P^n x_0$ at each step by dividing the vector by the magnitude of its largest component. We use the **max** and **vc▸li** features of the OPS submenu of the LIST menu to accomplish this normalization. In particular, **vc▸li** converts a vector to a list and **max** finds the largest member of a list.

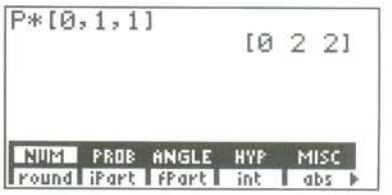
1. Starting with the home screen as in (7.10.1), we can just repeatedly press [ENTER] to see the sequence of vectors generated by the Power Method. Note that we are using $[1 \quad 2 \quad 3]$ as x_0.

(7.10.1)

2. After ten or so presses of [ENTER], we use **round** from the NUM submenu of the MATH menu as in (7.10.2). From this result it is reasonable to conjecture that $[0 \quad 1 \quad 1]$ is a dominant eigenvector for **P**, and computing $P*[0 \quad 1 \quad 1]$ as in (7.10.3) verifies that $[0 \quad 1 \quad 1]$ is an eigenvector belonging to the eigenvalue 2. We further conjecture that 2 is the dominant eigenvalue for **P**, a conjecture verified in (7.9.1) using other means.



(7.10.2)



(7.10.3)

We should note that the Power Method can fail. In particular, try repeating the steps in this section using $x_0 = [1 \quad 1 \quad 1]$. Theoretically, no convergence should occur for this choice of x_0, since in exact arithmetic, $\mathbf{P}*[1 \quad 1 \quad 1] = [3 \quad -1 \quad 3]$ and $\mathbf{P}*[3 \quad -1 \quad 3] = [1 \quad 1 \quad 1]$. However, round-off error turns out to be helpful here, since pressing [ENTER] many times (in fact, 50 or more times) with this choice of x_0 will still lead to the conjecture that $[0 \quad 1 \quad 1]$ is an eigenvector.

§11 – The Gram-Schmidt Orthogonalization Process

The Gram-Schmidt Orthogonalization process for finding an orthogonal basis is an important process in linear algebra and is linked to many important results in linear algebra and matrix theory. Recall that given a basis $v_1, v_2, \ldots, v_n$, we generate an orthogonal basis as follows:

$$w_1 = v_1, \text{ and for } k = 2, \ldots, n,$$

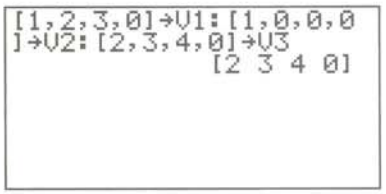
$$w_k = v_k - \left(\langle v_k, w_1\rangle / \langle w_1, w_1\rangle\right) w_1 - \cdots - \left(\langle v_k, w_{k-1}\rangle / \langle w_{k-1}, w_{k-1}\rangle\right) w_{k-1}$$

where $\langle x, y\rangle$ indicates the dot product (more generally, any inner product).

Consider then the three-dimensional subspace of $\mathbf{R}^4$ generated by the basis

$$v_1 = [1 \quad 2 \quad 3 \quad 0],\ v_2 = [1 \quad 0 \quad 0 \quad 0],\ v_3 = [2 \quad 3 \quad 4 \quad 0].$$

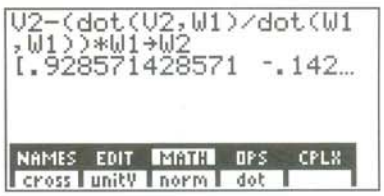
1. We will use the TI-86 to implement the Gram-Schmidt process to find an orthogonal basis w_1, w_2, w_3 for this same subspace of $\mathbf{R}^4$. Note that the dot product function is found in the MATH submenu of the VECTR menu. Figure (7.11.1) creates the first basis using the TI-86 variable names *V*1, *V*2, and *V*3.



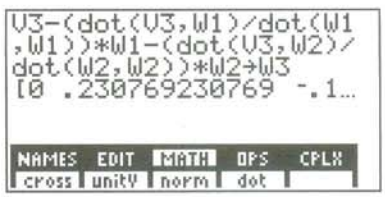
(7.11.1)

2. In (7.11.2) through (7.11.4) the Gram-Schmidt Process is used to generate an orthogonal basis with variable names *W*1, *W*2, and *W*3.

(7.11.2)

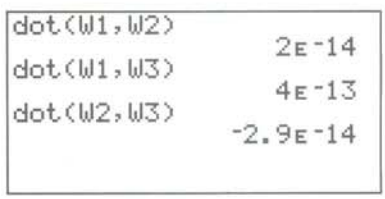


(7.11.3)



(7.11.4)

3. Figure (7.11.5) indicates that the new basis is, in fact, orthogonal. Also, in (7.11.3)–(7.11.5), we have made frequent use of the ability of the TI-86 to recall previous entries by pressing [2nd] [ENTRY] in order to do a little less typing on home screen entries, which are similar to one another.



(7.11.5)

Exercises

1. Create the following 3×4 matrix $\mathbf{A} = \begin{bmatrix} 2 & 1 & -1 & 1 \\ 1 & 0 & 1 & 1 \\ -2 & 1 & 0 & -3 \end{bmatrix}$ using the matrix editor.

   (a) Use elementary row operations to put **A** in row reduced echelon form.

   (b) Use **rref** to double check the result obtained in (a).

   (c) By trial and error, find a 2×2 submatrix **S** of **A** with the property that $S * \begin{bmatrix} 1 & -1 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 3 \end{bmatrix}$.

   (d) Delete the third row of **A** and find the row reduced echelon form of the resulting matrix.

2. Create the 3×3 matrix $\mathbf{B} = \begin{bmatrix} -1 & 1 & -1 \\ 2 & 3 & 0 \\ 5 & 1 & 2 \end{bmatrix}$ and verify that $\det(4\mathbf{B}) = 64 \det(\mathbf{B})$.

3. Diagonalize the matrix **B** from Exercise 2 as was done in §9. Compute **eigVc B** and conjecture as to what at least one "nice" eigenvector is for **B**.

4. Use the Power Method on the matrix $\mathbf{S} = \begin{bmatrix} 7 & 9 \\ 4 & 2 \end{bmatrix}$. Conclude that [1 4/9] is an eigenvector for **S**.

5. Find the row reduced echelon form for $\begin{bmatrix} 2 & -1 & 1 \\ 1 & 0 & 1 \\ 3 & 1 & 4 \\ 4 & 2 & 6 \end{bmatrix}$. Note that the TI-86 will not do this problem directly, so you should first consider inserting an extra column.

6. Verify that $\mathbf{W}^{-1}$ is equal to $(1/2)\mathbf{W}^2 - 2\mathbf{W} + (5/2)\mathbf{I}$, where **I** is the 3×3 identity matrix and $\mathbf{W} = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 2 & 0 \\ 0 & 2 & 1 \end{bmatrix}$.