1a) Please see the SOR method MATLAB code in the file named SOR.m. I input these variables:

```
A=[3 -1 1; 3 6 -2; 3 3 7];
b=[5;1;3];
x0=[0;0;0];
w=1.2;
tol=0.000001;
maxiter=100;
[x,iter] = SOR(A,b,x0,w,tol,maxiter)
```

And got back this output where x is the solution vector and iter is the maximum number of iterations it took the algorithm to converge:

```
>> HW5_script
x =
    1.469135979927580
    -0.555555695339397
    0.037037002444266
iter =
    21
```

b) I used these values for omega: 0.1, 0.5, 1.0, 1.5, 1.9 which produced the following output:

```
>> HW5_script

For omega = 0.1

x =

    1.469135213678013
    -0.555546191756907
    0.037037048619889

iter =
    137

For omega = 0.5

x =

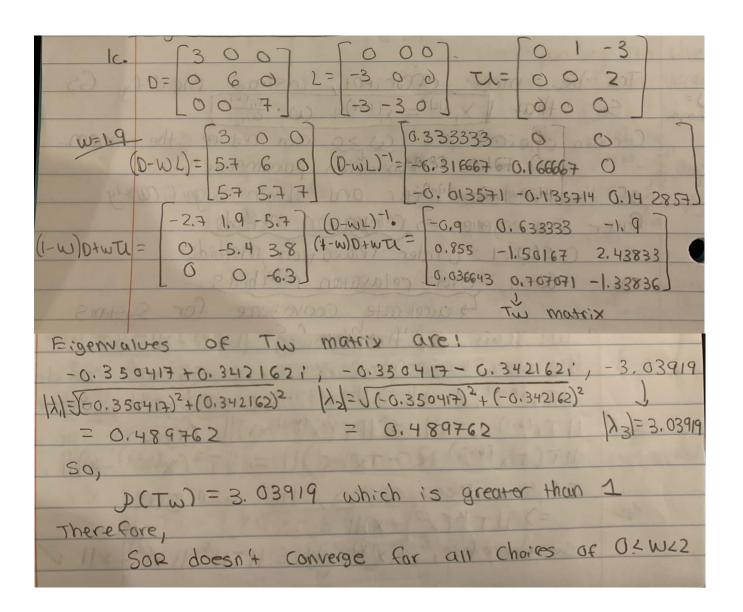
    1.469136777659948
    -0.555555612344203
    0.037036296060309
```

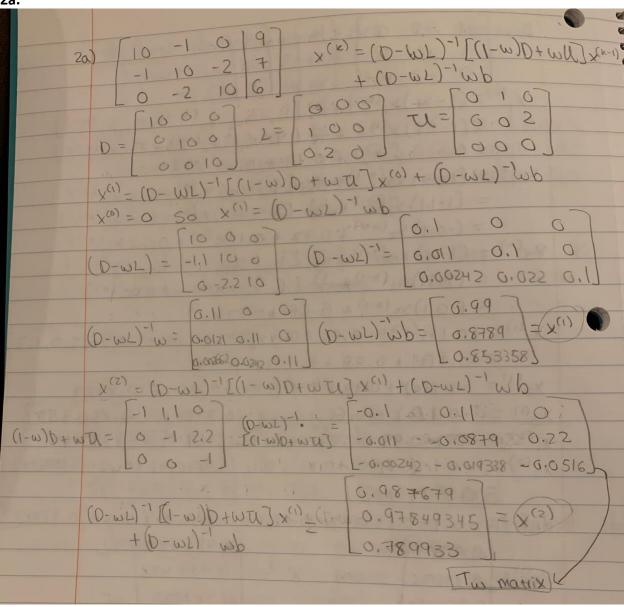
iter =

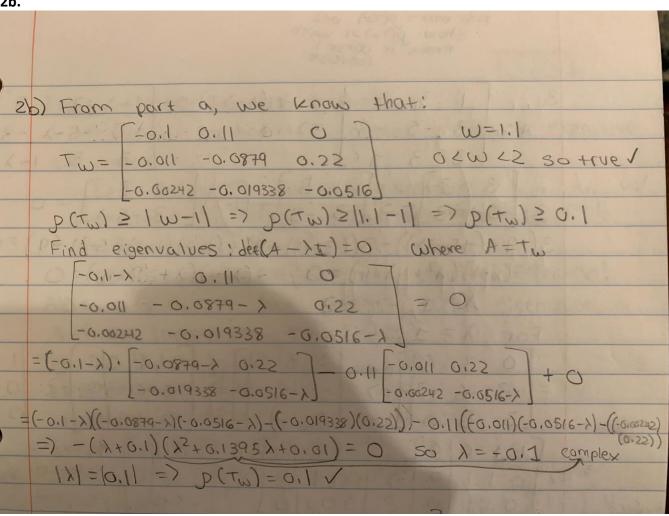
```
24
For omega = 1.0
x =
  1.469135995626869
  -0.555555703628854
   0.037037017715137
iter =
    10
For omega = 1.5
x =
  1.0e+21 *
  -1.464147513806442
  2.036080682058773
  -0.573925632861893
iter =
   150
For omega = 1.9
x =
  1.0e+63 *
  -1.235803812457661
  2.212892977860248
  -1.204491819835551
iter =
   150
```

There are many differences in the solution and number of iterations it took the algorithm to converge for the different omega values. For example, the omega values that are closer to the limit values for omega, 0 and 2, seem to take a lot longer to converge. This is especially true for the omega values that are closer to 2 in which the algorithm takes much longer to converge than the rest of the omega values I input and as a result, the solutions are very off. The omega value 1 took the least amount of iterations, 10, to converge and has the most accurate solution.

1c. The SOR algorithm does not converge for all choices of omega between 0 and 2. This is because rho of Tw is greater than 1 as shown below:







2c. The following is the output where x is the solution vector and iter is the maximum number of times it took the algorithm to converge.

```
>> HW5_script

x =

0.995789772828388
0.957894940328669
0.791579010306630

iter =
```

2d. The following is the output where x is the solution vector and iter is the maximum number of times it took the algorithm to converge.

```
>> HW5_script
x =
    0.995789409632900
    0.957894716954159
    0.791578945197880

iter =
    5
```

The number of iterations reduced by two, from 7 to 5, when using the optimal value for omega, which was calculated to be 1.01282. It speeds up convergence in this case because you are using the omega value that can quickly reach the desired solution.

2d) We can use this eq. $W = 1 + \sqrt{1 - p(T_{GS})}$ if matrix is Symmetric positive definite and tridiagonal, which it is V $T_{GS} = (D-L)^{-1}U$ where D, L, U used from part (a) $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1} = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.001 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.1 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.2$ $D-L = -1 \cdot 10 \cdot 0 \cdot (D-L)^{-1}U = 0.01 \cdot 0.2$ D-L

3. det(A->I)=0 $-3((-3+3\lambda)+9)+(1-\lambda)((1-\lambda)(-5-\lambda)+9)=0$ Xz=free 00 0 X3=free 0 0 (1) 0 0

4. A strictly diagonally dominant matrix means that its diagonal elements are larger in magnitude than the sum of the elements in its row. So, the center of each Gersgorin Circle has a larger magnitude than the radius of the circle. Because of this, a circle cannot be at the origin, or zero, meaning that zero cannot be an eigenvalue of the matrix and the matrix is nonsingular.