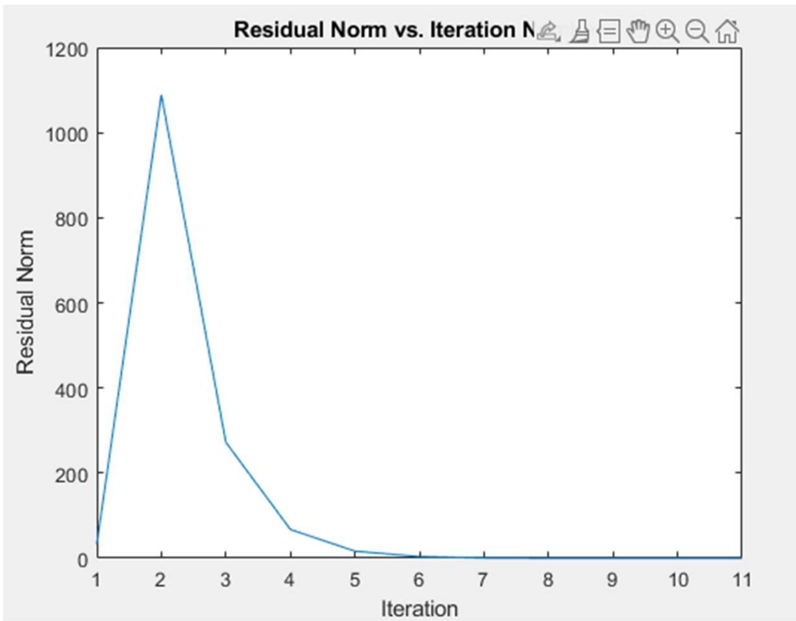
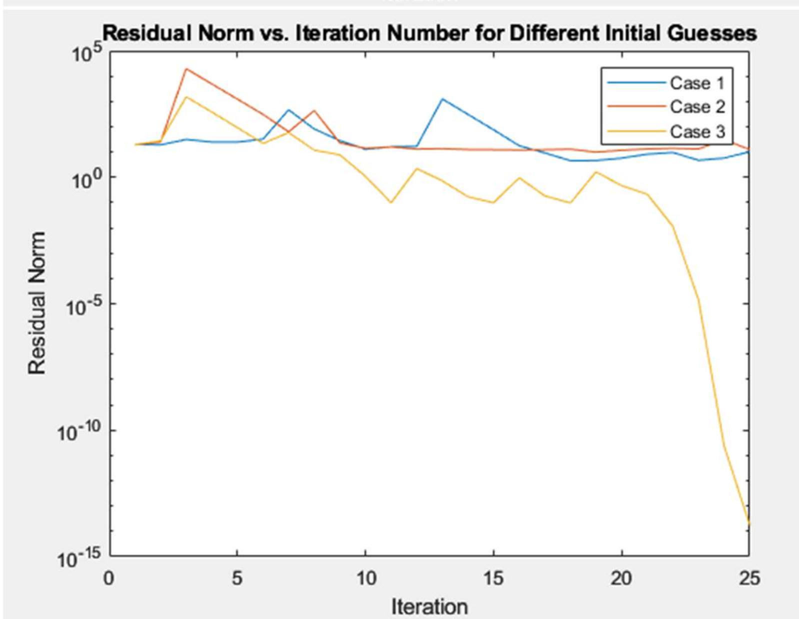
1. The code implements the Newton's method for solving a system of nonlinear equations, with a function f and its Jacobian J defined to compute the update at each iteration. The algorithm iteratively updates the initial guess until the residual norm is below the specified tolerance tol, or the maximum number of iterations is reached. To make the code more reusable, one could consider defining the function f and its Jacobian J as separate functions that can be passed as arguments to the Newton's method algorithm. This would allow the algorithm to be used with different functions without having to modify the code.





In this problem, we could see the iteration times in these two question.

```
Here is the iteration time:
    11

>> Q1_2
Number of iterations for the third case: 25
```

2.

In part (a), the code calculates the largest eigenvalue of matrix A and its corresponding eigenvector using the power method function. The eigenvectors and eigenvalues are also calculated using the eig function in MATLAB for comparison.

In part (b), the code attempts to calculate the largest eigenvalue of matrix B and its corresponding eigenvector using the power method. However, it is known that the power method fails to converge for this matrix.

In part (c), the code calculates the largest eigenvalue of matrix B with a different starting vector than in part (b).

In part (d), the code modifies the power method function to calculate the smallest eigenvalue of matrix A using the inverse power method.

```
>> Q2
Converged in 5 iterations
Eigen Vector is [2.833487e-01, 6.416743e-01]^T, value is 1.611684e+01
Power method failed to converge in 100 iterations
Converged in 1 iterations
Eigen Vector is [1.000000e+00, -1.000000e+00]^T, value is 1.000000e+00
```

```
Power method failed to converge in 100 iterations
Part (a): Largest eigenvalue of A (Power Method) = 16.116841
Eigenvalues of A (MATLAB eig) =
   16.1168   -1.1168   -0.0000

Part b: Fail to converge(Eigen Value):
    3.0000    1.0000   -3.0000


Part (c): Largest eigenvalue of A with starting vector [1; -1; 1] (Power Method) = 1.000000

Part (d): Smallest eigenvalue of A (Power Method with A inverse) = 0.000000
```

For the questions in the problem:
The power method wasn't converged because in the power method, the algorithm converges to the eigenvector corresponding to the dominant eigenvalue of the matrix. In this case, since matrix B does not have a dominant eigenvalue (i.e., the eigenvalues have similar magnitudes), the power method does not converge to the dominant eigenvector. Instead, the algorithm oscillates between the different eigenvectors, resulting in a failure to converge.

Part c: The power method algorithm converges faster when the starting vector is closer to the eigenvector of the dominant eigenvalue. Since the starting vector is already relatively close to the eigenvector corresponding to the largest eigenvalue of matrix B, the power method only needs one iteration to converge.

3. In the third question, we solve the birth and dead matrix. I used the lecture's way to build a matrix and according to the instructions on powerpoints. I created the new matrix and printed the result. By solving the eigenvalue of the matrix to show the change rate of the population. If the eigenvalue is less than 1, means the population will decrease and if it's larger than 1, it means the population will grow.

   We could see the part d's eigenvalue is greater than 1, which means that the population could be larger and larger to infinity.

```
Part (c): Largest eigenvalue of A with starting vector [1; -1; 1] (Power Method) = 1.000000

Part (d): Smallest eigenvalue of A (Power Method with A inverse) = 0.000000
>> Q3
Largest eigenvalue (original): 0.910582
Population at year 1000:
   1.0e-38 *

    0.3049
    0.3014
    0.2648
    0.1633

Largest eigenvalue (new death rate): 1.079689
   1.0e+35 *

    1.5641
    1.3038
    0.9661
    5.3857
```