1.(a)

$R_{k+1} = I - AX_{k+1} = I - A(X_k + X_k(I - AX_k)) = I - AX_k - AX_k(I - AX_k)$

$= I - AX_k - AX_k + (AX_k)^2 = (I - AX_k)^2 = {R_k}^2$

$E_{k+1} = A^{-1} - X_{k+1} = A^{-1} - (X_k + X_k(I - AX_k)) = A^{-1} - X_k - X_k + AX_kX_k$

$= (A^{-1} - X_k)*(I - AX_k) = (A^{-1} - X_k)*A*(A^{-1} - X_k) = E_kAE_k$

(b)

q1()

```
|          Accuracy and efficiency for getting A inverse          |
|                 Newton's                 |          sla.inv          |
size |     runtime     |      error      |    runtime    |    error    |
 50  |     3.13ms      |    1.69e-15     |    3.99ms     |   7.00e-15   |
 100 |     4.64ms      |    3.16e-15     |    4.33ms     |   2.22e-14   |
 150 |     5.86ms      |    4.27e-15     |    2.47ms     |   4.55e-14   |
 200 |     8.88ms      |    5.50e-15     |    2.93ms     |   6.33e-14   |
 250 |     14.24ms     |    6.59e-15     |    4.93ms     |   9.90e-14   |
 300 |     21.10ms     |    7.66e-15     |    5.40ms     |   1.28e-13   |
 350 |     28.32ms     |    7.55e-15     |    5.73ms     |   1.65e-13   |
 400 |     43.71ms     |    9.61e-15     |    7.65ms     |   2.04e-13   |
 450 |     63.08ms     |    8.53e-15     |    10.05ms    |   2.24e-13   |
```

2

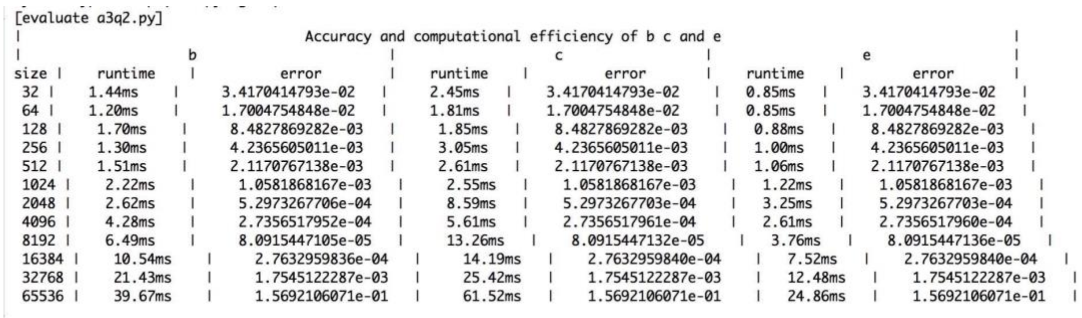
(a)

We can first factorize A by A = LU , second, perform forward substitution n times on Ly = b + f. When we get y, we know from LU method, Ud = y. Since U is upper triangular, by matrix multiplication rules, the lowest right index of matrix U (U[n][n]), times dn, equals to y[n]: dn = y[n] / U[n][n] .

(d)

Fn = fn - $\frac{get\_dn(x)}{get\_dn\prime(x)}$

(f)



```
[evaluate a3q2.py]
|                                   Accuracy and computational efficiency of b c and e                                   |
|                         b                          |                          c                          |                          e                          |
size |    runtime    |          error          |    runtime    |          error          |    runtime    |          error          |
 32 |    1.44ms     |    3.4170414793e-02    |    2.45ms     |    3.4170414793e-02    |    0.85ms     |    3.4170414793e-02    |
 64 |    1.20ms     |    1.7004754848e-02    |    1.81ms     |    1.7004754848e-02    |    0.85ms     |    1.7004754848e-02    |
 128 |    1.70ms     |    8.4827869282e-03    |    1.85ms     |    8.4827869282e-03    |    0.88ms     |    8.4827869282e-03    |
 256 |    1.30ms     |    4.2365605011e-03    |    3.05ms     |    4.2365605011e-03    |    1.00ms     |    4.2365605011e-03    |
 512 |    1.51ms     |    2.1170767138e-03    |    2.61ms     |    2.1170767138e-03    |    1.06ms     |    2.1170767138e-03    |
 1024 |    2.22ms     |    1.0581868167e-03    |    2.55ms     |    1.0581868167e-03    |    1.22ms     |    1.0581868167e-03    |
 2048 |    2.62ms     |    5.2973267706e-04    |    8.59ms     |    5.2973267703e-04    |    3.25ms     |    5.2973267703e-04    |
 4096 |    4.28ms     |    2.7356517952e-04    |    5.61ms     |    2.7356517961e-04    |    2.61ms     |    2.7356517960e-04    |
 8192 |    6.49ms     |    8.0915447105e-05    |    13.26ms    |    8.0915447132e-05    |    3.76ms     |    8.0915447136e-05    |
 16384 |    10.54ms    |    2.7632959836e-04    |    14.19ms    |    2.7632959840e-04    |    7.52ms     |    2.7632959840e-04    |
 32768 |    21.43ms    |    1.7545122287e-03    |    25.42ms    |    1.7545122287e-03    |    12.48ms    |    1.7545122287e-03    |
 65536 |    39.67ms    |    1.5692106071e-01    |    61.52ms    |    1.5692106071e-01    |    24.86ms    |    1.5692106071e-01    |
```

By efficiency, method (e) shows smallest runtime, and c has biggest runtime, and, three methods shows nearly the same accuracy, so I would choose (e) method.

3(a)

The non-linear system is:

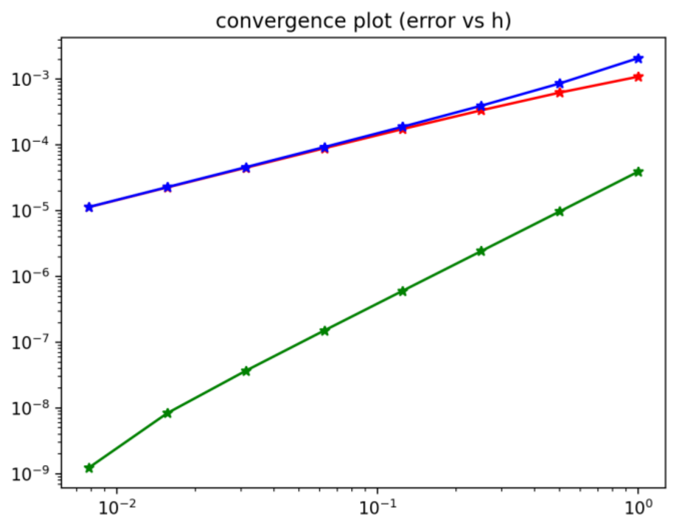
$$\begin{gathered} S(t_{i+1}) - S(t_i) + h\frac{\beta}{N}S(t_{i+1})I(t_{i+1}) \\ E(t_{i+1}) - E(t_i) - h\frac{\beta}{N}S(t_{i+1})I(t_{i+1}) + whE(t_{i+1}) \\ I(t_{i+1}) - I(t_i) - whE(t_{i+1}) + h\gamma I(t_{i+1}) \\ R(t_{i+1}) - R(t_i) - h\gamma I(t_{i+1}) \end{gathered}$$

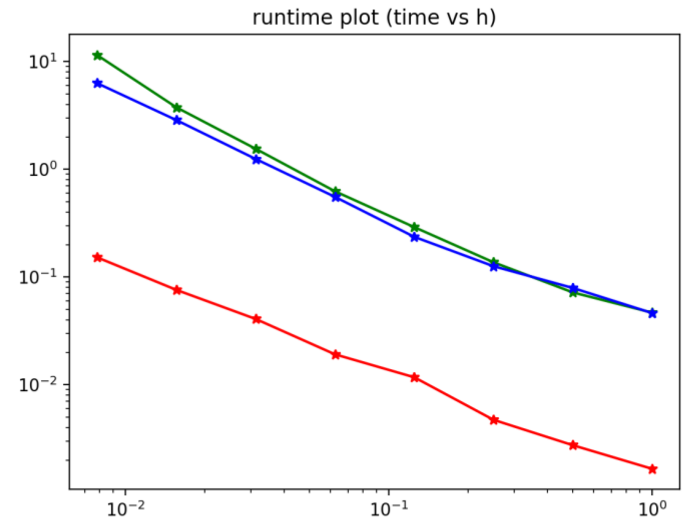
The Jacobian matrix for this system:

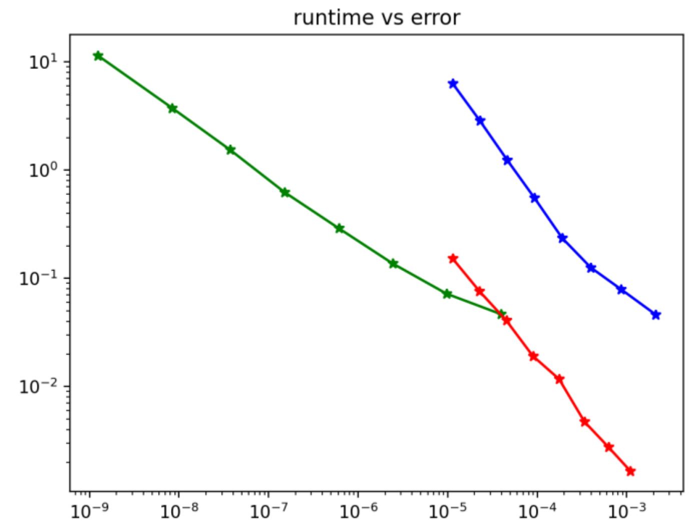
$$\begin{array}{cccc} 1 + h\frac{\beta}{N}I(t_{i+1}), & 0, & h\frac{\beta}{N}S(t_{i+1}), & 0 \\ -h\frac{\beta}{N}I(t_{i+1}), & 1 + wh, & -h\frac{\beta}{N}S(t_{i+1}), & 0 \\ 0, & -wh, & 1 + h\gamma, & 0 \\ 0, & 0, & -h\gamma, & 1 \end{array}$$

For method three, the Jacobian matrix is very similar to the one in method two, but for every term with multiplier ‘h’, we need to divide h by 2.

c)







I conv order 0.9963405043688611

II conv order 1.0036792433754846

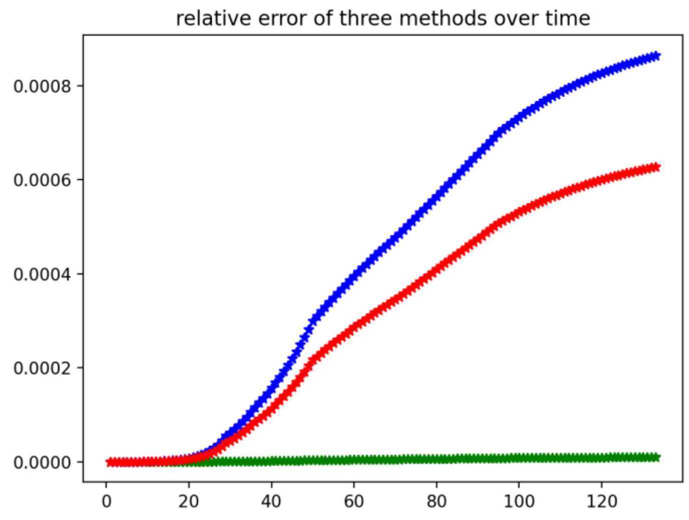
III conv order 2.7566660944860453

From the convergence rate, method 1,2 are linear convergence, and method 3 is quadratic convergence.

Method 3 takes longest run time, and method one takes smallest. Method 2,3 takes longer time because they have a non – linear system, and needs to call fsolve to get the solution, while method 1 can return the value simply using xi and f'(xi).

For a same method, the error tends to be smaller if there is a longer runtime, so 'run time vs error' plot is downward sloping.

d)



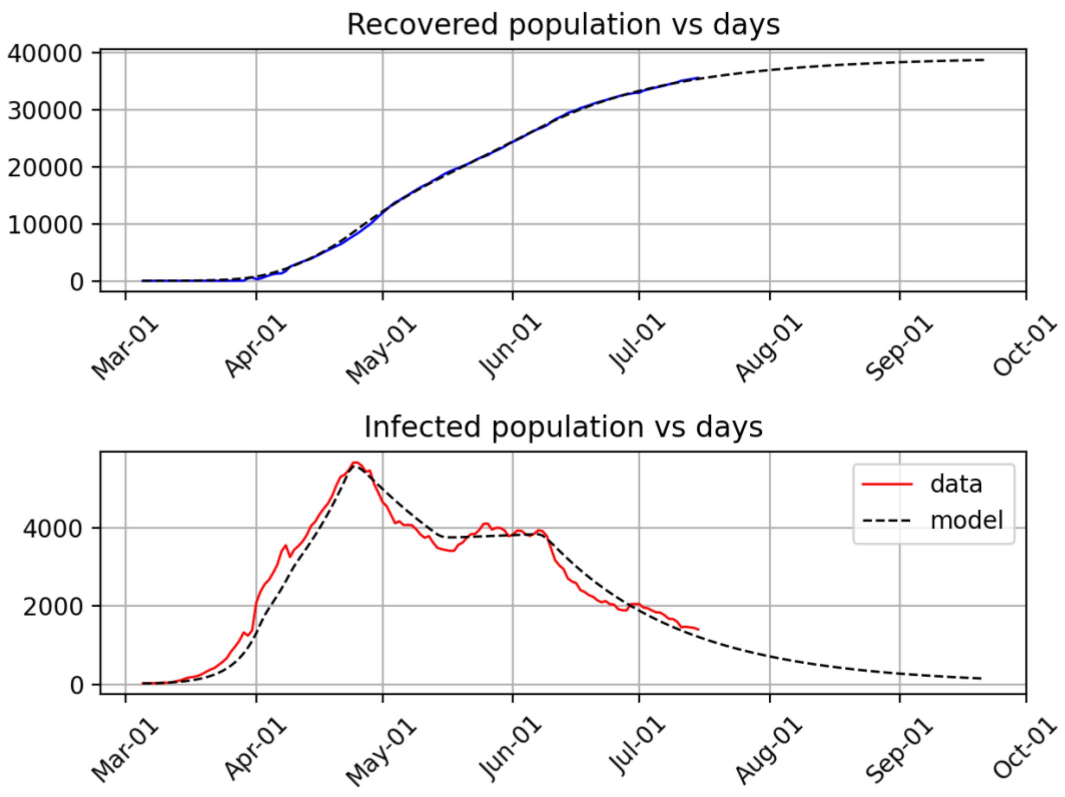
From the graph, we can observe that for over 133 days, the error of three methods are non-decreasing, or always increasing, because when we want to estimate something, we are losing precision when we are predicting for a further time.

Method three seems to be the most accurate approach in the methods, while it takes most time to calculate. Because it takes both method one and method two into consideration, the arithmetic is more complicated. For the accuracy method three gets, it is because it is quadratic convergence, while method one and two are linear convergence.

Method one is more accurate and takes less time than method two. The ways they predict future values are similar, by using the slope(or derivative) of x. Method two takes more time because it calls fsolve function, method one simply return the value, calling fsolve to solve the linear system can be the main reason it loss precision, compared to method one.

Along the three methods, method three should be used if accuracy is very important and we do not care about run time, we can use method two if we want to get the answer quicker but not care about accuracy.

e)



From the figure, we can observe that, as time going, smaller portion of population will be infected, and more and more people will be recovered from the disease.