Logan Singerman Final Project

1.

A) $x'(t) = \cos(t) - \sin(t) + t^2$, $t > 0$
$x(0) = 3$, over the interval $0 \le t \le 1.0$.

To find the exact solution to the ODE, we use separation of variables and integrate both sides:

$$\int dx = \int \cos(t) - \sin(t) + t^2 \, dt$$

$$x(t) = \sin(t) + \cos(t) + (t^3)/3 + c$$

when we plug in the initial condition $x(0) = 3$, we find $c=2$, thus

$$x(t) = \sin(t) + \cos(t) + (t^3)/3 + 2$$

B) The iterations of the TS(3) and TS(4) methods are found by using the Taylor expansion about $x(t+h)$.

TS(3): $x_{n+1} = x_n + h(\cos(t_n)-\sin(t_n)+t_n^2) + ½ * h^2(-\sin(t_n)-\cos(t_n) + 2t_n) + 1/6 * h^3(-\cos(t_n)+\sin(t_n)+2)$

TS(4): $x_{n+1} = x_n + h(\cos(t_n)-\sin(t_n)+t_n^2) + ½ * h^2(-\sin(t_n)-\cos(t_n) + 2t_n) + 1/6 * h^3(-\cos(t_n)+\sin(t_n)+2) + 1/24 * h^4(\sin(t_n)+\cos(t_n)$

The Local Truncation Error for the Taylor Series methods is the p+1 term when the method is consistent to order p:

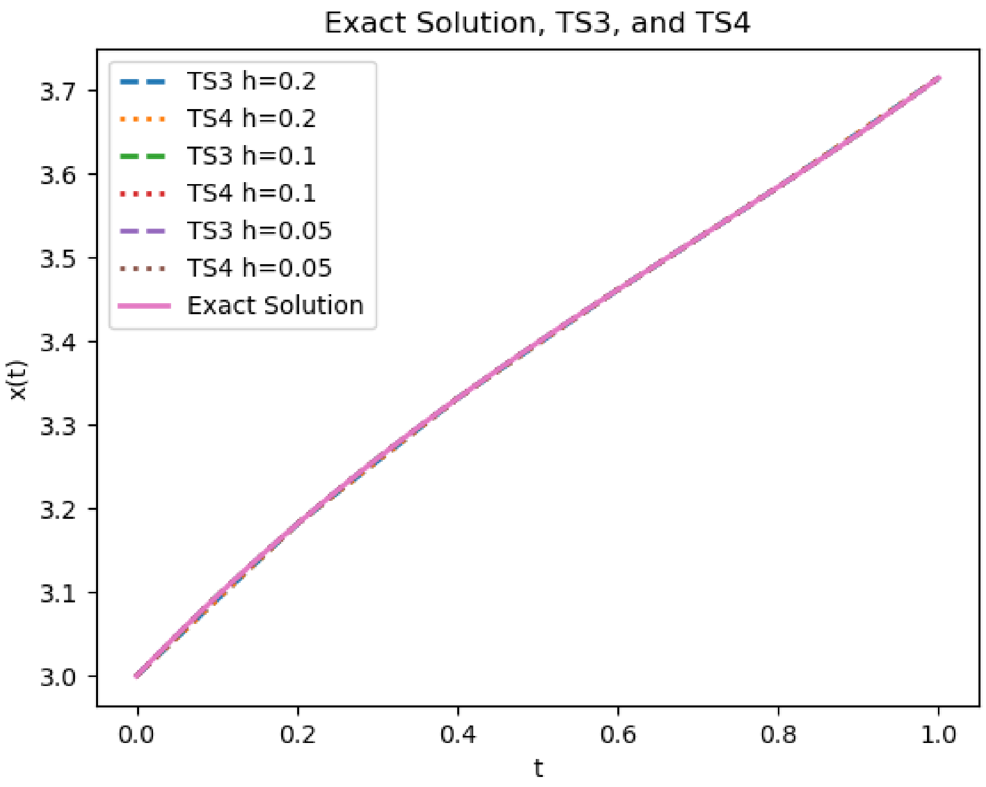
LTE(TS3) = $1/24 * h^4(\sin(\xi) + \cos(\xi)$ for $\xi \in [t_n, t_{n+1}]$

LTE(TS4) $1/120 * h^5(\cos(\xi)-\sin(\xi))$ for $\xi \in [t_n, t_{n+1}]$

D)

h	TS3	TS4	GE-TS3	GE-TS4	GE-TS3 /h^3	GE-TS4 /h^4
0.20	3.714681	3.715100	0.000426	6.242742e-06	0.053224	0.003902
0.10	3.715053	3.715106	0.000054	3.542176e-07	0.053729	0.003542
0.05	3.715100	3.715107	0.000007	2.101237e-08	0.053974	0.003362

E)



F) For each step size in the TS(3) and TS(4) methods, the approximate solution has very low error, and the error continues to decrease by a factor of h^p when h is decreased.

2.

A) Method 1 is a 4-stage 4th order Runge-Kutta method and is generally referred to as "The" Runge-Kutta method. From the Butcher array, we find that the values of K1 through K4 are:

$$K1 = f(t,x) = x'(t)$$

$$K2 = f(t + h/2,\ x + h/2 * K1)$$

$$K3 = f(t + h/2,\ x + h/2 * K2)$$

$$K4 = f(t + h,\ x + h*K3)$$

$$\text{And } x_{n+1} = x_n + h/6(K1 + 2K2 + 2K3 + K4)$$

Method 2 is a 3-stage 3rd order Runge-Kutta method, and we observe the Butcher array to find the values of K1 through K3 and the iterative formula:

K1 = f(t,x) = x'(t)

K2 = f(t + h/2, x + h/2 * K1)

K3 = f(t + h, x - h*K1 + 2h*K2)

And $x_{n+1} = x_n + h/6(K1 + 4K2 + K3)$

To find the regions of absolute stability for methods 1 and 2, we need to compute the stability function of each method and plot the regions in the complex plane where:

$$|R(\hat{h})| < 1$$

To do this, we multiply each step K by h, when we use $x'(t) = \lambda x(t)$ for $\lambda \in \mathbf{C}$, $\hat{h} = \lambda h$

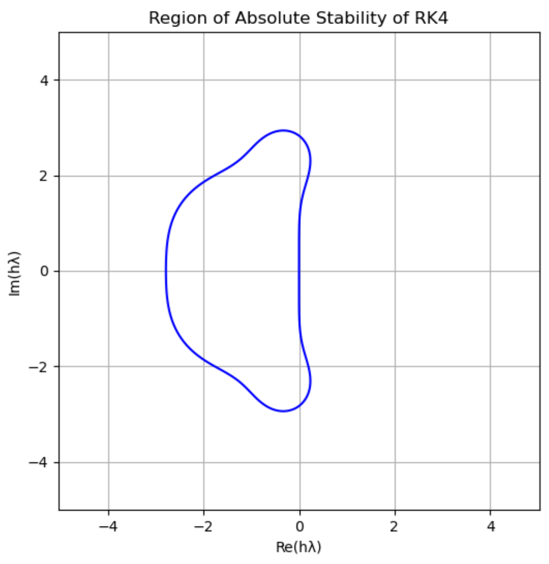
To reduce the amount of computation, we know that when an RK method is s-stage order s, it's stability function is the first s+1 terms in the Taylor series expansion of e^ $\hat{h}$. Thus, the stability functions of method 1 and 2 are:

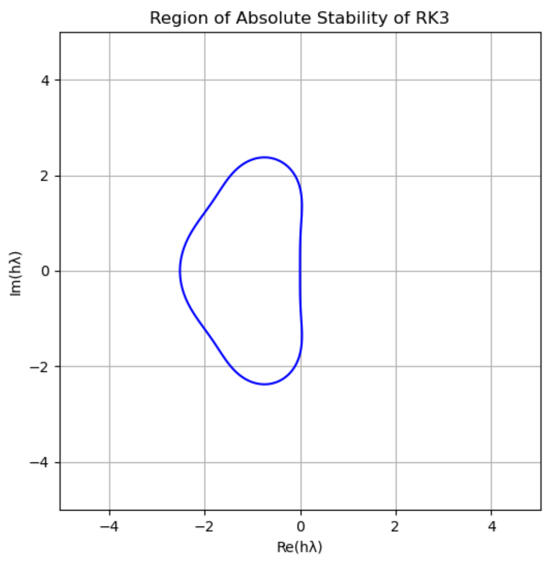
$$\text{RK4: } R(\hat{h}) = 1 + \hat{h} + ½ * \hat{h}^2 + 1/6 * \hat{h}^3 + 1/24 * \hat{h}^4$$

Similarly

$$\text{RK3: } R(\hat{h}) = 1 + \hat{h} + ½ * \hat{h}^2 + 1/6 * \hat{h}^3$$

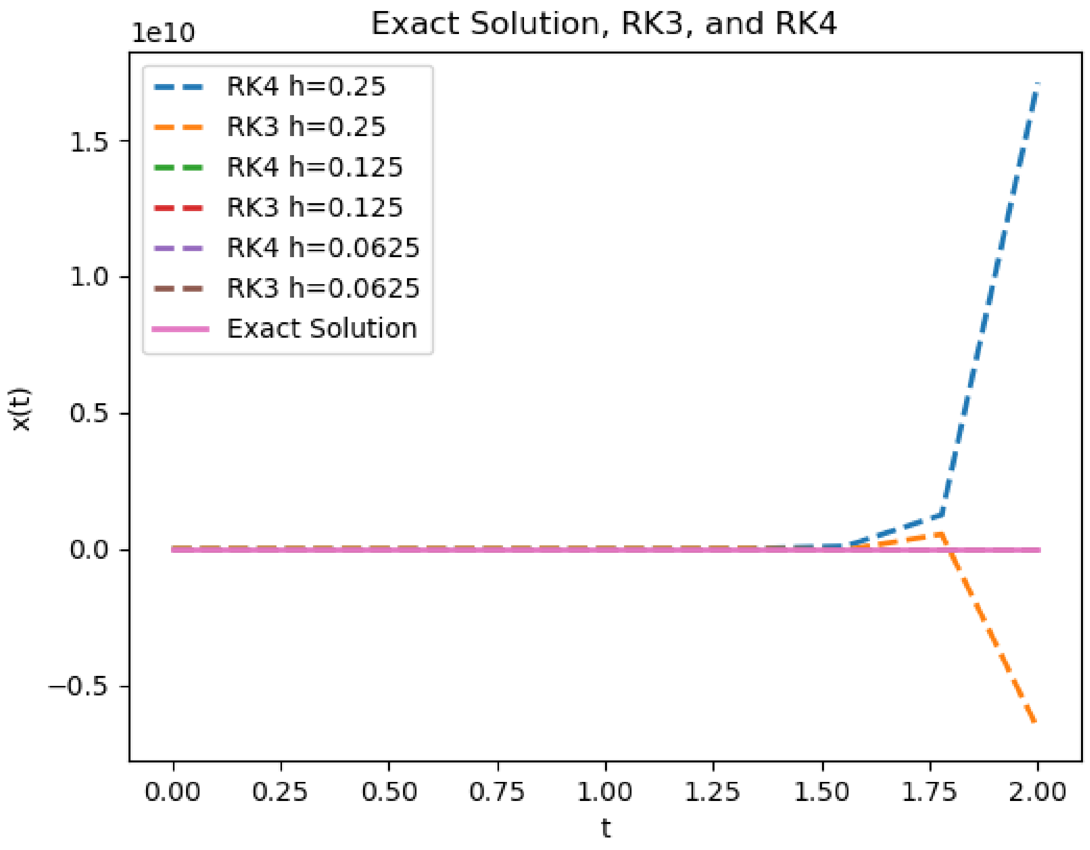
Plotting these regions in the complex plane gives the following graphs:





C)

h	RK3 t=2	RK4 t=2	GE_RK3 t=2	GE_RK4 t=2
0.2500	-6.602740e+09	1.709472e+10	6.602740e+09	-1.709472e+10
0.1250	-6.991374e-01	6.335167e-04	6.991374e-01	-6.335167e-04
0.0625	2.181542e-23	1.249582e-17	4.248332e-18	-8.247466e-18



E) For h=0.25 in the RK3 and RK4 methods, the value of λh is no longer in the region of absolute stability, therefore these approximations diverge and become very unstable, while the other approximations have very low error.

3.

A) To find a finite difference approximation formula to the BVP:

$$-0.05x''(t) + r(x)x'(t) = 2$$

$$x(0) = 0 \text{ and } x(1) = 2$$

We use the central difference operators $\delta\ ^2U_m$ and ΔU_m:

$$L_hU_m = -0.05*h^{-2}[U_{m-1} – 2U_m + U_{m+1}] + r(x_m)/2h[U_{m+1} – U_{m-1}]$$

To show that the method is consistent of order 2, we subtract the exact form from the linear difference operator:

$$R_m = L_hU_m – f_m$$

$$R_m = -0.05*h^{-2}\ \delta^2U_m + r(x_m)*h^{-1}\Delta U_m – (-0.05x''(t) + r(x)x'(t))$$

From Taylor series expansions, we know that x'' and x' can be written as:

$$x'' = h^{-2}(x_{m+1} -2x_m + x_{m-1}) – 1/12 * h^2x_m'''' + . . . O(h^4)$$

$$x' = ½*h^{-1}(x_{m+1} – x_{m-1}) + 1/6 * h^2x_m''' + . . . O(h^4)$$

Therefore, $R_m = 0.05/12*h^2x_m'''' + r(x)/6 *h^2x_m''' + O(h^4)$

As $h \to 0$, $R_m \to 0$ with order h^2

B) To find the conditions for h where the method is stable, we need to rearrange the terms in the linear difference operator and conclude that they satisfy the theorem of Inverse Monotonicity:

$$L_hU_m = -[0.05/h^2 + r(x)/2h]U_{m-1} + [0.1/h^2]U_m – [0.05/h^2 – r(x)/2h]U_{m+1}$$

We need the first and last coefficients to be greater than or equal to 0, and the middle coefficient to be greater than 0, and greater than or equal to the sum of the first and last coefficients.

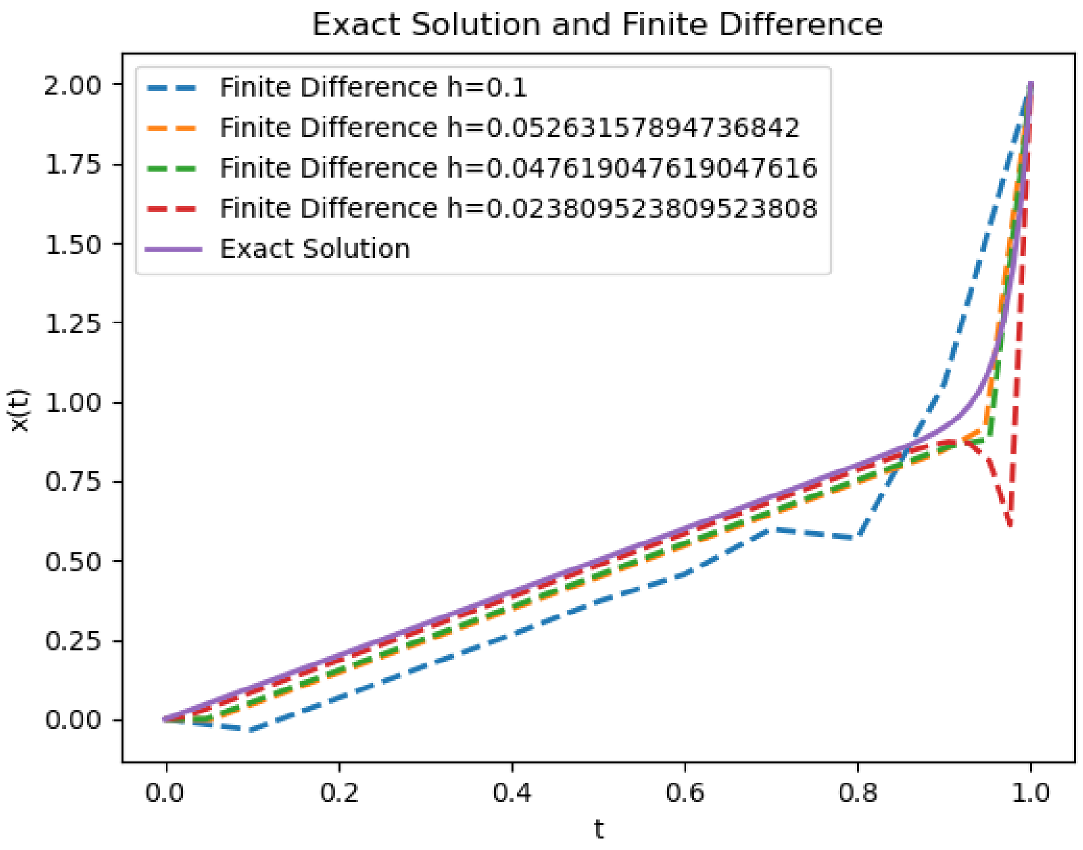
$$[0.05/h^2 + r(x)/2h] + [0.05/h^2 – r(x)/2h] = [0.1/h^2]$$

When we rearrange the inequalities for the first and last coefficients, we find that:

$$h <= 0.1/r(x) \text{ and } h >= -0.1/r(x)$$

D)

h	Max Difference
0.100000	0.229952
0.052632	0.151538
0.047619	0.218573
0.023810	0.750570



F) The order of convergence is O(h^2), as shown in part A

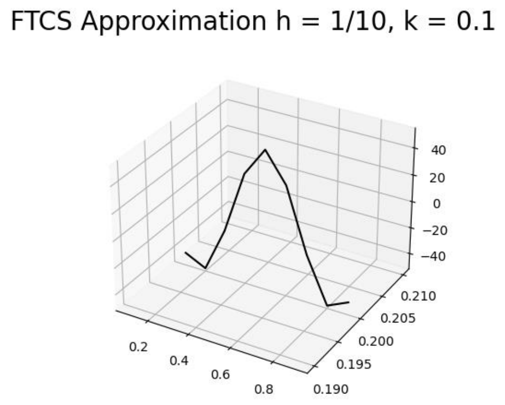
G) The results show that when we approximate the solution using the smallest step size h, it leads to the largest max difference between the exact solution and the approximate solution. In theory, as h -> 0, the error -> 0, however in numerical computation of matrices, round off error can accumulate when the matrices are very large, they can also be ill-conditioned. This can lead to a specific value of h performing better than smaller values of h when it reaches an optimal ratio of error in the exact solution and less round-off error in the matrix computations.

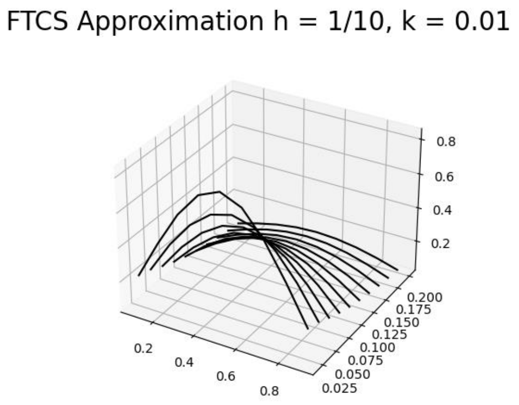
4.

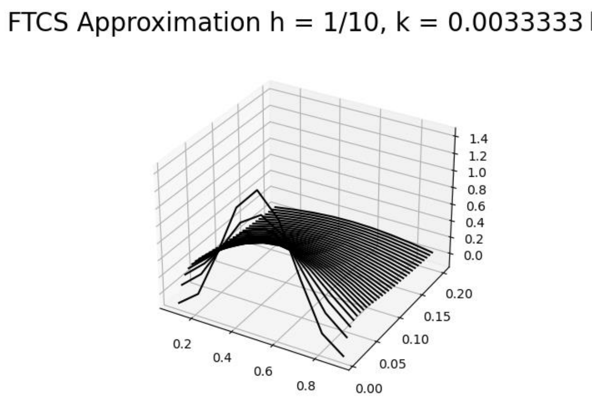
Parts A and B are done in code, these are the values for part C data:

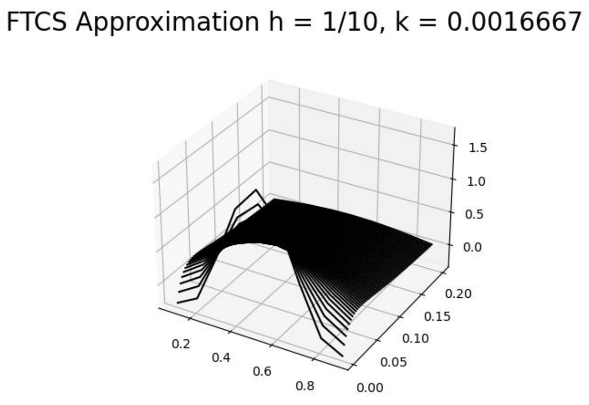
r	Error FTSC	Error BTSC	Error Crank-Nicholson
0.166667	0.000005	0.004514	0.002260
0.333333	0.002269	0.006750	0.002242
1.000000	0.011496	0.015566	0.002045
10.000000	52.341345	0.128158	0.365935

These are the generated graphs of each method at each value of r:

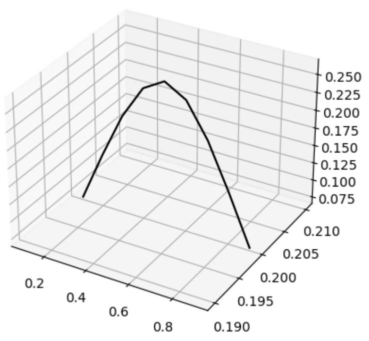




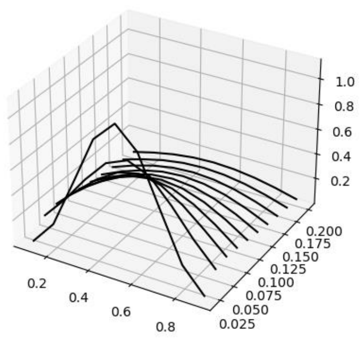




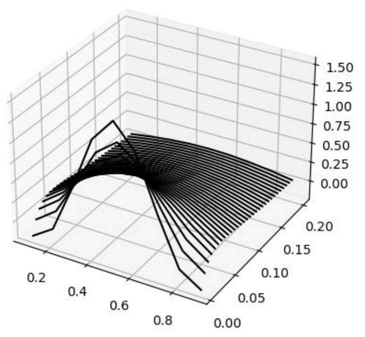
BTCS Approximation h = 1/10, k = 0.1



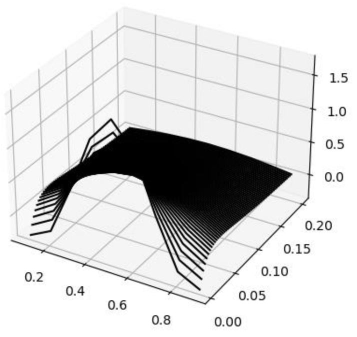
BTCS Approximation h = 1/10, k = 0.01

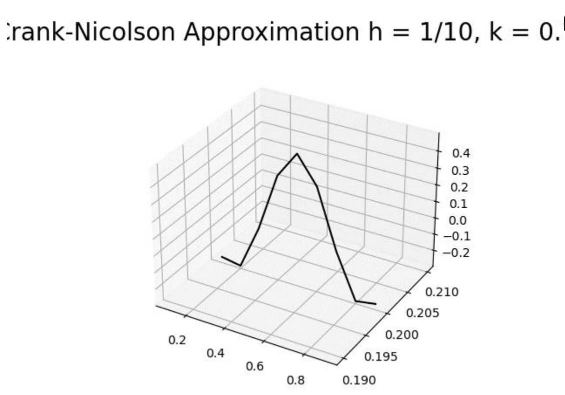


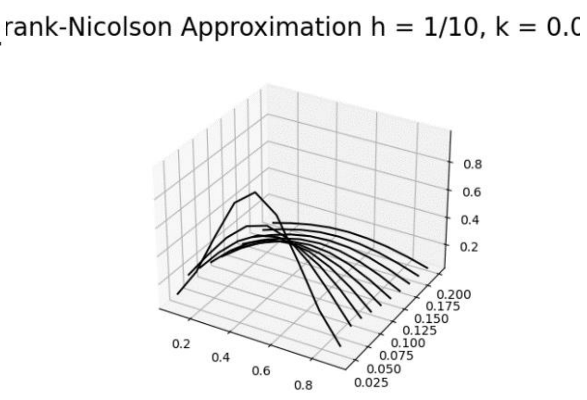
BTCS Approximation h = 1/10, k = 0.0033333

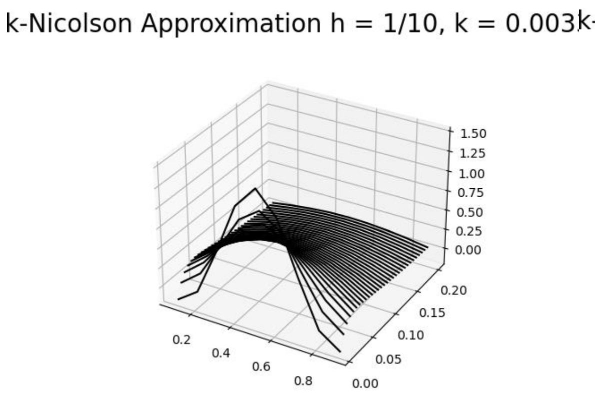


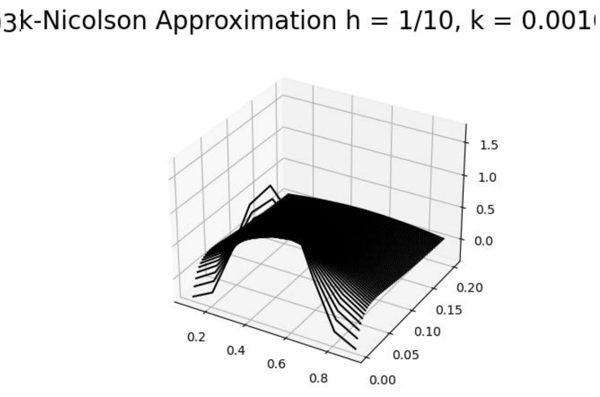
BTCS Approximation h = 1/10, k = 0.0016667











E)

1. FTCS (Forward Time, Central Space) method: This method has a first-order convergence in time and second-order convergence in space. The local truncation error is O(k) + O(h^2), where k is the time step size and h is the spatial step size.
2. BTCS (Backward Time, Central Space) method: This method has a first-order convergence in time and second-order convergence in space. The local truncation error is O(k) + O(h^2), where k is the time step size and h is the spatial step size.
3. Crank-Nicolson method: This method has a second-order convergence in both time and space. The local truncation error is O(k^2) + O(h^2), where k is the time step size and h is the spatial step size.

F) Conclusions: The FTCS method has the lowest error between the 3 methods when the ratio between the step size in time and space is small. This method also has the largest error when the ratio is large. The BTSC has a very consistent amount of error, since it is unconditionally stable, and the Crank-Nicholson method has, in general, a lower error than the BTSC method.

Instructions to run the python file are included inside the file. The file will produce all of the graphs shown in this report when executed.