TUT 9

1.cpp

```
#include <iostream>
#include<bits/stdc++.h>
using namespace std;
// Function definition for every methods
double analy(double tau) {
    return 1.0/(1.0+tau);
}

double exp_Eul(double y,double dt) {
    return y-dt*y*y;
}

double imp_Eul(double y, double dt) {
    return y/(1.0+dt*y);
}

double c_Nich(double y, double dt) {
    double y_new=y-dt*y*y/2.0;
    return y_new/(1.0+dt*y_new/ 2.0);
}

void solution(double y0,double tauEnd,const vector<double>&timeSteps,const
string& method) {
    cout<<method<<" method:\n";
    cout <<"dt\t\ttau\t\tNumer\tAnalyt\tR E(%)\n";

    for(double dt:timeSteps){
        double tau = 0.0;
        double y = y0;

        while (tau <= tauEnd) {
            double analytical=analy(tau);
            double relativeError=100.0*abs((y-analytical)/analytical);

            cout<<fixed<<setprecision(2)<<dt<<"\t"
```

```
                <<fixed<<setprecision(3)<<tau<<"\t"
                <<y<<"\t"<<analytical<<"\t"
                <<relativeError<<"\n";

            if (method =="Explicit Euler") {
                y=exp_Eul(y,dt);
            }else if(method=="Implicit Euler"){
                y=imp_Eul(y, dt);
            } else if (method=="Crank Nicholson") {
                y=c_Nich(y,dt);
            }
            tau+=dt;
        }
        cout << endl;
    }
}

int main() {
    double y0 =1.0;
    double stop_time=2.0;
    vector<double> timeSteps = {0.1,0.2,0.5,1.0,2.0};
    solution(y0,stop_time,timeSteps,"Explicit Euler");
    solution(y0,stop_time,timeSteps,"Implicit Euler");
    solution(y0,stop_time,timeSteps,"Crank-Nicholson");
    return 0;
}
```

Output

```
Explicit Euler method:
dt     tau   Numer Analyt      R E(%)
0.10   0.000 1.000 1.000 0.000
0.10   0.100 0.900 0.909 1.000
0.10   0.200 0.819 0.833 1.720
0.10   0.300 0.752 0.769 2.250
0.10   0.400 0.695 0.714 2.646
0.10   0.500 0.647 0.667 2.946
0.10   0.600 0.605 0.625 3.174
0.10   0.700 0.569 0.588 3.348
0.10   0.800 0.536 0.556 3.481
0.10   0.900 0.507 0.526 3.582
0.10   1.000 0.482 0.500 3.657
0.10   1.100 0.459 0.476 3.713
```

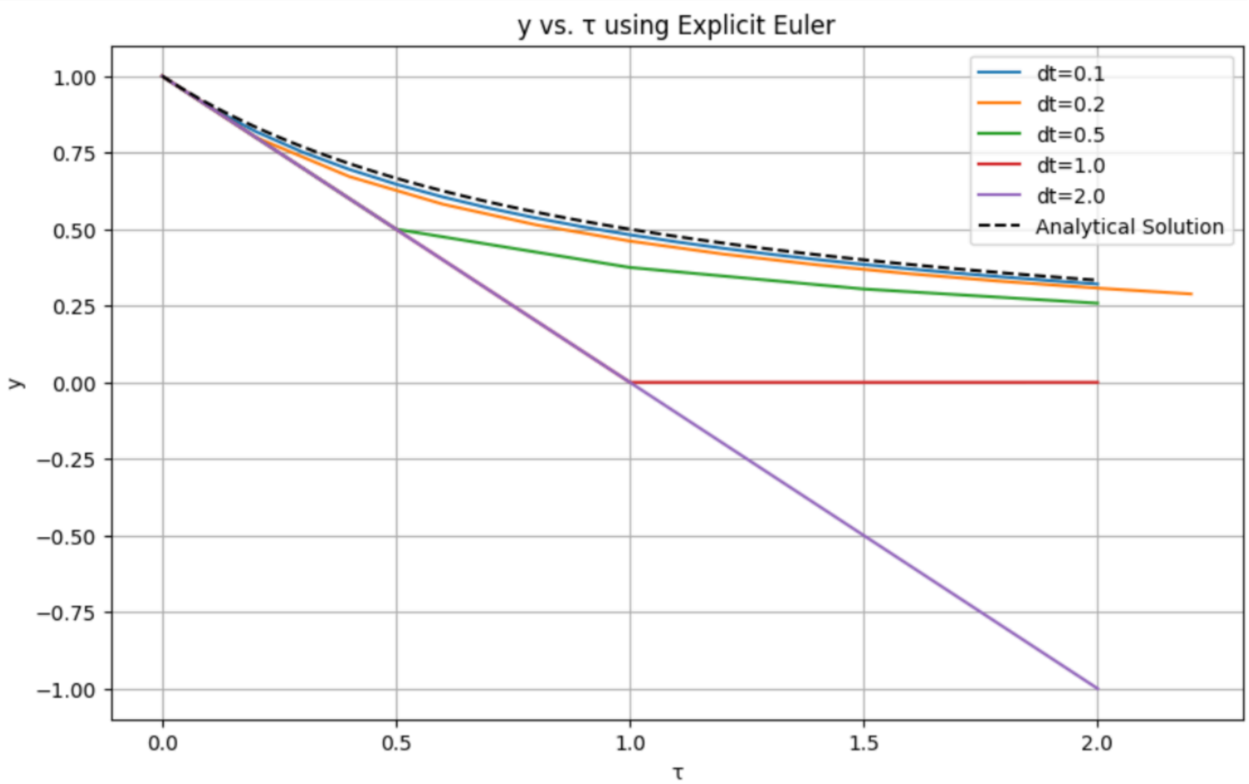
```
0.10  1.200 0.437 0.455 3.753
0.10  1.300 0.418 0.435 3.780
0.10  1.400 0.401 0.417 3.797
0.10  1.500 0.385 0.400 3.806
0.10  1.600 0.370 0.385 3.807
0.10  1.700 0.356 0.370 3.803
0.10  1.800 0.344 0.357 3.795
0.10  1.900 0.332 0.345 3.783

0.20  0.000 1.000 1.000 0.000
0.20  0.200 0.800 0.833 4.000
0.20  0.400 0.672 0.714 5.920
0.20  0.600 0.582 0.625 6.931
0.20  0.800 0.514 0.556 7.478
0.20  1.000 0.461 0.500 7.766
0.20  1.200 0.419 0.455 7.900
0.20  1.400 0.384 0.417 7.940
0.20  1.600 0.354 0.385 7.919
0.20  1.800 0.329 0.357 7.860
0.20  2.000 0.307 0.333 7.776

0.50  0.000 1.000 1.000 0.000
0.50  0.500 0.500 0.667 25.000
0.50  1.000 0.375 0.500 25.000
0.50  1.500 0.305 0.400 23.828
0.50  2.000 0.258 0.333 22.519

1.00  0.000 1.000 1.000 0.000
1.00  1.000 0.000 0.500 100.000
1.00  2.000 0.000 0.333 100.000

2.00  0.000 1.000 1.000 0.000
2.00  2.000 -1.000      0.333 400.000

Implicit Euler method:
dt    tau   Numer Analyt      R E(%)
0.10  0.000 1.000 1.000 0.000
0.10  0.100 0.909 0.909 0.000
0.10  0.200 0.833 0.833 0.000
0.10  0.300 0.769 0.769 0.000
0.10  0.400 0.714 0.714 0.000
0.10  0.500 0.667 0.667 0.000
0.10  0.600 0.625 0.625 0.000
```

```
0.10  0.700 0.588 0.588 0.000
0.10  0.800 0.556 0.556 0.000
0.10  0.900 0.526 0.526 0.000
0.10  1.000 0.500 0.500 0.000
0.10  1.100 0.476 0.476 0.000
0.10  1.200 0.455 0.455 0.000
0.10  1.300 0.435 0.435 0.000
0.10  1.400 0.417 0.417 0.000
0.10  1.500 0.400 0.400 0.000
0.10  1.600 0.385 0.385 0.000
0.10  1.700 0.370 0.370 0.000
0.10  1.800 0.357 0.357 0.000
0.10  1.900 0.345 0.345 0.000

0.20  0.000 1.000 1.000 0.000
0.20  0.200 0.833 0.833 0.000
0.20  0.400 0.714 0.714 0.000
0.20  0.600 0.625 0.625 0.000
0.20  0.800 0.556 0.556 0.000
0.20  1.000 0.500 0.500 0.000
0.20  1.200 0.455 0.455 0.000
0.20  1.400 0.417 0.417 0.000
0.20  1.600 0.385 0.385 0.000
0.20  1.800 0.357 0.357 0.000
0.20  2.000 0.333 0.333 0.000

0.50  0.000 1.000 1.000 0.000
0.50  0.500 0.667 0.667 0.000
0.50  1.000 0.500 0.500 0.000
0.50  1.500 0.400 0.400 0.000
0.50  2.000 0.333 0.333 0.000

1.00  0.000 1.000 1.000 0.000
1.00  1.000 0.500 0.500 0.000
1.00  2.000 0.333 0.333 0.000

2.00  0.000 1.000 1.000 0.000
2.00  2.000 0.333 0.333 0.000

Crank-Nicholson method:
dt    tau   Numer Analyt      R E(%)
0.10  0.000 1.000 1.000 0.000
0.10  0.100 1.000 0.909 10.000
```

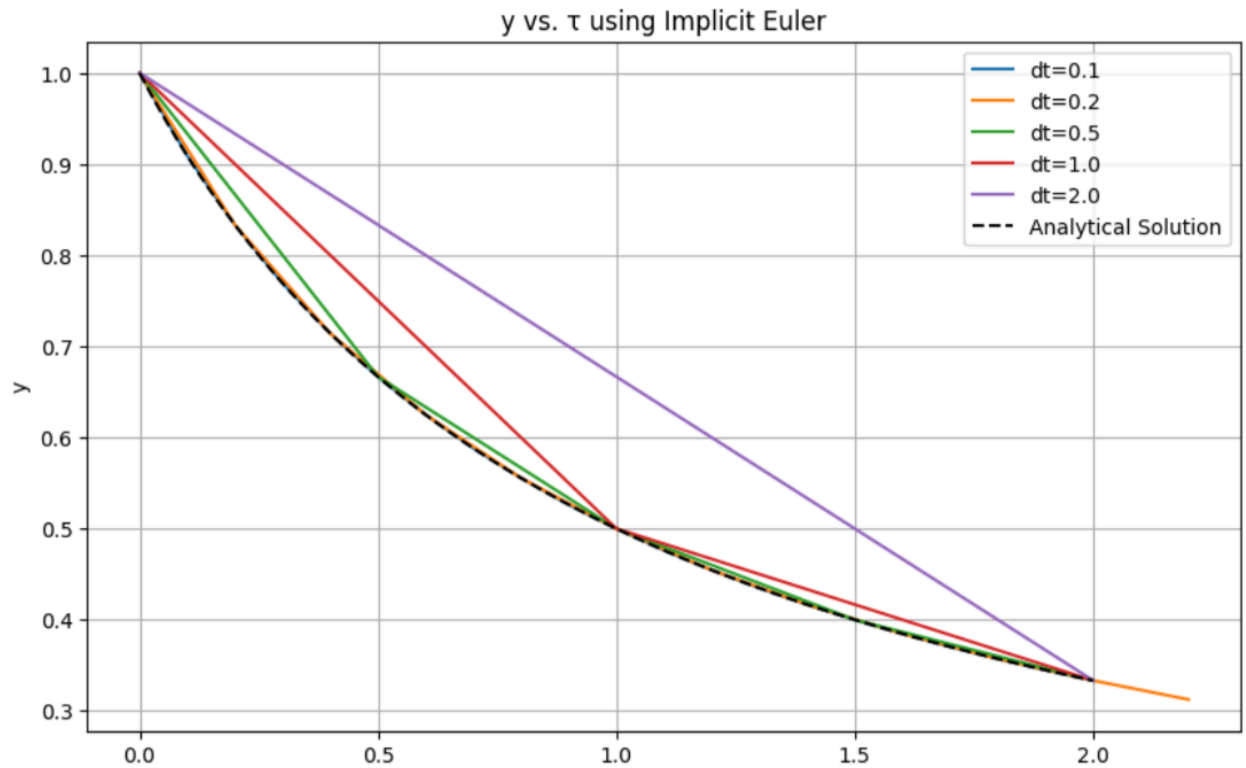
```
0.10  0.200 1.000 0.833 20.000
0.10  0.300 1.000 0.769 30.000
0.10  0.400 1.000 0.714 40.000
0.10  0.500 1.000 0.667 50.000
0.10  0.600 1.000 0.625 60.000
0.10  0.700 1.000 0.588 70.000
0.10  0.800 1.000 0.556 80.000
0.10  0.900 1.000 0.526 90.000
0.10  1.000 1.000 0.500 100.000
0.10  1.100 1.000 0.476 110.000
0.10  1.200 1.000 0.455 120.000
0.10  1.300 1.000 0.435 130.000
0.10  1.400 1.000 0.417 140.000
0.10  1.500 1.000 0.400 150.000
0.10  1.600 1.000 0.385 160.000
0.10  1.700 1.000 0.370 170.000
0.10  1.800 1.000 0.357 180.000
0.10  1.900 1.000 0.345 190.000

0.20  0.000 1.000 1.000 0.000
0.20  0.200 1.000 0.833 20.000
0.20  0.400 1.000 0.714 40.000
0.20  0.600 1.000 0.625 60.000
0.20  0.800 1.000 0.556 80.000
0.20  1.000 1.000 0.500 100.000
0.20  1.200 1.000 0.455 120.000
0.20  1.400 1.000 0.417 140.000
0.20  1.600 1.000 0.385 160.000
0.20  1.800 1.000 0.357 180.000
0.20  2.000 1.000 0.333 200.000

0.50  0.000 1.000 1.000 0.000
0.50  0.500 1.000 0.667 50.000
0.50  1.000 1.000 0.500 100.000
0.50  1.500 1.000 0.400 150.000
0.50  2.000 1.000 0.333 200.000

1.00  0.000 1.000 1.000 0.000
1.00  1.000 1.000 0.500 100.000
1.00  2.000 1.000 0.333 200.000

2.00  0.000 1.000 1.000 0.000
2.00  2.000 1.000 0.333 200.000
```

```
=== Code Execution Successful ===
```

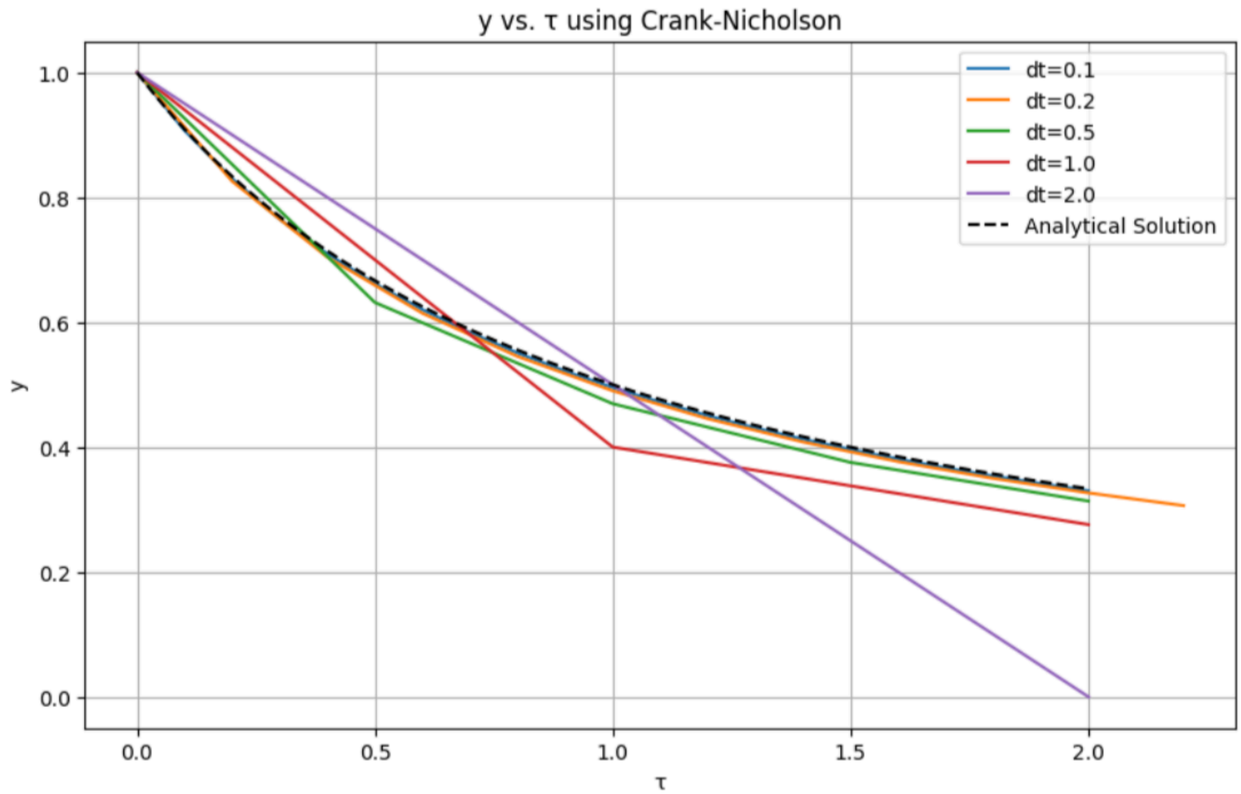
A1



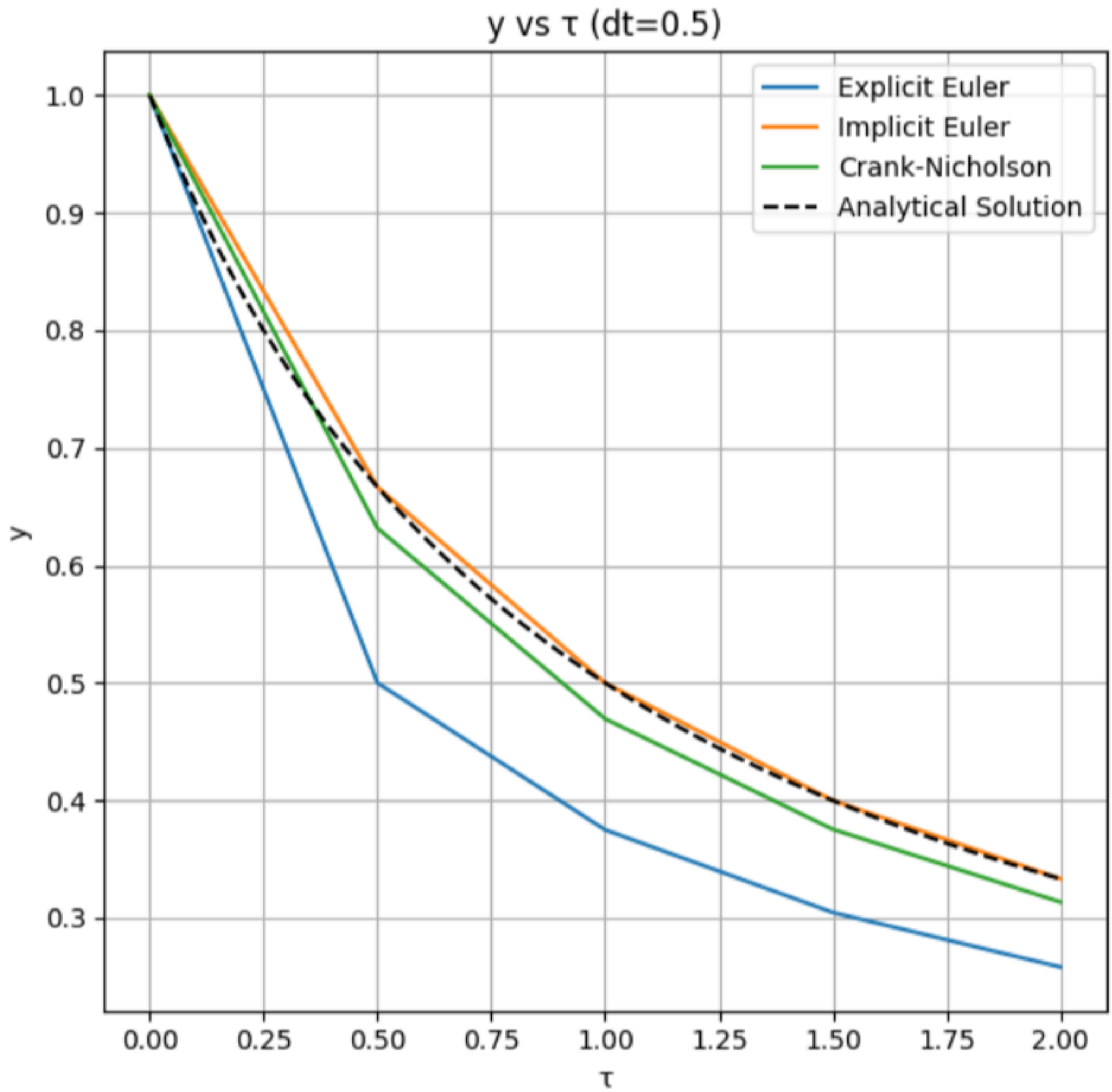
A2


y vs. τ using Implicit Euler
dt=0.1
dt=0.2
dt=0.5
dt=1.0
dt=2.0
Analytical Solution
y
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.0
0.5
1.0
1.5
2.0

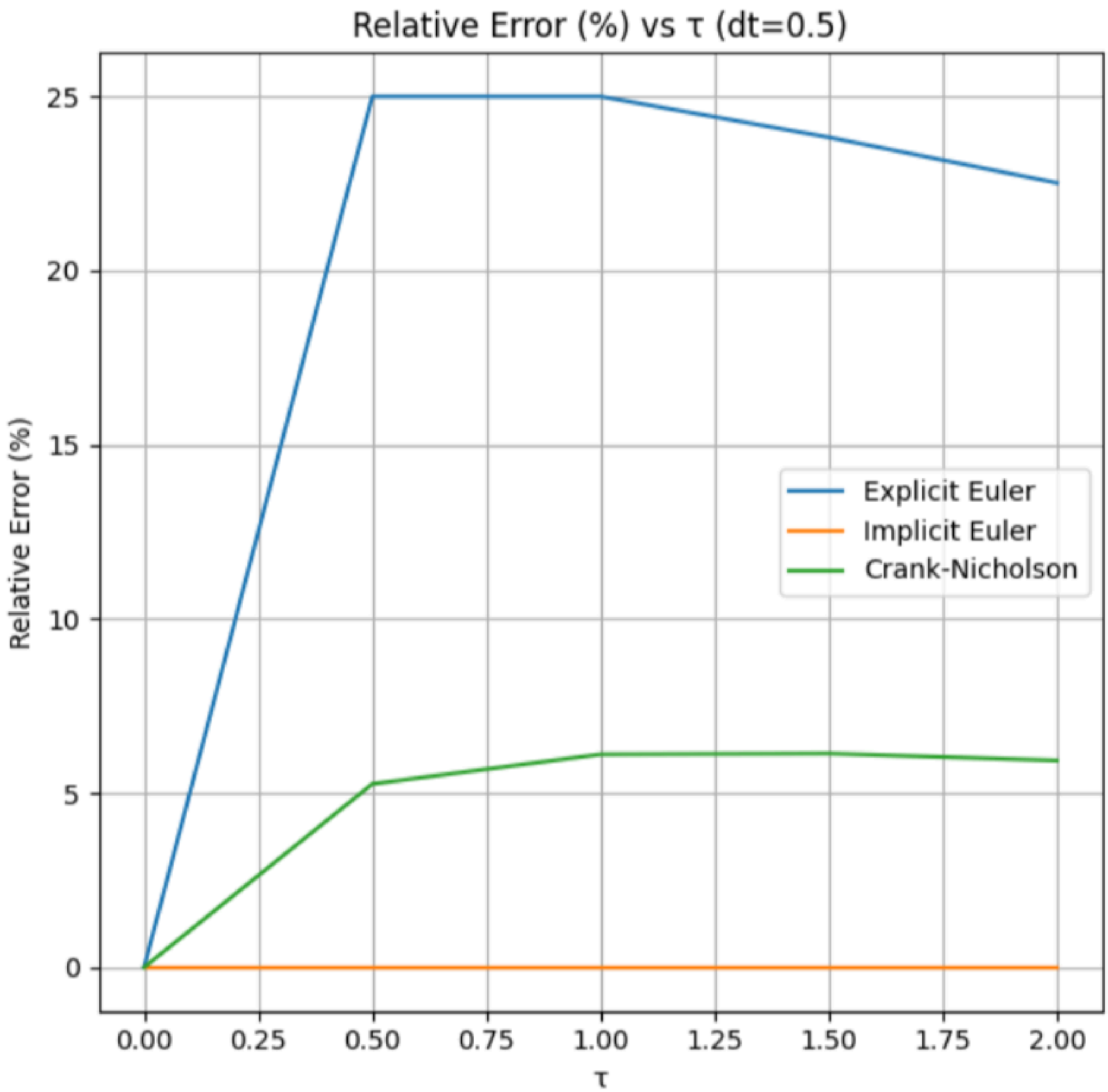
A3


y vs. τ using Crank-Nicholson
dt=0.1
dt=0.2
dt=0.5
dt=1.0
dt=2.0
Analytical Solution
y
τ
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.5
2.0

B1


y vs τ (dt=0.5)
Explicit Euler
Implicit Euler
Crank-Nicholson
Analytical Solution
y
τ
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

B2



Explanation of the Graphs:

Explicit Euler:
Error increases as dt increases due to its simple forward approximation. It means results deviate significantly from the analytical solution with time.

Implicit Euler: More stable than Explicit Euler and it is not affected by step size ie dt. However, it tends to slightly overestimate the solution, showing moderate accuracy.

Crank-Nicholson: Achieves the best balance of accuracy and stability by averaging forward and backward steps. This method closely matches the analytical solution, with minimal error across time steps.

In case of the Graph for Relative Error:
Explicit Euler shows the highest error growth, Implicit Euler has a moderate and stable error, while Crank-Nicholson displays the lowest and constant error, making it the accurate and more reliable.