Computational programming in Chemical Reaction Engineering

Name of the Team members

Arunima Bhatnagar, Bhavya Singh, Hasika Penna

INTRODUCTION

This project explores the integration of the Runge-Kutta method into computational programming for modeling dynamic chemical reactions. Recognized for its accuracy, the Runge-Kutta method offers a robust solution for solving complex ordinary differential equations. Our goal is to create a versatile computational framework that enables precise prediction of chemical reaction kinetics, facilitating optimization of process parameters and ensuring the safety and efficiency of chemical processes. Join us as we merge mathematical principles, numerical methods, and programming to empower researchers and engineers in Chemical Reaction Engineering

OBJECTIVE

The project aims to develop efficient C++ programs utilizing the Runge-Kutta method for solving chemical reactions in chemical reaction engineering. Key objectives include:

1. C++ Program Development:
   - Create modular and maintainable C++ programs using object-oriented principles.
2. Chemical Reaction Simulation:
   - Apply the Runge-Kutta method to accurately simulate chemical reactions with various kinetics and multi-species interactions.
4. Accuracy and Visualization:
   - Ensure accurate and stable numerical solutions with error control mechanisms.
   - Incorporate visualization tools for graphical representation of reaction progress.
5. Scalability and Optimization:
   - Develop scalable programs for simulations of varying complexity.
   - Implement optimization techniques for computational efficiency.
6. Documentation and Support:
   - Provide comprehensive documentation, including user manuals and code documentation.
   - Offer user support to facilitate effective program utilization.

MATERIALS AND METHODS

The Runge-Kutta method is a numerical technique used to solve ordinary differential equations (ODEs) or systems of ODEs..The method involves iterative calculations to estimate the next value of the dependent variable based on the current value and the rate of change (slope) at that point.

Here's a concise explanation of the Runge-Kutta method:

Problem Formulation:

- Given a differential equation or a system of differential equations that describes how a variable or a set of variables changes with respect to an independent variable.

–Initialization:

- Start with an initial value for the dependent variable(s).

–Iteration:

- At each iteration, calculate the slope at the current point.
- Use this slope to estimate the change in the variable over a small interval.
- Update the variable based on this estimation.

–Repeat:

- Repeat these calculations until you reach the desired endpoint or a specified number of iterations.

series.cpp > ...

```cpp
#include <iostream>
using namespace std;
float func1(float t, float a, float b, float c)
{
    float f=-a;
    return f;
}
float func2(float t, float a, float b, float c)
{
    float f=a-b;
    return f;
}
float func3(float t, float a, float b, float c)
{
    float f=b;
    return f;
}

int main()
{
    float h,x0,xf,t0,a,b,c,k1,k2,k3,k4,l1,l2,l3,l4,m1,m2,m3,m4;
    int n;
    t0=0,a=1,b=0,c=0,x0=0;
    cin>>xf>>h;
    n=(xf-x0)/h;

    for(int i=0;i<=n-1;++i)
    {
        k1=h*func1(t0,a,b,c);
        l1=h*func2(t0,a,b,c);
```

RESULTS

The following were the outputs of the C++ codes for solving chemical reaction equations by Runge Kutta method

On further analysis we see that the analytical solution and numerical solution comes out to be the same which can be showed by finding the solution through stoichiometric coefficients.

Output

```
a =4.54e-05
b =0.000454
c =0.999503
t =10.0001
```

Custom Input

```
10
```

Code for Series Reaction

```cpp
for(int i=0;i<=n-1;++i)
{
    k1=h*func1(t0,a,b,c);
    l1=h*func2(t0,a,b,c);
    m1=h*func3(t0,a,b,c);
    k2=h*func1(t0+(0.5*h),a+(0.5*k1),b+(0.5*l1),(0.5*m1));
    l2=h*func2(t0+(0.5*h),a+(0.5*k1),b+(0.5*l1),(0.5*m1));
    m2=h*func3(t0+(0.5*h),a+(0.5*k1),b+(0.5*l1),(0.5*m1));
    k3=h*func1(t0+(0.5*h),a+(0.5*k2),b+(0.5*l2),(0.5*m2));
    l3=h*func2(t0+(0.5*h),a+(0.5*k2),b+(0.5*l2),(0.5*m2));
    m3=h*func3(t0+(0.5*h),a+(0.5*k2),b+(0.5*l2),(0.5*m2));
    k4=h*func1(t0+h,a+k3,b+l3,c+m3);
    l4=h*func2(t0+h,a+k3,b+l3,c+m3);
    m4=h*func3(t0+h,a+k3,b+l3,c+m3);
    a=a+(k1+(2*k2)+(2*k3)+k4)/6;
    b=b+(l1+(2*l2)+(2*l3)+l4)/6;
    c=c+(m1+(2*m2)+(2*m3)+m4)/6;
    t0=t0+h;
}
cout<<"a ="<<a<<endl;
cout<<"b ="<<b<<endl;
cout<<"c ="<<c<<endl;
cout<<"t ="<<t0<<endl;
return 0;
```

Function for Runge Kutta

CONCLUSION

In conclusion, the project successfully achieved its objectives of developing a robust and versatile C++ program for solving chemical reactions in chemical reaction engineering using the Runge-Kutta method. The key accomplishments and findings are summarized below:

1. Efficient Program Development:
   - Implemented modular and maintainable C++ programs, adhering to object-oriented principles, providing a solid foundation for code scalability and future enhancements.
2. Chemical Reaction Simulation:
   - Successfully applied the Runge-Kutta method to accurately simulate a wide range of chemical reactions, accommodating diverse kinetics and multi-species interactions.
3. Accuracy and Visualization:
   - Ensured accurate and stable numerical solutions through the implementation of error control mechanisms, accompanied by effective graphical visualization tools for monitoring reaction progress.

REFERENCES

1.Chemical Reaction Engineering Book by Octave Levenspiel.

2.Introduction to numerical methods in Chemical engineering

Name of the Supervisor: Dr. Pradeep Ahuja

Signature of the Supervisor with date