Программная реализация

```python
k = [0.85, 0.1]
print(eiler(equations, 0, 1, [1, 0], 0.1, args=(k, )))

```

```
[[1, 0],
 [0.915, 0.085],
 [0.838075, 0.16192500000000004],
 [0.7684578750000001, 0.23154212500000004],
 [0.7054543768750001, 0.29454562312500004],
 [0.6484362110718751, 0.35156378892812506],
 [0.596834771020047, 0.403165228979953],
 [0.5501354677731425, 0.4498645322268577],
 [0.5078725983346939, 0.4921274016653062],
 [0.469624701492898, 0.5303752985071022],
 [0.4350103548510727, 0.5649896451489275]]
```

Метод Рунге-Кутты

Пусть дана следующая система обыкновенных дифференциальных уравнений:

$$\begin{cases} \dfrac{dy_1}{dx} = f_1\left(x, y_1, y_2\right) \\ \dfrac{dy_2}{dx} = f_2\left(x, y_1, y_2\right) \end{cases} \tag{6}$$

с начальными условиями:

$$\begin{gathered} y_1\big|_{x=x_0} = y_{01} \\ y_2\big|_{x=x_0} = y_{02} \end{gathered} \tag{7}$$

Метод Рунге-Кутты

При использовании метода Рунге-Кутты, расчетные формулы примут следующий вид:

$$\begin{cases} y_{i,1} = y_{(i-1),1} + h/6 \cdot (k_{1,1} + 2 \cdot k_{2,1} + 2 \cdot k_{3,1} + k_{4,1}) \\ y_{i,2} = y_{(i-1),2} + h/6 \cdot (k_{1,2} + 2 \cdot k_{2,2} + 2 \cdot k_{3,2} + k_{4,2}) \\ \quad x_i = x_{i-1} + h \end{cases} \quad (8)$$

где

$$k_{1,1} = f_1\left(x, y_{(i-1),1}, y_{(i-1),2}\right); \qquad k_{1,2} = f_2\left(x, y_{(i-1),1}, y_{(i-1),2}\right);$$

$$k_{2,1} = f_1\left(x + \frac{h}{2}, y_{(i-1),1} + k_{1,1} \cdot \frac{h}{2}, y_{(i-1),2} + k_{1,2} \cdot \frac{h}{2}\right); \qquad k_{2,2} = f_2\left(x + \frac{h}{2}, y_{(i-1),1} + k_{1,1} \cdot \frac{h}{2}, y_{(i-1),2} + k_{1,2} \cdot \frac{h}{2}\right);$$

$$k_{3,1} = f_1\left(x + \frac{h}{2}, y_{(i-1),1} + k_{2,1} \cdot \frac{h}{2}, y_{(i-1),2} + k_{2,2} \cdot \frac{h}{2}\right); \qquad k_{3,2} = f_2\left(x + \frac{h}{2}, y_{(i-1),1} + k_{2,1} \cdot \frac{h}{2}, y_{(i-1),2} + k_{2,2} \cdot \frac{h}{2}\right);$$

$$k_{4,1} = f_1\left(x + h, y_{(i-1),1} + k_{3,1} \cdot h, y_{(i-1),2} + k_{3,2} \cdot h\right); \qquad k_{4,2} = f_2\left(x + h, y_{(i-1),1} + k_{3,1} \cdot h, y_{(i-1),2} + k_{3,2} \cdot h\right). \quad (9)$$

где h – шаг интегрирования; $f_1\left(x_i, y_{(i-1),1}, y_{(i-1),2}\right)$ и $f_2\left(x_i, y_{(i-1),1}, y_{(i-1),2}\right)$ – правые части дифференциальных уравнений, $k_{1,j}$, $k_{2,j}$, $k_{3,j}$, $k_{4,j}$ – параметры метода Рунге-Кутты для j-го уравнения.

Пример 1

Решим методом Рунге-Кутты пример, приведенный на слайде **??**. Воспользуемся формулами (8), (9) и запишем выражения для нахождения значений искомых переменных $y_{i,1}$ и $y_{i,2}$:

$$
\begin{aligned}
&k_{1,1} = y_{(i-1),2}; && k_{1,2} = \exp\left(-x_i \cdot y_{(i-1),1}\right);\\
&k_{2,1} = y_{(i-1),2} + k_{1,2} \cdot \frac{h}{2}; && k_{2,2} = \exp\left[-\left(x_i + \frac{h}{2}\right) \cdot \left(y_{(i-1),1} + k_{1,1} \cdot \frac{h}{2}\right)\right]\\
&k_{3,1} = y_{(i-1),2} + k_{2,2} \cdot \frac{h}{2}; && k_{3,2} = \exp\left[-\left(x_i + \frac{h}{2}\right) \cdot \left(y_{(i-1),1} + k_{2,1} \cdot \frac{h}{2}\right)\right]\\
&k_{4,1} = y_{(i-1),2} + k_{3,2} \cdot h; && k_{4,2} = \exp\left[-\left(x_i + h\right) \cdot \left(y_{(i-1),1} + k_{3,1} \cdot h\right)\right]
\end{aligned}
$$

$$
\begin{cases}
y_{i,1} = y_{(i-1),1} + \dfrac{0.1}{6} \cdot (k_{1,1} + 2 \cdot k_{2,1} + 2 \cdot k_{3,1} + k_{4,1})\\
y_{i,2} = y_{(i-1),2} + \dfrac{0.1}{6} \cdot (k_{1,2} + 2 \cdot k_{2,2} + 2 \cdot k_{3,2} + k_{4,2})\\
\quad x_i = x_{i-1} + 0.1
\end{cases}
$$

Пример 1

Результаты вычислений сведем в таблице.

i	x_i	$k_{1,1}$	$k_{2,1}$	$k_{3,1}$	$k_{4,1}$	$y_{i,1}$	$k_{1,2}$	$k_{2,2}$	$k_{3,2}$	$k_{4,2}$	$y_{i,2}$
0	0.0	–	–	–	–	0.0000	–	–	–	–	0.0000
1	0.1	0.0000	0.0500	0.0500	0.1000	0.0050	1.0000	1.0000	0.9999	0.9995	0.1000
2	0.2	0.1000	0.1500	0.1499	0.1998	0.0200	0.9995	0.9985	0.9981	0.9960	0.1998
3	0.3	0.1998	0.2496	0.2494	0.2990	0.0449	0.9960	0.9925	0.9919	0.9866	0.2990
4	0.4	0.2990	0.3483	0.3480	0.3968	0.0797	0.9866	0.9793	0.9784	0.9686	0.3968
5	0.5	0.3968	0.4453	0.4446	0.4923	0.1242	0.9686	0.9562	0.9551	0.9398	0.4924
6	0.6	0.4924	0.5393	0.5384	0.5844	0.1781	0.9398	0.9214	0.9202	0.8987	0.5844
7	0.7	0.5844	0.6293	0.6281	0.6716	0.2409	0.8987	0.8739	0.8727	0.8448	0.6717
8	0.8	0.6717	0.7139	0.7124	0.7529	0.3122	0.8448	0.8139	0.8126	0.7790	0.7529
9	0.9	0.7529	0.7919	0.7901	0.8271	0.3913	0.7790	0.7427	0.7415	0.7032	0.8271
10	1.0	0.8271	0.8623	0.8603	0.8933	0.4774	0.7032	0.6630	0.6619	0.6204	0.8933

Программная реализация


```python
import numpy as np


def rk(func, x0, xf, y0, h):
    count = int((xf - x0) / h) + 1
    y = [y0[:]]
    x = x0
    for i in range(1, count):
        k1 = func(x, y[i-1])
        k2 = func(x + h / 2, [y + k * h / 2 for y, k in zip(y[i-1], k1)])
        k3 = func(x + h / 2, [y + k * h / 2 for y, k in zip(y[i-1], k2)])
        k4 = func(x + h, [y + k * h for y, k in zip(y[i-1], k3)])
        y.append([])
        for j in range(len(y0)):
            y[i].append(
                y[i-1][j] + h / 6 * (k1[j] + 2 * k2[j] + 2 * k3[j] + k4[j])
            )
        x += h
    return y
```

Программная реализация

```python


def equations(x, y):  # Функция, содержащая правые части дифф. уравнений
    return [y[1], np.exp(-x * y[0])]


if __name__ == '__main__':
    print(rk(equations, 0, 1, [0, 0], 0.1))

```

```
[[0, 0],
 [0.004999791679686959, 0.09998750234339197],
 [0.019992089353337197, 0.19980027824237273],
 [0.0449395453295417 8, 0.2989921821997826],
 [0.0797458927313852 2, 0.3968347761839209 3],
 [0.124229226130722 7, 0.4923515428080233 5],
 [0.178100008129217 4, 0.5843789596377397],
 [0.240946624321046 96, 0.67165612248553],
 [0.312231135461859 6, 0.7529375201538153],
 [0.391296952548 54, 0.827116006499604 7],
 [0.4773885589403407, 0.8933374434985747]]
```

Пример 2

Рассмотрим также решение примера, приведенного на слайде **??**, методом Рунге-Кутты. Воспользуемся формулами (8), (9) и запишем выражения для нахождения значений искомых концентраций компонентов $C_{A,i}$ и $C_{B,i}$:

$$\begin{cases} C_{A,i} = C_{A,(i-1)} + \dfrac{0.1}{6} \cdot (k_{1,1} + 2 \cdot k_{2,1} + 2 \cdot k_{3,1} + k_{4,1}) \\ C_{B,i} = C_{B,(i-1)} + \dfrac{0.1}{6} \cdot (k_{1,2} + 2 \cdot k_{2,2} + 2 \cdot k_{3,2} + k_{4,2}) \\ \quad t_i = t_{i-1} + 0.1 \end{cases}$$

$$k_{1,1} = -k_1 C_{A,(i-1)} + k_2 C_{B,(i-1)};$$

$$k_{2,1} = -k_1 \left(C_{A,(i-1)} + k_{1,1} \frac{h}{2} \right) + k_2 \left(C_{B,(i-1)} + k_{1,2} \cdot \frac{h}{2} \right);$$

$$k_{3,1} = -k_1 \left(C_{A,(i-1)} + k_{2,1} \frac{h}{2} \right) + k_2 \left(C_{B,(i-1)} + k_{2,2} \frac{h}{2} \right);$$

$$k_{4,1} = -k_1 \left(C_{A,(i-1)} + k_{3,1} h \right) + k_2 \left(C_{B,(i-1)} + k_{3,2} h \right);$$

$$k_{1,2} = k_1 C_{A,(i-1)} - k_2 C_{B,(i-1)};$$

$$k_{2,2} = k_1 \left(C_{A,(i-1)} + k_{1,1} \frac{h}{2} \right) - k_2 \left(C_{B,(i-1)} + k_{1,2} \frac{h}{2} \right);$$

$$k_{3,2} = k_1 \left(C_{A,(i-1)} + k_{2,1} \frac{h}{2} \right) - k_2 \left(C_{B,(i-1)} + k_{2,2} \frac{h}{2} \right);$$

$$k_{4,2} = k_1 \left(C_{A,(i-1)} + k_{3,1} h \right) - k_2 \left(C_{B,(i-1)} + k_{3,2} h \right)$$

Пример 2

Результаты вычислений сведем в таблице.

i	t_i	$k_{1,1}$	$k_{2,1}$	$k_{3,1}$	$k_{4,1}$	$C_{A,i}$	$k_{1,2}$	$k_{2,2}$	$k_{3,2}$	$k_{4,2}$	$C_{B,i}$
0	0.0	–	–	–	–	1.0000	–	–	–	–	0.0000
1	0.1	−0.8500	−0.8096	−0.8115	−0.7729	0.9189	0.8500	0.8096	0.8115	0.7729	0.0811
2	0.2	−0.7730	−0.7363	−0.7380	−0.7029	0.8452	0.7730	0.7363	0.7380	0.7029	0.1548
3	0.3	−0.7029	−0.6695	−0.6711	−0.6392	0.7781	0.7029	0.6695	0.6711	0.6392	0.2219
4	0.4	−0.6392	−0.6088	−0.6103	−0.5812	0.7171	0.6392	0.6088	0.6103	0.5812	0.2829
5	0.5	−0.5813	−0.5537	−0.5550	−0.5286	0.6617	0.5813	0.5537	0.5550	0.5286	0.3383
6	0.6	−0.5286	−0.5035	−0.5047	−0.4807	0.6113	0.5286	0.5035	0.5047	0.4807	0.3887
7	0.7	−0.4807	−0.4579	−0.4589	−0.4371	0.5654	0.4807	0.4579	0.4589	0.4371	0.4346
8	0.8	−0.4371	−0.4164	−0.4174	−0.3975	0.5237	0.4371	0.4164	0.4174	0.3975	0.4763
9	0.9	−0.3975	−0.3786	−0.3795	−0.3615	0.4858	0.3975	0.3786	0.3795	0.3615	0.5142
10	1.0	−0.3615	−0.3443	−0.3451	−0.3287	0.4513	0.3615	0.3443	0.3451	0.3287	0.5487

Программная реализация


```python
def equations(t, c, k):  # Функция, содержащая правые части дифф. уравнений
    right_parts = [-k[0] * c[0] + k[1] * c[1],
                   k[0] * c[0] - k[1] * c[1]]
    return right_parts

def rk(func, x0, xf, y0, h, args=()):
    count = int((xf - x0) / h) + 1
    y = [y0[:]]
    x = x0
    for i in range(1, count):
        k1 = func(x, y[i-1], *args)
        k2 = func(x + h / 2, [y + k * h / 2 for y, k in zip(y[i-1], k1)], *args)
        k3 = func(x + h / 2, [y + k * h / 2 for y, k in zip(y[i-1], k2)], *args)
        k4 = func(x + h, [y + k * h for y, k in zip(y[i-1], k3)], *args)
        y.append([])
        for j in range(len(y0)):
            y[i].append(y[i-1][j] + h / 6 * (k1[j] + 2 * k2[j] + 2 * k3[j] + k4[j]))
        x += h
    return y
```

Программная реализация

```


if __name__ == '__main__':
    k = [0.85, 0.1]
    print(rk(equations, 0, 1, [1, 0], 0.1, args=(k, )))

```

```
[[1, 0],
 [0.9189126823697916, 0.08108731763020834],
 [0.8451740652412765, 0.15482593475872353],
 [0.7781181579189691, 0.22188184208103093],
 [0.717139326447514, 0.282860673552486],
 [0.6616868236612737, 0.33831317633872626],
 [0.6112598149591241, 0.388740185040876],
 [0.5654028548783672, 0.4345971451216329],
 [0.5237017736131901, 0.4762982263868099],
 [0.4857799363256236, 0.5142200636743764],
 [0.4512948414639336, 0.5487051585360664]]
```

Графическая визуализация

Построим графическую визуализацию полученного решения:

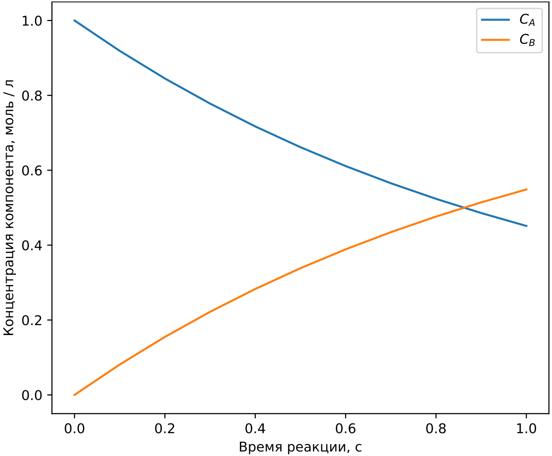

Рисунок 1 – Изменение концентрации реагирующих веществ во времени

Расчет схемы химических реакций

Рассмотрим следующую схему химических реакций:

$$A \longleftrightarrow 2B \longleftrightarrow C$$

с константами скоростей k_1, k_2, k_3 и k_4. Уравнения, описывающие скорость изменения концентраций компонентов по времени, записываются следующим образом:

1. $A \longrightarrow 2B \qquad r_1 = k_1 \cdot C_A$
2. $2B \longrightarrow A \qquad r_2 = k_2 \cdot C_B^2$
3. $2B \longrightarrow C \qquad r_3 = k_3 \cdot C_B^2$
4. $C \longrightarrow 2B \qquad r_4 = k_4 \cdot C_C$

$$\begin{cases} \dfrac{dC_A}{dt} = -r_1 + r_2 \\ \dfrac{dC_B}{dt} = 2 \cdot (r_1 - r_2 - r_3 + r_4) \\ \dfrac{dC_C}{dt} = r_3 - r_4 \end{cases}$$

Метод Эйлера


```python
import numpy as np


def func(time: float, c: np.ndarray,
         k: np.ndarray) -> np.ndarray:
    ca, cb, cc = c
    k1, k2, k3, k4 = k
    r1, r2, r3, r4 = [
        k1 * ca,
        k2 * cb ** 2,
        k3 * cb ** 2,
        k4 * cc,
    ]
    dca_dt = -r1 + r2
    dcb_dt = 2 * (r1 - r2 - r3 + r4)
    dcc_dt = r3 - r4

    return dca_dt, dcb_dt, dcc_dt
```

Метод Эйлера


```python
def eiler(func, x0, xf, y0, h, args=()):
    count = int((xf - x0) / h) + 1
    y = np.zeros((count, y0.shape[0]))
    x, y[0] = x0, y0[:].copy()
    for i in range(1, count):
        right_parts = func(x, y[i-1], *args)
        for j in range(len(y0)):
            y[i][j] = y[i-1][j] + h * right_parts[j]
        x += h
    return y


if __name__ == '__main__':
    k, y0 = np.array([.2, .1, .1, .1]), np.array([1, .5, .2])
    t0, tf = 0, 10
    y_eiler = eiler(func, t0, tf, y0, 0.1, args=(k, ))
```

Метод Рунге-Кутты


```python
import numpy as np


def func(time: float, c: np.ndarray,
         k: np.ndarray) -> np.ndarray:
    ca, cb, cc = c
    k1, k2, k3, k4 = k
    r1, r2, r3, r4 = [
        k1 * ca,
        k2 * cb ** 2,
        k3 * cb ** 2,
        k4 * cc,
    ]
    dca_dt = -r1 + r2
    dcb_dt = 2 * (r1 - r2 - r3 + r4)
    dcc_dt = r3 - r4

    return dca_dt, dcb_dt, dcc_dt
```

Метод Рунге-Кутты


```python
def rk(func, x0, xf, y0, h, args=()):
    count = int((xf - x0) / h) + 1
    y = np.zeros((count, y0.shape[0]))
    x, y[0] = x0, y0[:].copy()
    for i in range(1, count):
        k1 = func(x, y[i-1], *args)
        k2 = func(x + h / 2, [y + k * h / 2 for y, k in zip(y[i-1], k1)], *args)
        k3 = func(x + h / 2, [y + k * h / 2 for y, k in zip(y[i-1], k2)], *args)
        k4 = func(x + h, [y + k * h for y, k in zip(y[i-1], k3)], *args)
        for j in range(len(y0)):
            y[i][j] = y[i-1][j] + h / 6 * (k1[j] + 2 * k2[j] + 2 * k3[j] + k4[j])
        x += h
    return y


if __name__ == '__main__':
    k, y0 = np.array([.2, .1, .1, .1]), np.array([1, .5, .2])
    t0, tf = 0, 10
    y_rk = rk(func, t0, tf, y0, 0.1, args=(k, ))
```