Лабораторная работа №1 по численным методам

Численное решение начальной краевой задачи для дифф. уравнения параболического типа

Син Д.Д. М8О-407Б-18

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
# Граничные условия
def phi0(t: float, a: float, x=0, ):
    return np.exp(-a * t)


def phil(t: float, a: float, x=np.pi):
    return -np.exp(-a * t)


# Начальные условия
def psi(x: float, t=0):
    return np.sin(x)


# Аналитическое решение
def U(x: float, t: float, a: float) -> float:
    return np.exp(-a * t) * np.sin(x)
```

In [3]:
```
# Метод прогонки
def equation_solve(a, b, c, d):
    size = len(a)
    p = np.zeros(size)
    q = np.zeros(size)
    p[0] = -c[0] / b[0]
    q[0] = d[0] / b[0]
    for i in range(1, size):
        p[i] = -c[i] / (b[i] + a[i] * p[i - 1])
        q[i] = (d[i] - a[i] * q[i - 1]) / (b[i] + a[i] * p[i - 1])
    x = np.zeros(size)
    x[-1] = q[-1]
    for i in range(size - 2, -1, -1):
        x[i] = p[i] * x[i + 1] + q[i]
    return x
```

In [4]:

```
def explicit_schema(a: float, n: int, tc: int, tau: float, x_min: float, x_
    """
    Явная конечно-разностная схема
    :param a: коэффициент температуропровдности
    :param n: количество точек в пространстве
    :param tc: количество временных точек
    :param tau: временной шаг
    :param x_min: левая граница
    :param x_max: правая граница
    :param al: тип апроксимации
        al = 1: двухточечная аппроксимация с первым порядком
        al = 2: трехточечная аппроксимация со вторым порядком
        al = 3: двухточечная аппроксимация со вторым порядком
    :return: сеточную функцию
    """
    h = (x_max - x_min) / n
    sigma = a ** 2 * tau / h ** 2
    if sigma > 0.5:
        raise Exception(f'Явная схема не устойчива sigma = {sigma}')
    u = np.zeros((tc, n))
    for j in range(1, n - 1):
        u[0][j] = psi(x_min + j * h)
    for k in range(1, tc):
        for j in range(1, n - 1):
            u[k][j] = sigma * (u[k - 1][j + 1] + u[k - 1][j - 1]) + (1 - 2
        if al == 1:
            u[k][0] = u[k][1] - h * phi0(k * tau, a)
            u[k][-1] = u[k][-2] + h * phil(k * tau, a)
        elif al == 2:
            u[k][0] = (phi0(k * tau, a) + u[k][2] / (2 * h) - 2 * u[k][1] /
            u[k][-1] = (phil(k * tau, a) - u[k][-3] / (2 * h) + 2 * u[k][-2
        elif al == 3:
            u[k][0] = (u[k][1] - h * phi0(k * tau, a) + (h ** 2 / (2 * tau)
            u[k][-1] = (u[k][-2] + h * phil(k * tau, a) + h ** 2 / (2 * tau
        else:
            raise Exception('Такого типа апроксимации граничных условий не
    return u
```

In [5]:

```
def explicit_implicit_schema(ap: float, n: int, tc: int, tau: float, x_min:
    """
    Явно-неявная конечно-разностная схема
    :param a: коэффициент температуропровдности
    :param n: количество точек в пространстве
    :param tc: количество временных точек
    :param tau: временной шаг
    :param x_min: левая граница
    :param x_max: правая граница
    :param al: тип апроксимации
        al = 1: двухточечная аппроксимация с первым порядком
        al = 2: трехточечная аппроксимация со вторым порядком
        al = 3: двухточечная аппроксимация со вторым порядком
    :param eta: коэффициет
    :return: сеточную функцию
    """
```

```
    u = np.zeros((tc, n))

    h = (x_max - x_min) / n
    sigma = ap ** 2 * tau / h ** 2

    for i in range(1, n - 1):
        u[0][i] = psi(x_min + i * h)

    for k in range(1, tc):
        a = np.zeros(n)
        b = np.zeros(n)
        c = np.zeros(n)
        d = np.zeros(n)
        for j in range(1, n - 1):
            a[j] = sigma
            b[j] = -(1 + 2 * sigma)
            c[j] = sigma
            d[j] = -u[k - 1][j]

        # Аппроксимация граничных условий неявной схемы
        if al == 1:
            b[0] = -1 / h
            c[0] = 1 / h
            d[0] = phi0((k + 1) * tau, ap)
            a[-1] = -1 / h
            a[-1] = 1 / h
            d[-1] = phil((k + 1) * tau, ap)
        elif al == 2:
            k0 = 1 / (2 * h) / c[1]
            b[0] = (-3 / (2 * h) + a[1] * k0)
            c[0] = 2 / h + b[1] * k0
            d[0] = phi0((k + 1) * tau, ap) + d[1] * k0
            k1 = -(1 / (h * 2)) / a[-2]
            a[-1] = (-2 / h) + b[-2] * k1
            b[-1] = (3 / (h * 2)) + c[-2] * k1
            d[-1] = phil((k + 1) * tau, ap) + d[-2] * k1
        elif al == 3:
            b[0] = 2 * ap ** 2 / h + h / tau
            c[0] = - 2 * ap ** 2 / h
            d[0] = (h / tau) * u[k - 1][0] - phi0((k + 1) * tau, ap) * 2 *
            a[-1] = -2 * ap ** 2 / h
            b[-1] = 2 * ap ** 2 / h + h / tau
            d[-1] = (h / tau) * u[k - 1][-1] + phil((k + 1) * tau, ap) * 2
        else:
            raise Exception('Такого типа апроксимации граничных условий не

        # Решение неявной схемой
        u[k] = eta * equation_solve(a, b, c, d)

        # Решение явной схемой
        explicit_part = np.zeros(n)

        # Аппроксимация граничных условий явной схемы
        for j in range(1, n - 1):
            explicit_part[j] = (sigma * (u[k - 1][j + 1] + u[k - 1][j - 1])
        if al == 1:
            explicit_part[0] = (explicit_part[1] - h * phi0(k * tau, ap))
```

```
            explicit_part[-1] = (explicit_part[-2] + h * phil(k * tau, ap))
        elif al == 2:
            explicit_part[0] = ((phi0(k * tau, ap) + explicit_part[2] / (2
            explicit_part[-1] = ((phil(k * tau, ap) - explicit_part[-3] / (
        elif al == 3:
            explicit_part[0] = (explicit_part[1] - h * phi0(k * tau, ap) +
                    1 + h ** 2 / (2 * tau))
            explicit_part[-1] = (explicit_part[-2] + h * phil(k * tau, ap)
                    1 + h ** 2 / (2 * tau))
        else:
            raise Exception('Такого типа апроксимации граничных условий не

        u[k] += (1 - eta) * explicit_part
    return u
```

In [6]:

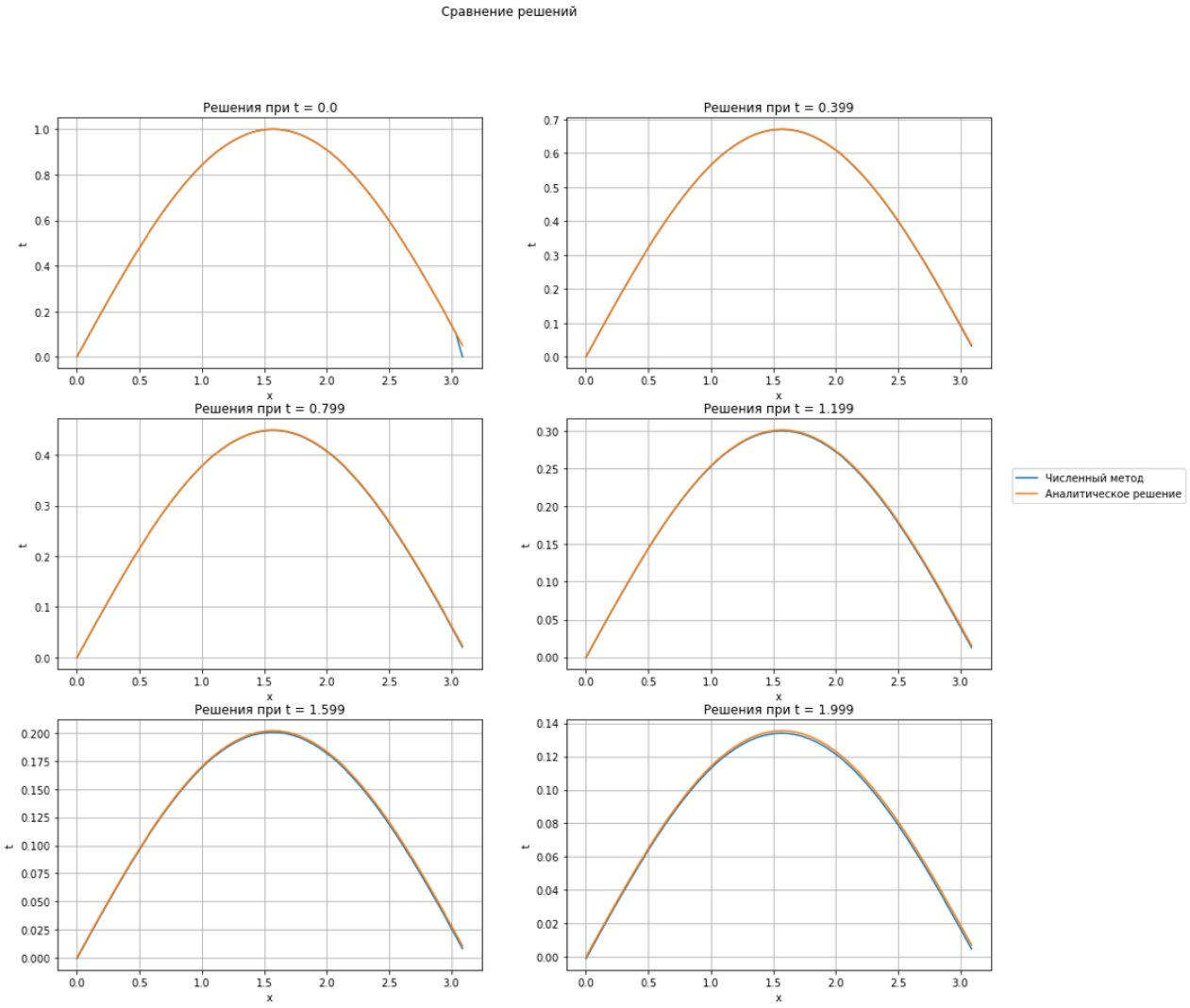
```
def draw_results(tc, x_max, x_min, u, a):
    """
    Построение графиков
    :param tc: количество временных точек
    :param x_max: правая граница
    :param x_min: левая граница
    :param u: сеточная функция
    :param a: коэффициент температуропровдности
    """
    times = np.zeros(tc)
    for i in range(tc):
        times[i] = tau * i
    space = np.zeros(n)
    step = (x_max - x_min) / n
    for i in range(n):
        space[i] = x_min + i * step

    times_idx = np.linspace(0, times.shape[0] - 1, 6, dtype=np.int32)
    fig, ax = plt.subplots(3, 2)
    fig.suptitle('Сравнение решений')
    fig.set_figheight(15)
    fig.set_figwidth(16)
    k = 0
    for i in range(3):
        for j in range(2):
            time_idx = times_idx[k]
            ax[i][j].plot(space, u[time_idx], label='Численный метод')
            ax[i][j].plot(space, [U(x, times[time_idx], a) for x in space],
            ax[i][j].grid(True)
            ax[i][j].set_xlabel('x')
            ax[i][j].set_ylabel('t')
            ax[i][j].set_title(f'Решения при t = {times[time_idx]}')
            k += 1
    plt.legend(bbox_to_anchor=(1.05, 2), loc='upper left', borderaxespad=0.
    error = np.zeros(tc)
    for i in range(tc):
        error[i] = np.max(np.abs(u[i] - np.array([U(x, times[i], a) for x i
    plt.figure(figsize=(12, 7))
    plt.plot(times[1:], error[1:], 'violet', label='Ошибка')
    plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', borderaxespad=0.
    plt.title('График изменения ошибки во времени')
    plt.xlabel('t')
    plt.ylabel('error')
    plt.grid(True)
    plt.show()
```

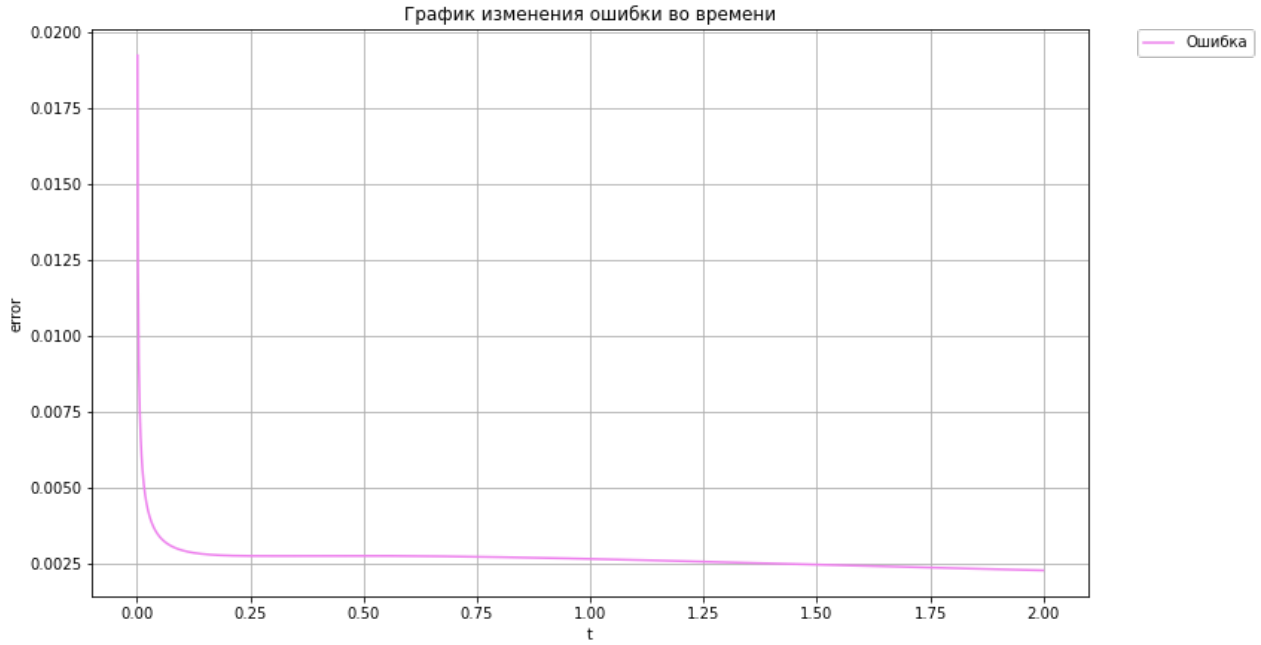
Явная конечно-разностная схема

In [7]:
```
a = 1
n = 60
tc = 2000
tau = 0.001
x_min, x_max = 0, np.pi
u = explicit_schema(a=a, n=n, tc=tc, tau=tau, x_min=x_min, x_max=x_max, al=
u
```

Out[7]:
```
array([[ 0.00000000e+00,  5.23359562e-02,  1.04528463e-01, ...,
         1.56434465e-01,  1.04528463e-01,  0.00000000e+00],
       [-2.39116110e-05,  5.22836322e-02,  1.04423959e-01, ...,
         1.56278066e-01,  8.53340910e-02,  3.30265471e-02],
       [-3.26238105e-05,  5.22226386e-02,  1.04319559e-01, ...,
         1.49158675e-01,  9.21318460e-02,  3.98765836e-02],
       ...,
       [-1.07705291e-03,  6.03037628e-03,  1.31171653e-02, ...,
         1.90090185e-02,  1.19395622e-02,  4.83213299e-03],
       [-1.07747767e-03,  6.02284765e-03,  1.31025528e-02, ...,
         1.89881024e-02,  1.19257114e-02,  4.82538604e-03],
       [-1.07790223e-03,  6.01532631e-03,  1.30879547e-02, ...,
         1.89672074e-02,  1.19118746e-02,  4.81864605e-03]])
```

In [8]:
```
draw_results(tc, x_max, x_min, u, a)
```



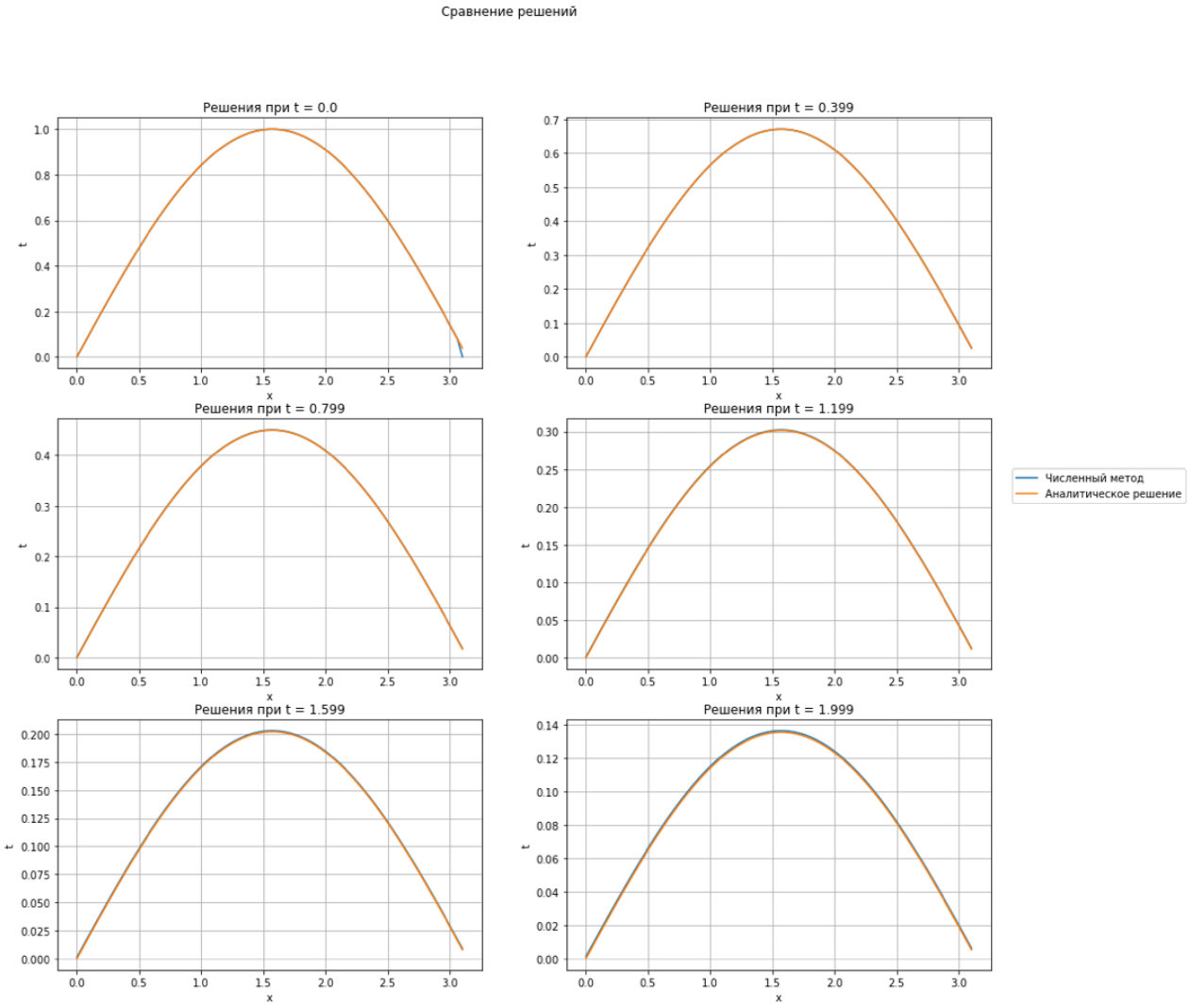


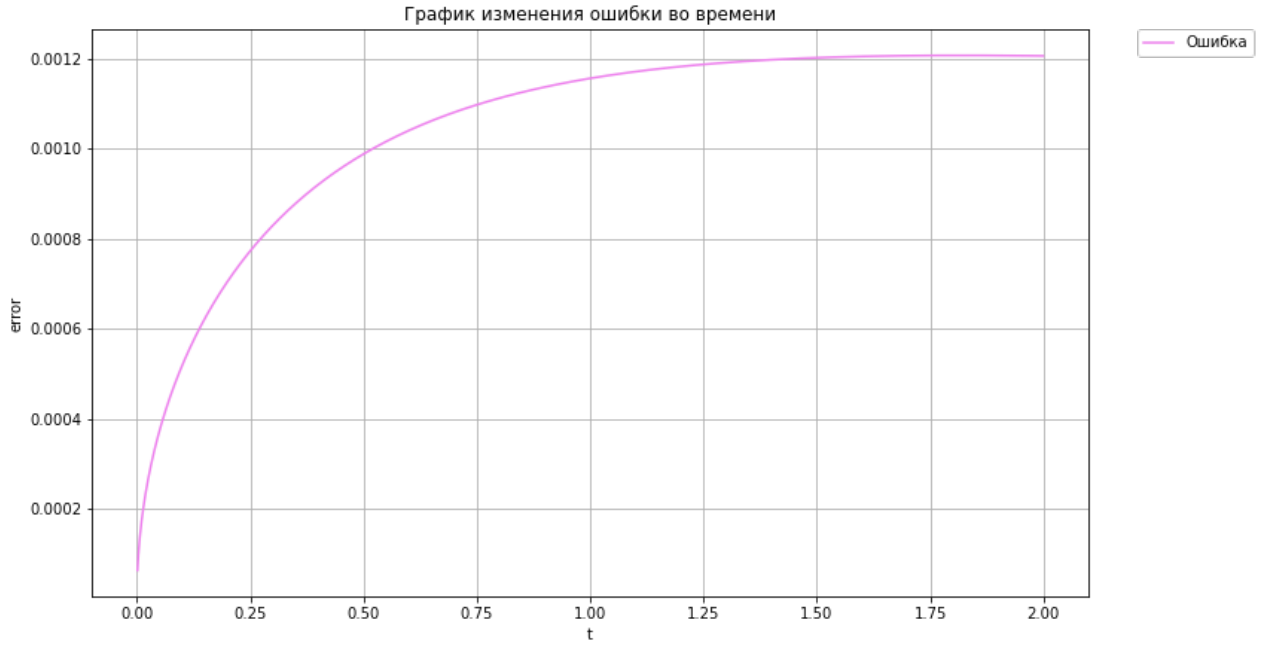
Неявная конечно-разностная схема

In [9]:
```
a = 1
n = 80
tc = 2000
tau = 0.001
x_min, x_max = 0, np.pi
u = explicit_implicit_schema(ap=a, n=n, tc=tc, tau=tau, x_min=x_min, x_max=
u
```

Out[9]:
```
array([[0.00000000e+00, 3.92598158e-02, 7.84590957e-02, ...,
        1.17537397e-01, 7.84590957e-02, 0.00000000e+00],
       [6.39174674e-05, 3.92403718e-02, 7.83868410e-02, ...,
        1.17422990e-01, 7.83904149e-02, 3.92519253e-02],
       [8.28567838e-05, 3.92174834e-02, 7.83168131e-02, ...,
        1.17309754e-01, 7.83201092e-02, 3.92220443e-02],
       ...,
       [1.20613495e-03, 6.52726497e-03, 1.18401670e-02, ...,
        1.68642141e-02, 1.15632566e-02, 6.24610818e-03],
       [1.20612328e-03, 6.52193477e-03, 1.18295264e-02, ...,
        1.68484273e-02, 1.15527680e-02, 6.24093401e-03],
       [1.20611155e-03, 6.51660984e-03, 1.18188964e-02, ...,
        1.68326563e-02, 1.15422900e-02, 6.23576506e-03]])
```

In [10]:
```
draw_results(tc, x_max, x_min, u, a)
```





Явно-неявная конечно-разностная схема

In [11]:
```
a = 1
n = 80
tc = 2000
tau = 0.0001
x_min, x_max = 0, np.pi
u = explicit_implicit_schema(ap=a, n=n, tc=tc, tau=tau, x_min=x_min, x_max=
u
```

Out[11]:
```
array([[0.00000000e+00, 3.92598158e-02, 7.84590957e-02, ...,
        1.17537397e-01, 7.84590957e-02, 0.00000000e+00],
       [1.54159003e-05, 3.92563910e-02, 7.84512800e-02, ...,
        1.17525638e-01, 7.71782099e-02, 3.75530550e-02],
       [1.66756386e-05, 3.92534257e-02, 7.84434919e-02, ...,
        1.17437383e-01, 7.72224944e-02, 3.76441370e-02],
       ...,
       [2.67423917e-04, 3.24023127e-02, 6.44879384e-02, ...,
        9.61051009e-02, 6.40946982e-02, 3.19848837e-02],
       [2.67477077e-04, 3.23991524e-02, 6.44815693e-02, ...,
        9.60954878e-02, 6.40882862e-02, 3.19816825e-02],
       [2.67530215e-04, 3.23959924e-02, 6.44752008e-02, ...,
        9.60858756e-02, 6.40818747e-02, 3.19784817e-02]])
```

In [12]:
```
draw_results(tc, x_max, x_min, u, a)
```

Сравнение решений