Вычислительный практикум (9-ый семестр, 2021)

Дифференциальные уравнения эллиптического типа

Руководитель: А. Г. Доронина *Выполнил: П. Л. Соболев*

Задачи

- Создать алгоритм решения дифференциального уравнения эллиптического типа при заданном виде дифференциального оператора и протестировать его на предложенном решении.

Теория

Постановка задачи: найти решение $u(x,t)$ проблемы

$$-Lu = f(x,t),\ (x,y) \in G, \tag{1}$$

$$u = \mu(x,y),\ (x,y) \in \Gamma, \tag{2}$$

где $G \cup \Gamma = \left\{0 \leqslant x \leqslant l_x,\ 0 \leqslant y \leqslant l_y\right\}$, а

$$Lu = \frac{\partial}{\partial x}\left(p(x,y)\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(q(x,y)\frac{\partial u}{\partial y}\right). \tag{3}$$

Здесь $p(x,y)$ и $q(x,y)$ — достаточно гладкие функции, такие, что
$0 \le c_1 \leqslant p(x,y) \leqslant c_2$, $0 \le d_1 \leqslant q(x,y) \leqslant d_2$, где c_1, c_2, d_1, d_2 — постоянные.

Разностный аналог дифференциального оператора (3) на равномерной сетке $\overline{\omega}_{h_x h_y} = \left\{(x_i = h_x i, y_j = h_y j),\ h_x = 1/N,\ h_y = 1/M,\ i = \overline{0,N},\ j = \overline{0,M}\right\}$:

$$\begin{aligned} L_h u_{i,j} = p_{i+\frac{1}{2},j}\frac{u_{i+1,j}-u_{i,j}}{h_x^2} - p_{i-\frac{1}{2},j}\frac{u_{i,j}-u_{i-1,j}}{h_x^2} + \\ + q_{i,j+\frac{1}{2}}\frac{u_{i,j+1}-u_{i,j}}{h_y^2} - q_{i,j-\frac{1}{2}}\frac{u_{i,j}-u_{i,j-1}}{h_y^2} \end{aligned} \tag{4}$$

Решение системы (1)–(2) сводится к решению линейной системы $AU = F$ путем итераций (m – оптимальное число итераций, ε – требуемая точность) над внутренними узлами матрицы U. В качестве нулевой итерации берется матрица U_0, такая, что её значения в граничных узлах ($\gamma_{h_x h_y}$) совпадают со значениями функции $\mu(x,y)$, а значения во внутренние узлах ($\omega_{h_x h_y}$) равны нулю.

Рассмотренные методы:

- Метод простой итерации

$$u_{i,j}^k = \frac{\frac{p_{i-\frac{1}{2},j}u_{i-1,j}^{k-1}}{h_x^2} + \frac{p_{i+\frac{1}{2},j}u_{i+1,j}^{k-1}}{h_x^2} + \frac{q_{i,j-\frac{1}{2}}u_{i,j-1}^{k-1}}{h_y^2} + \frac{q_{i,j+\frac{1}{2}}u_{i,j+1}^{k-1}}{h_y^2} + f_{i,j}}{\frac{p_{i-\frac{1}{2},j}}{h_x^2} + \frac{p_{i+\frac{1}{2},j}}{h_x^2} + \frac{q_{i,j-\frac{1}{2}}}{h_y^2} + \frac{q_{i,j+\frac{1}{2}}}{h_y^2}}, \tag{5}$$

$$\delta = c_1 \frac{4}{h_x^2} \sin^2 \frac{\pi h_x}{2 l_x} + d_1 \frac{4}{h_y^2} \sin^2 \frac{\pi h_y}{2 l_y}, \tag{6}$$

$$\Delta = c_2 \frac{4}{h_x^2} \cos^2 \frac{\pi h_x}{2 l_x} + d_2 \frac{4}{h_y^2} \cos^2 \frac{\pi h_y}{2 l_y}, \tag{7}$$

$$\xi = \delta / \Delta, \tag{8}$$

$$m \geqslant \frac{\ln(1/\varepsilon)}{2\xi}. \tag{9}$$

- Метод Зейделя

$$u_{i,j}^k = \frac{\frac{p_{i-\frac{1}{2},j} u_{i-1,j}^k}{h_x^2} + \frac{p_{i+\frac{1}{2},j} u_{i+1,j}^{k-1}}{h_x^2} + \frac{q_{i,j-\frac{1}{2}} u_{i,j-1}^k}{h_y^2} + \frac{q_{i,j+\frac{1}{2}} u_{i,j+1}^{k-1}}{h_y^2} + f_{i,j}}{\frac{p_{i-\frac{1}{2},j}}{h_x^2} + \frac{p_{i+\frac{1}{2},j}}{h_x^2} + \frac{q_{i,j-\frac{1}{2}}}{h_y^2} + \frac{q_{i,j+\frac{1}{2}}}{h_y^2}}, \tag{10}$$

$$\delta = c_1 \frac{4}{h_x^2} \sin^2 \frac{\pi h_x}{2 l_x} + d_1 \frac{4}{h_y^2} \sin^2 \frac{\pi h_y}{2 l_y}, \tag{11}$$

$$\Delta = c_2 \frac{4}{h_x^2} \cos^2 \frac{\pi h_x}{2 l_x} + d_2 \frac{4}{h_y^2} \cos^2 \frac{\pi h_y}{2 l_y}, \tag{12}$$

$$\xi = \delta / \Delta, \tag{13}$$

$$m \geqslant \frac{\ln(1/\varepsilon)}{4\xi}. \tag{14}$$

- Метод верхней релаксации

$$\begin{gathered} u_{i,j}^k = u_{i,j}^{k-1} + \\ + \omega \frac{p_{i+\frac{1}{2},j} \frac{u_{i-1,j}^{k-1} - u_{i,j}^{k-1}}{h_x^2} - p_{i-\frac{1}{2},j} \frac{u_{i,j}^{k-1} - u_{i-1,j}^{k}}{h_x^2}}{\frac{p_{i-\frac{1}{2},j}}{h_x^2} + \frac{p_{i+\frac{1}{2},j}}{h_x^2} + \frac{q_{i,j-\frac{1}{2}}}{h_y^2} + \frac{q_{i,j+\frac{1}{2}}}{h_y^2}} + \\ + \omega \frac{q_{i,j+\frac{1}{2}} \frac{u_{i,j+1}^{k-1} - u_{i,j}^{k-1}}{h_y^2} - q_{i,j-\frac{1}{2}} \frac{u_{i,j}^{k-1} - u_{i,j-1}^{k}}{h_y^2}}{\frac{p_{i-\frac{1}{2},j}}{h_x^2} + \frac{p_{i+\frac{1}{2},j}}{h_x^2} + \frac{q_{i,j-\frac{1}{2}}}{h_y^2} + \frac{q_{i,j+\frac{1}{2}}}{h_y^2}}, \\ 0 \le \omega \le 2, \end{gathered} \tag{15}$$

$$\delta = c_1 \frac{4}{h_x^2} \sin^2 \frac{\pi h_x}{2 l_x} + d_1 \frac{4}{h_y^2} \sin^2 \frac{\pi h_y}{2 l_y}, \tag{16}$$

$$\Delta = c_2 \frac{4}{h_x^2} \cos^2 \frac{\pi h_x}{2 l_x} + d_2 \frac{4}{h_y^2} \cos^2 \frac{\pi h_y}{2 l_y}, \tag{17}$$

$$\xi = \delta / \Delta, \tag{18}$$

$$m \geqslant \frac{\ln(1/\varepsilon)}{\sqrt{\xi}}. \tag{19}$$

- Попеременно-треугольный итерационный метод

$$\begin{gathered}\tilde{U}^k_{i,j} = \frac{\kappa_1 p_{i-\frac{1}{2},j}\tilde{U}_{i-1,j} + \kappa_2 q_{i,j-\frac{1}{2}}\tilde{U}_{i,j-1} + L_h u^{k-1}_{i,j} + f_{i,j}}{1 + \kappa_1 p_{i-\frac{1}{2},j} + \kappa_2 q_{i,j-\frac{1}{2}}},\\ i = 1, \dots, N-1,\ j = 1, \dots, M-1,\\ \tilde{U}^k\Big|_{\gamma_{h_x h_y}} = 0,\end{gathered} \tag{20}$$

$$\begin{gathered}\overline{U}^k_{i,j} = \frac{\kappa_1 p_{i+\frac{1}{2},j}\overline{U}_{i+1,j} + \kappa_2 q_{i,j+\frac{1}{2}}\overline{U}_{i,j+1} + \tilde{U}_{i,j}}{1 + \kappa_1 p_{i+\frac{1}{2},j} + \kappa_2 q_{i,j+\frac{1}{2}}},\\ i = N-1, \dots, 1,\ j = M-1, \dots, 1,\\ \overline{U}^k\Big|_{\gamma_{h_x h_y}} = 0,\end{gathered} \tag{21}$$

$$u^k_{i,j} = u^{k-1}_{i,j} + \tau \overline{U}^k_{i,j}, \tag{22}$$

$$\delta = c_1 \frac{4}{h_x^2}\sin^2\frac{\pi h_x}{2l_x} + d_1\frac{4}{h_y^2}\sin^2\frac{\pi h_y}{2l_y},\ \Delta = c_2\frac{4}{h_x^2} + d_2\frac{4}{h_y^2}, \tag{23}$$

$$\eta = \delta/\Delta,\ \omega = 2/\sqrt{\delta\Delta},\ \gamma_1 = \frac{\delta}{2+2\sqrt{\eta}},\ \gamma_2 = \frac{\delta}{4\sqrt{\eta}}\ \tau = 2/(\gamma_1+\gamma_2), \tag{24}$$

$$\kappa_1 = \omega/h_x^2,\ \kappa_2 = \omega/h_y^2,\ \xi = \gamma_1/\gamma_2,\ \rho = (1-\xi)/(1+\xi), \tag{25}$$

$$m \geqslant \frac{\ln \varepsilon^{-1}}{\ln \rho^{-1}}. \tag{26}$$

- Метод переменных направлений

$$\begin{cases} u^{k+\frac{1}{2}}_{0,j} = \mu(0, y_j),\\ \overline{A}_{i,j}u^{k+\frac{1}{2}}_{i-1,j} - \overline{B}_{i,j}u^{k+\frac{1}{2}}_{i,j} + \overline{C}_{i,j}u^{k+\frac{1}{2}}_{i+1,j} = \overline{G}^{k+\frac{1}{2}}_{i,j},\ 1 \leqslant i \leqslant N-1,\\ u^{k+\frac{1}{2}}_{N,j} = \mu(l_x, y_j),\end{cases} \tag{27}$$

$$\overline{G}^{k+\frac{1}{2}}_{i,j} = -u^k_{i,j} - \frac{\tau}{2}(\Lambda_2 u^k_{i,j} + f(x_i, y_j)),\ j = 1, \dots, M-1, \tag{28}$$

$$\begin{cases} u^{k+1}_{i,0} = \mu(x_i, 0),\\ \overline{\overline{A}}_{i,j}u^{k+1}_{i,j-1} - \overline{\overline{B}}_{i,j}u^{k+1}_{i,j} + \overline{\overline{C}}_{i,j}u^{k+1}_{i,j+1} = \overline{\overline{G}}^{k+1}_{i,j},\ 1 \leqslant j \leqslant M-1,\\ u^{k+1}_{i,M} = \mu(x_i, l_y),\end{cases} \tag{29}$$

$$\overline{\overline{G}}^{k+1}_{i,j} = -u^{k+\frac{1}{2}}_{i,j} - \frac{\tau}{2}(\Lambda_1 u^{k+\frac{1}{2}}_{i,j} + f(x_i, y_j)),\ i = 1, \dots, N-1, \tag{30}$$

$$\begin{gathered} h_x = h_y = h,\ \overline{A}_{i,j} = \overline{\overline{A}}_{i,j} = \frac{\tau}{2h^2},\\ \overline{B}_{i,j} = \overline{\overline{B}}_{i,j} = \frac{\tau}{h^2} + 1,\ \overline{C}_{i,j} = \overline{\overline{C}}_{i,j} = \frac{\tau}{2h^2},\end{gathered} \tag{31}$$

$$\Lambda_1 u_{i,j} = p_{i+\frac{1}{2},j}\frac{u_{i+1,j} - u_{i,j}}{h_x^2} - p_{i-\frac{1}{2},j}\frac{u_{i,j} - u_{i-1,j}}{h_x^2} \tag{32}$$

$$\Lambda_2 u_{i,j} = q_{i,j+\frac{1}{2}} \frac{u_{i,j+1} - u_{i,j}}{h_y^2} - q_{i,j-\frac{1}{2}} \frac{u_{i,j} - u_{i,j-1}}{h_y^2} \tag{33}$$

$$\delta_1 = c_1 \frac{4}{h_x^2} \sin^2 \frac{\pi h_x}{2l_x}, \ \Delta_1 = c_2 \frac{4}{h_x^2} \cos^2 \frac{\pi h_x}{2l_x}, \tag{34}$$

$$\delta_2 = d_1 \frac{4}{h_y^2} \sin^2 \frac{\pi h_y}{2l_y}, \ \Delta_2 = +d_2 \frac{4}{h_y^2} \cos^2 \frac{\pi h_y}{2l_y}, \tag{35}$$

$$\delta = \min(\delta_1, \delta_2), \ \Delta = \max(\Delta_1, \Delta_2), \ \tau = \frac{2}{\sqrt{\delta \Delta}} \tag{36}$$

$$m \geqslant \frac{N}{2\pi} \ln \frac{1}{\varepsilon} \tag{37}$$

Реализация

Алгоритм реализован на языке программирования Julia в виде скрипта и расположен в GitLab репозитории Computational Workshop S09-2021 в папке A3. Для воспроизведения результатов следуй инструкциям в файле `README.md`.

Проблема:

$$\begin{gathered} Lu = -f(x, y), \\ Lu = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}, \\ 0 \leq x \leq 1, \ 0 \leq y \leq 1, \\ u(x, y)|_{\Gamma} = \mu(x, y) \end{gathered} \tag{38}$$

Для проверки алгоритма предложено решение $u(x, t) = x^3 y + xy^2$.

Листинг 1: Определение проблемы

```julia
# Define the problem
l_x = 1
l_y = 1
p(x, y) = 1
q(x, y) = 1
u(x, y) = x^3 * y + y^2 * x
f(x, y) = -(6 * x * y + 2 * x)

# Set the expected precision of the results
ε = 1e-4
digits = 4
```

Листинг 2: Дифференциальные операторы

```
"Compute the value of the differential operator L"
function L(U, x, y, h_x, h_y, i, j)::Float64
    return Λ₁(U, x, y, h_x, i, j) + Λ₂(U, x, y, h_y, i, j)
end

"Compute the value of the differential operator Λ₁"
function Λ₁(U, x, y, h_x, i, j)::Float64
    return p(x[i] + h_x / 2, y[j]) *
           (U[i+1, j] - U[i, j]) / h_x^2 -
           p(x[i] - h_x / 2, y[j]) *
           (U[i, j] - U[i-1, j]) / h_x^2
end

"Compute the value of the differential operator Λ₂"
function Λ₂(U, x, y, h_y, i, j)::Float64
    return q(x[i], y[j] + h_y / 2) *
           (U[i, j+1] - U[i, j]) / h_y^2 -
           q(x[i], y[j] - h_y / 2) *
           (U[i, j] - U[i, j-1]) / h_y^2
end
```

Листинг 3: Вычисление ξ для первых трех методов

```
"Compute ξ, which is needed to compute
 an optimal number of iterations"
function compute_xi(l_x, l_y, h_x, h_y)::Float64
    rx = 0:0.001:l_x
    ry = 0:0.001:l_y
    pairs = [(x, y) for x in rx, y in ry]
    c₁ = minimum(args -> p(args...), pairs)
    c₂ = maximum(args -> p(args...), pairs)
    d₁ = minimum(args -> q(args...), pairs)
    d₂ = maximum(args -> q(args...), pairs)
    k₁ = 4 / h_x^2
    k₂ = 4 / h_y^2
    arg₁ = (π * h_x) / (2 * l_x)
    arg₂ = (π * h_y) / (2 * l_y)
    δ = c₁ * k₁ * sin(arg₁)^2 +
        d₁ * k₂ * sin(arg₂)^2
    Δ = c₂ * k₁ * cos(arg₁)^2 +
        d₂ * k₂ * cos(arg₂)^2
    return δ / Δ
end
```

Листинг 4: Вычисление матрицы U^0

```
"Initialize a new matrix and compute the boundary values"
function boundary_values(N, M, x, y)::Matrix{Float64}
    # Prepare a matrix for the solution
    U = zeros(N + 1, M + 1)
    # Compute the boundary values
    U[:, 1] .= u.(x, 0)
    U[:, M+1] .= u.(x, l_y)
    U[1, 2:M] .= u.(0, y[2:M])
    U[N+1, 2:M] .= u.(l_x, y[2:M])
    return U
end
```

Листинг 5: Метод простых итераций

```
# Compute the optimal number of iterations
m = ceil(Int, log(1 / ε) / (2ξ))
# Iterate enough times to achieve desired precision
for _ in 1:m
    # Compute values at the inner nodes
    for i = 2:N, j = 2:M
        k₁ = p(x[i] - h_x / 2, y[j]) / h_x^2
        k₂ = p(x[i] + h_x / 2, y[j]) / h_x^2
        k₃ = q(x[i], y[j] - h_y / 2) / h_y^2
        k₄ = q(x[i], y[j] + h_y / 2) / h_y^2
        Uₖ[i, j] = (k₁ * Uₖ₋₁[i-1, j] +
                    k₂ * Uₖ₋₁[i+1, j] +
                    k₃ * Uₖ₋₁[i, j-1] +
                    k₄ * Uₖ₋₁[i, j+1] +
                    f(x[i], y[j])) /
                   (k₁ + k₂ + k₃ + k₄)
    end
    # Reassign the current iteration as the previous one
    Uₖ₋₁ = Uₖ
end
# <...>
```

Листинг 6: Метод Зейделя

```
# Compute the optimal number of iterations
m = ceil(Int, log(1 / ε) / (4ξ))
# Iterate enough times to achieve desired precision
for _ in 1:m
    # Compute values at the inner nodes
    for i = N:-1:2, j = M:-1:2
        k₁ = p(x[i] - h_x / 2, y[j]) / h_x^2
        k₂ = p(x[i] + h_x / 2, y[j]) / h_x^2
        k₃ = q(x[i], y[j] - h_y / 2) / h_y^2
        k₄ = q(x[i], y[j] + h_y / 2) / h_y^2
        Uₖ[i, j] = (k₁ * Uₖ[i-1, j] +
                    k₂ * Uₖ₋₁[i+1, j] +
                    k₃ * Uₖ[i, j-1] +
                    k₄ * Uₖ₋₁[i, j+1] +
                    f(x[i], y[j])) /
                    (k₁ + k₂ + k₃ + k₄)
    end
    # Reassign the current iteration as the previous one
    Uₖ₋₁ = Uₖ
end
# <...>
```

Листинг 7: Метод верхней релаксации

```
# <...>
# Compute the optimal number of iterations
m = ceil(Int, log(1 / ε) / sqrt(ξ))
# Define ω in (0,2), which affects the speed of convergence
ω = 1.0
# Iterate enough times to achieve desired precision
for _ in 1:m
    # Compute values at the inner nodes
    for i = N:-1:2, j = M:-1:2
        k₁ = p(x[i] + h_x / 2, y[j]) / h_x^2
        k₂ = p(x[i] - h_x / 2, y[j]) / h_x^2
        k₃ = q(x[i], y[j] + h_y / 2) / h_y^2
        k₄ = q(x[i], y[j] - h_y / 2) / h_y^2
        Uₖ[i, j] = Uₖ₋₁[i, j] +
                   ω * (k₁ * (Uₖ₋₁[i+1, j] - Uₖ₋₁[i, j]) -
                        k₂ * (Uₖ₋₁[i, j] - Uₖ[i-1, j]) +
                        k₃ * (Uₖ₋₁[i, j+1] - Uₖ₋₁[i, j]) -
                        k₄ * (Uₖ₋₁[i, j] - Uₖ[i, j-1]) +
                        f(x[i], y[j])) /
                   (k₁ + k₂ + k₃ + k₄)
    end
    # Reassign the current iteration as the previous one
    Uₖ₋₁ = Uₖ
end
# <...>
```

Листинг 8: Попеременно-треугольный итерационный метод (часть 1)

```
# <...>
# Compute the coefficients
pairs = [(x, y) for x in 0:0.001:l_x, y in 0:0.001:l_y]
c₁ = minimum(args -> p(args...), pairs)
c₂ = maximum(args -> p(args...), pairs)
d₁ = minimum(args -> q(args...), pairs)
d₂ = maximum(args -> q(args...), pairs)
k₁ = 4 / h_x^2
k₂ = 4 / h_y^2
δ = c₁ * k₁ * sin((π * h_x) / (2 * l_x))^2 +
    d₁ * k₂ * sin((π * h_y) / (2 * l_y))^2
Δ = c₂ * k₁ + d₂ * k₂
ω = 2 / sqrt(δ * Δ)
η = δ / Δ
γ₁ = δ / (2 + 2 * sqrt(η))
γ₂ = δ / (4 * sqrt(η))
ξ = γ₁ / γ₂
κ₁ = ω / h_x^2
κ₂ = ω / h_y^2
τ = 2 / (γ₁ + γ₂)
# <...>
```

Листинг 9: Попеременно-треугольный итерационный метод (часть 2)

```
# <...>
# Compute the optimal number of iterations
m = ceil(Int, log(1 / ε) / log((1 + ξ) / (1 - ξ)))
# Iterate enough times to achieve desired precision
for _ in 1:m
    # Prepare intermediate matrices
    Ũ = zeros(size(Uₖ₋₁)...)
    Ū = zeros(size(Uₖ₋₁)...)
    # Compute values at the inner nodes of the first
    intermediate matrix
    for i = 2:N, j = 2:M
        k₁ = κ₁ * p(x[i] - h_x / 2, y[j])
        k₂ = κ₂ * q(x[i], y[j] - h_y / 2)
        Ũ[i, j] = (k₁ * Ũ[i-1, j] +
                   k₂ * Ũ[i, j-1] +
                   L(Uₖ₋₁, x, y, h_x, h_y, i, j) +
                   f(x[i], y[j])) /
                  (1 + k₁ + k₂)
    end
    # Compute values at the inner nodes of the second
    intermediate matrix
    for i = N:-1:2, j = M:-1:2
        k₁ = κ₁ * p(x[i] + h_x / 2, y[j])
        k₂ = κ₂ * q(x[i], y[j] + h_y / 2)
        Ū[i, j] = (k₁ * Ū[i+1, j] +
                   k₂ * Ū[i, j+1] + Ũ[i, j]) /
                  (1 + k₁ + k₂)
    end
    # Add up the matrices
    Uₖ[2:N, 2:M] .= Uₖ₋₁[2:N, 2:M] .+ τ .* Ū[2:N, 2:M]
    # Reassign the current iteration as the previous one
    Uₖ₋₁ = Uₖ
end
# <...>
```

Листинг 10: Метод переменных направлений (часть 1)

```
# <...>
# Compute the optimal number of iterations
m = ceil(Int, N / (2 * π) * log(1 / ε))
# Compute the coefficients
pairs = [(x, y) for x in 0:0.001:l_x, y in 0:0.001:l_y]
c₁ = minimum(args -> p(args...), pairs)
c₂ = maximum(args -> p(args...), pairs)
d₁ = minimum(args -> q(args...), pairs)
d₂ = maximum(args -> q(args...), pairs)
k₁ = 4 / h_x^2
k₂ = 4 / h_y^2
arg₁ = (π * h_x) / (2 * l_x)
arg₂ = (π * h_y) / (2 * l_y)
δ₁ = c₁ * k₁ * sin(arg₁)^2
δ₂ = d₁ * k₂ * sin(arg₂)^2
Δ₁ = c₂ * k₁ * cos(arg₁)^2
Δ₂ = d₂ * k₁ * cos(arg₂)^2
δ = min(δ₁, δ₂)
Δ = max(Δ₁, Δ₂)
τ = 2 / sqrt(δ * Δ)
# <...>
```

Листинг 11: Метод переменных направлений (часть 2)

```julia
# <...>
# Iterate enough times to achieve desired precision
for _ in 1:m
    # Compute values at the inner nodes of the intermediate
     matrix
    for j = 2:M
        # Compute the linear system's matrix
        Ũ = Tridiagonal(
            # A's
            [repeat([τ / (2 * h_x^2)], N - 1); 0],
            # B's
            [1; repeat([-τ / h_x^2 + 1], N - 1); 1],
            # C's
            [0; repeat([τ / (2 * h_x^2)], N - 1)],
        )
        # Compute the linear system's right-hand side
    vector
        g = [
            Uₖ₋₁[1, j]
            [-Uₖ₋₁[i, j] -
             τ / 2 * (Λ₂(Uₖ₋₁, x, y, h_y, i, j) +
                       f(x[i], y[j]))
             for i in 2:N
            ]
            Uₖ₋₁[N+1, j]]
        # Compute the solution of the linear system
        Ū[:, j] .= Ũ \ g
    end
# <...>
```

Листинг 12: Метод переменных направлений (часть 3)

```
# <...>
    # Compute values at the inner nodes of the input matrix
    for i = 2:N
        # Compute the linear system's matrix
        Ũ = Tridiagonal(
            # A's
            [repeat([τ / (2 * h_y^2)], N - 1); 0],
            # B's
            [1; repeat([-τ / h_y^2 + 1], N - 1); 1],
            # C's
            [0; repeat([τ / (2 * h_y^2)], N - 1)],
        )
        # Compute the linear system's
        # right-hand side vector
        g = [
            Uₖ₋₁[i, 1]
            [-Ū[i, j] -
             τ / 2 * (Λ₁(Ū, x, y, h_x, i, j) +
                      f(x[i], y[j]))
             for j in 2:M
            ]
            Uₖ₋₁[i, M+1]]
        # Compute the solution of the linear system
        Uₖ[i, :] .= Ũ \ g
    end
    # Reassign the current iteration as the previous one
    Uₖ₋₁ = Uₖ
end
# <...>
```

Таблица 1: Решение дифференциального уравнения

$x \backslash y$	0	0.2	0.4	0.6	0.8	1
0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.2	0.0000	0.0096	0.0352	0.0768	0.1344	0.2080
0.4	0.0000	0.0288	0.0896	0.1824	0.3072	0.4640
0.6	0.0000	0.0672	0.1824	0.3456	0.5568	0.8160
0.8	0.0000	0.1344	0.3328	0.5952	0.9216	1.3120
1.0	0.0000	0.2400	0.5600	0.9600	1.4400	2.0000

Таблица 2: Сравнение результатов с точным решением

N, M	$m_1, \Delta u_1$	$m_2, \Delta u_2$	$m_3, \Delta u_3$	$m_4, \Delta u_4$	$m_5, \Delta u_5$
5	$44, 10^{-9}$	$22, 10^{-13}$	$29, 10^{-16}$	$9, 10^{-16}$	$8, 10^{-15}$
10	$184, 10^{-9}$	$92, 10^{-13}$	$59, 10^{-15}$	$17, 10^{-16}$	$15, 10^{-14}$
20	$744, 10^{-9}$	$372, 10^{-13}$	$118, 10^{-14}$	$32, 10^{-16}$	$30, 10^{-14}$