B. Численный метод для параболической модели (2D)

B1. Маршевая постановка по ξ и схема Кранка–Николсона

В этом разделе строится численный метод для параболической модели, полученной в разделе A. Основная идея состоит в том, что после параболизации уравнение становится *маршевым* по координате ξ: при заданном начальном профиле на $\xi = 0$ решение последовательно вычисляется на слоях

$$\xi_0 \to \xi_1 \to \xi_2 \to \cdots .$$

Зачем нужен этот раздел. Этот раздел является основой для вычислительной части:

- здесь фиксируется сетка по (ξ, s);
- вводятся дискретные операторы по s;
- выводится неявная схема Кранка–Николсона (CN);
- становится ясно, какую именно систему линейных уравнений нужно решать на каждом шаге.

B1.1. Исходная параболическая модель (напоминание)

Из раздела A получена параболическая модель для огибающей $u(\xi, s)$:

$$2ik\,u_\xi + \frac{1}{\rho+\xi}u_\xi + \frac{ik}{\rho+\xi}u + \left(\frac{\rho}{\rho+\xi}\right)^2 u_{ss} + \frac{\xi\rho\rho_s}{(\rho+\xi)^3}u_s = 0, \tag{1}$$

где

$$\rho = \rho(s), \qquad \rho_s = \frac{d\rho}{ds}.$$

Удобно собрать коэффициент при u_ξ:

$$\left(2ik + \frac{1}{\rho+\xi}\right) u_\xi + \left(\frac{\rho}{\rho+\xi}\right)^2 u_{ss} + \frac{\xi\rho\rho_s}{(\rho+\xi)^3}u_s + \frac{ik}{\rho+\xi}u = 0.$$

Отсюда:

$$u_\xi = -\frac{\left(\frac{\rho}{\rho+\xi}\right)^2 u_{ss} + \frac{\xi\rho\rho_s}{(\rho+\xi)^3}u_s + \frac{ik}{\rho+\xi}u}{2ik + \frac{1}{\rho+\xi}}. \tag{2}$$

Почему форма (2) удобна. Она явно показывает, что:

- ξ играет роль “времени” марша;
- вдоль s нужно аппроксимировать производные u_s и u_{ss};
- коэффициенты зависят от геометрии (через $\rho(s)$) и слоя ξ.

B1.2. Введение коэффициентов и операторная форма

Чтобы сделать запись компактной, введём коэффициенты:

$$A(\xi, s) := 2ik + \frac{1}{\rho(s) + \xi}, \tag{3}$$

$$B(\xi, s) := \left(\frac{\rho(s)}{\rho(s) + \xi}\right)^2, \tag{4}$$

$$C(\xi, s) := \frac{\xi\, \rho(s)\, \rho_s(s)}{(\rho(s) + \xi)^3}, \tag{5}$$

$$D(\xi, s) := \frac{ik}{\rho(s) + \xi}. \tag{6}$$

Тогда (2) переписывается как

$$u_\xi = -A^{-1}\big(Bu_{ss} + Cu_s + Du\big). \tag{7}$$

Ещё удобнее ввести

$$\alpha(\xi, s) := -\frac{B(\xi, s)}{A(\xi, s)}, \tag{8}$$

$$\beta(\xi, s) := -\frac{C(\xi, s)}{A(\xi, s)}, \tag{9}$$

$$\gamma(\xi, s) := -\frac{D(\xi, s)}{A(\xi, s)}. \tag{10}$$

Тогда получаем компактную маршевую форму:

$$u_\xi = \alpha(\xi, s)\, u_{ss} + \beta(\xi, s)\, u_s + \gamma(\xi, s)\, u. \tag{11}$$

Важно. Коэффициенты α, β, γ в общем случае:

- комплексные (из-за ik),
- зависят от ξ,
- зависят от s (через $\rho(s)$ и $\rho_s(s)$).

Поэтому численная схема должна корректно работать с *комплексными переменными коэффициентами*.

B1.3. Сетка по ξ и s

Сетка по направлению марша ξ. Пусть область по ξ имеет вид

$$0 \le \xi \le \Xi.$$

Вводится равномерная сетка:

$$\xi_n = n\Delta\xi, \qquad n = 0, 1, \ldots, N_\xi, \qquad \Delta\xi = \frac{\Xi}{N_\xi}. \tag{12}$$

Сетка вдоль контура s. Пусть длина контура равна L. Вводится равномерная сетка по s:

$$s_j = j\Delta s, \qquad j = 0, 1, \ldots, N_s - 1, \qquad \Delta s = \frac{L}{N_s}. \tag{13}$$

Периодичность по s. Так как s — координата вдоль замкнутого контура, естественно использовать периодические условия:

$$u(\xi, s + L) = u(\xi, s). \tag{14}$$

На сетке это означает циклические индексы:

$$U^n_{j+N_s} = U^n_j, \qquad U^n_{-1} = U^n_{N_s-1}, \qquad U^n_{N_s} = U^n_0.$$

Сеточные обозначения. Обозначим сеточную аппроксимацию:

$$U^n_j \approx u(\xi_n, s_j). \tag{15}$$

B1.4. Дискретизация производных по s

Для схемы Кранка–Николсона используется центральная аппроксимация производных по s.

Первая производная по s.

$$(\delta_s U)_j := \frac{U_{j+1} - U_{j-1}}{2\Delta s}. \tag{16}$$

Вторая производная по s.

$$(\delta_{ss} U)_j := \frac{U_{j+1} - 2U_j + U_{j-1}}{\Delta s^2}. \tag{17}$$

Порядок аппроксимации. При достаточной гладкости u:

$$(\delta_s U)_j = u_s(\xi, s_j) + O(\Delta s^2), \qquad (\delta_{ss} U)_j = u_{ss}(\xi, s_j) + O(\Delta s^2).$$

То есть по переменной s используется *второй порядок точности*.

B1.5. Полудискретный оператор на слое ξ

Для фиксированного слоя ξ определим дискретный оператор $\mathcal{M}(\xi)$ по формуле

$$(\mathcal{M}(\xi)U)_j = \alpha(\xi, s_j)(\delta_{ss} U)_j + \beta(\xi, s_j)(\delta_s U)_j + \gamma(\xi, s_j) U_j. \tag{18}$$

Тогда уравнение (11) после дискретизации по s принимает вид

$$\frac{d}{d\xi} U(\xi) = \mathcal{M}(\xi) U(\xi), \tag{19}$$

где $U(\xi)$ — вектор значений на сетке по s.

Почему это удобно. Теперь задача сведена к системе ОДУ по ξ:

$$U'(\xi) = \mathcal{M}(\xi) U(\xi),$$

к которой можно применить схему Кранка–Николсона по “времени” ξ.

B1.6. Схема Кранка–Николсона по ξ

Схема Кранка–Николсона (CN) для системы (19) на шаге $\xi_n \to \xi_{n+1}$ записывается как

$$\frac{U^{n+1} - U^n}{\Delta\xi} = \frac{1}{2}\,\mathcal{M}^{n+\frac{1}{2}} U^{n+1} + \frac{1}{2}\,\mathcal{M}^{n+\frac{1}{2}} U^n, \tag{20}$$

где

$$\mathcal{M}^{n+\frac{1}{2}} := \mathcal{M}(\xi_{n+\frac{1}{2}}), \qquad \xi_{n+\frac{1}{2}} := \xi_n + \frac{\Delta\xi}{2}.$$

Почему коэффициенты берутся на полушаге. Это стандартный способ получить второй порядок по ξ:

- коэффициенты фиксируются на середине слоя,
- правая часть усредняется между U^n и U^{n+1}.

Такой вариант естественен для переменных коэффициентов.

Переход к матричной форме. Переносим члены с U^{n+1} влево:

$$U^{n+1} - \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}} U^{n+1} = U^n + \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}} U^n.$$

То есть

$$\boxed{\left(I - \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^{n+1} = \left(I + \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^n.} \tag{21}$$

Смысл формулы (21)**.** На каждом шаге по ξ нужно:

1. вычислить правую часть

$$b^n := \left(I + \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^n;$$

2. решить линейную систему

$$\left(I - \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^{n+1} = b^n.$$

B1.7. Покомпонентная запись схемы и коэффициенты матрицы

Теперь выпишем (21) по компонентам, чтобы было ясно, какая именно структура у матрицы.

Коэффициенты на полушаге. Для краткости обозначим

$$\alpha_j^{n+\frac{1}{2}} := \alpha(\xi_{n+\frac{1}{2}}, s_j), \tag{22}$$

$$\beta_j^{n+\frac{1}{2}} := \beta(\xi_{n+\frac{1}{2}}, s_j), \tag{23}$$

$$\gamma_j^{n+\frac{1}{2}} := \gamma(\xi_{n+\frac{1}{2}}, s_j). \tag{24}$$

Тогда

$$(\mathcal{M}^{n+\frac{1}{2}}U)_j = \alpha_j^{n+\frac{1}{2}}\frac{U_{j+1}-2U_j+U_{j-1}}{\Delta s^2} + \beta_j^{n+\frac{1}{2}}\frac{U_{j+1}-U_{j-1}}{2\Delta s} + \gamma_j^{n+\frac{1}{2}}U_j.$$

Собираем коэффициенты при U_{j-1}, U_j, U_{j+1}:

$$(\mathcal{M}^{n+\frac{1}{2}}U)_j = p_j^{n+\frac{1}{2}}U_{j-1} + q_j^{n+\frac{1}{2}}U_j + r_j^{n+\frac{1}{2}}U_{j+1}, \tag{25}$$

где

$$p_j^{n+\frac{1}{2}} = \frac{\alpha_j^{n+\frac{1}{2}}}{\Delta s^2} - \frac{\beta_j^{n+\frac{1}{2}}}{2\Delta s}, \tag{26}$$

$$q_j^{n+\frac{1}{2}} = -\frac{2\alpha_j^{n+\frac{1}{2}}}{\Delta s^2} + \gamma_j^{n+\frac{1}{2}}, \tag{27}$$

$$r_j^{n+\frac{1}{2}} = \frac{\alpha_j^{n+\frac{1}{2}}}{\Delta s^2} + \frac{\beta_j^{n+\frac{1}{2}}}{2\Delta s}. \tag{28}$$

Левая часть CN-схемы. Из (21) левая часть имеет вид

$$\left(I - \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}}\right)U^{n+1}.$$

Поэтому в j-й строке коэффициенты равны:

$$\text{при } U_{j-1}^{n+1}: \quad -\frac{\Delta\xi}{2}p_j^{n+\frac{1}{2}}, \tag{29}$$

$$\text{при } U_j^{n+1}: \quad 1 - \frac{\Delta\xi}{2}q_j^{n+\frac{1}{2}}, \tag{30}$$

$$\text{при } U_{j+1}^{n+1}: \quad -\frac{\Delta\xi}{2}r_j^{n+\frac{1}{2}}. \tag{31}$$

Правая часть CN-схемы. Аналогично, правая часть

$$\left(I + \frac{\Delta\xi}{2}\mathcal{M}^{n+\frac{1}{2}}\right)U^n$$

имеет компоненты

$$b_j^n = \frac{\Delta\xi}{2}p_j^{n+\frac{1}{2}}U_{j-1}^n + \left(1 + \frac{\Delta\xi}{2}q_j^{n+\frac{1}{2}}\right)U_j^n + \frac{\Delta\xi}{2}r_j^{n+\frac{1}{2}}U_{j+1}^n. \tag{32}$$

Итоговая строка системы. Таким образом, для каждого $j = 0, \ldots, N_s - 1$:

$$-\frac{\Delta\xi}{2}p_j^{n+\frac{1}{2}}U_{j-1}^{n+1} + \left(1 - \frac{\Delta\xi}{2}q_j^{n+\frac{1}{2}}\right)U_j^{n+1} - \frac{\Delta\xi}{2}r_j^{n+\frac{1}{2}}U_{j+1}^{n+1} = b_j^n. \tag{33}$$

Это и есть явная форма схемы Кранка–Николсона на шаге по ξ.

B1.8. Структура матрицы: циклическая трёхдиагональная система

Из-за периодичности по s (см. (14)) индексы $j-1$ и $j+1$ берутся по модулю N_s. Поэтому матрица в (21) имеет структуру:

- трёхдиагональная “внутри”,
- плюс два угловых элемента (связь $j = 0$ с $j = N_s - 1$ и наоборот).

Такая матрица называется *циклической трёхдиагональной*.

Почему это важно вычислительно. Для этой структуры можно использовать эффективные решатели:

- модифицированный метод прогонки (для циклического случая),
- Sherman–Morrison / Woodbury-поправка,
- FFT-диагонализация (в частном случае постоянных коэффициентов по s).

Именно этот факт позволяет сделать шаг CN быстрым даже при больших N_s.

B1.9. Начальные данные и граничные условия по ξ

Так как задача маршевая, требуется начальный профиль на $\xi = 0$:

$$u(0, s) = u_0(s). \tag{34}$$

На сетке:

$$U_j^0 = u_0(s_j), \qquad j = 0, \dots, N_s - 1. \tag{35}$$

Что задаётся в $u_0(s)$. Это определяется физической постановкой:

- поле на стартовом сечении,
- или огибающая, полученная из известного полного поля,
- или начальный профиль для тестовой задачи (например, для верификации).

Замечание. В отличие от эллиптической постановки Гельмгольца, здесь *не нужно* задавать граничные условия по ξ на обоих концах интервала. Достаточно начального слоя $\xi = 0$, что и делает метод быстрым.

B1.10. Частный случай: окружность (важный эталон)

Для окружности

$$\rho(s) \equiv R = \text{const}, \qquad \rho_s(s) = 0.$$

Тогда коэффициенты упрощаются:

$$A(\xi) = 2ik + \frac{1}{R + \xi}, \tag{36}$$

$$B(\xi) = \left(\frac{R}{R + \xi}\right)^2, \tag{37}$$

$$C(\xi) = 0, \tag{38}$$

$$D(\xi) = \frac{ik}{R + \xi}. \tag{39}$$

Следовательно,

$$\beta(\xi, s) \equiv 0,$$

и оператор (11) становится

$$u_\xi = \alpha(\xi) u_{ss} + \gamma(\xi) u. \tag{40}$$

Почему это удобно. В случае окружности:

- нет члена с первой производной u_s;
- коэффициенты не зависят от s;
- матрица на каждом шаге имеет циклическую трёхдиагональную структуру с одинаковыми коэффициентами по строкам (циклическая Toeplitz-матрица).

Это идеальный эталон для:

- отладки схемы CN,
- проверки порядка сходимости,
- исследования ошибки округления и выбора шага.

B1.11. Точность схемы и что проверять в вычислительных тестах

Порядок аппроксимации. При гладком решении и коэффициентах:

- по ξ схема CN даёт второй порядок: $O(\Delta\xi^2)$;
- по s центральные разности дают второй порядок: $O(\Delta s^2)$.

Итого в базовой конфигурации ожидается

$$\text{ошибка} = O(\Delta\xi^2 + \Delta s^2),$$

пока не начинают доминировать модельная ошибка и/или округление.

Что нужно проверять численно. В обязательный набор тестов входят:

1. **Сходимость по** Δs при фиксированном малом $\Delta\xi$.
2. **Сходимость по** $\Delta\xi$ при фиксированном достаточно большом N_s.
3. **Сходимость по двум параметрам одновременно** (log–log графики).
4. **Невязка PE и полной модели** (см. раздел A2).
5. **Сравнение double / long double** для диагностики округления.

B1.12. Итог по разделу B1

В этом разделе получена полная рабочая численная схема для параболической модели:

1. уравнение приведено к маршевой форме (11);
2. введена сетка по (ξ, s) и периодические условия по s;
3. построены дискретные операторы δ_s, δ_{ss};
4. выведена схема Кранка–Николсона (21);
5. выписана покомпонентная структура циклической трёхдиагональной системы;
6. выделен частный случай окружности как основной эталон для верификации.

Что должно быть понятно после B1. После этого раздела должно быть полностью понятно:

1. какую именно задачу решает схема CN;
2. какие коэффициенты входят в оператор и откуда они берутся;
3. как устроена матрица на каждом шаге;
4. почему задача решается маршем по ξ;
5. почему окружность является первым и обязательным тестовым случаем.

B2. Апостериорная оценка ошибки, адаптивный шаг по ξ и выбор параметров расчёта

В этом разделе вводится практический механизм контроля точности при марше по ξ: локальная апостериорная оценка ошибки, адаптивный выбор шага $\Delta\xi$ и правила принятия/отклонения шага.

Зачем нужен этот раздел. Даже при хорошей схеме Кранка–Николсона (CN) фиксированный шаг по ξ неудобен:

- в одних участках решения шаг можно брать крупнее (и считать быстрее),
- в других участках нужен более мелкий шаг (иначе теряется точность),
- слишком малый шаг может привести к лишним вычислениям и усилению округлительной ошибки.

Поэтому нужен *адаптивный* механизм, который:

- автоматически оценивает локальную ошибку,
- уменьшает шаг там, где решение быстро меняется,
- увеличивает шаг там, где решение гладкое.

B2.1. Напоминание: маршевая задача по ξ

Из B1 получена полудискретная система (после дискретизации по s):

$$\frac{d}{d\xi}U(\xi) = \mathcal{M}(\xi)U(\xi), \tag{41}$$

где $U(\xi) \in$

B2.2. Схема CN на переменном шаге

Для переменного шага h_n схема Кранка–Николсона записывается так же, как в B1, только с заменой $\Delta\xi \mapsto h_n$:

$$\boxed{\left(I - \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}\right)U^{n+1} = \left(I + \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}\right)U^n,} \tag{42}$$

где

$$\xi_{n+\frac{1}{2}} := \xi_n + \frac{h_n}{2}, \qquad \mathcal{M}^{n+\frac{1}{2}} := \mathcal{M}(\xi_{n+\frac{1}{2}}).$$

Замечание. Структура линейной системы сохраняется: на каждом шаге по-прежнему решается циклическая трёхдиагональная система (см. B1.8), только коэффициенты теперь пересчитываются для текущего шага h_n и текущего полуслоя $\xi_{n+\frac{1}{2}}$.

B2.3. Идея апостериорной оценки: "один шаг" vs "два полушага"

Для оценки локальной ошибки на шаге $[\xi_n, \xi_n + h_n]$ используется стандартный приём:

- выполнить **один полный шаг** длины h_n;
- выполнить **два полушага** длины $h_n/2$;
- сравнить результаты в точке $\xi_{n+1} = \xi_n + h_n$.

Обозначения. Пусть:

- U^{n+1}_{full} — результат одного шага CN длины h_n;
- U^{n+1}_{half} — результат двух последовательных шагов CN длины $h_n/2$.

Тогда разность

$$\Delta U^{n+1} := U^{n+1}_{\text{half}} - U^{n+1}_{\text{full}} \tag{43}$$

является апостериорным индикатором локальной ошибки.

Почему это работает. CN имеет второй порядок точности по ξ (глобально), а локальная ошибка шага имеет порядок $O(h_n^3)$. Поэтому:

- ошибка одного полного шага — порядка $O(h_n^3)$;
- ошибка двух полушагов — тоже порядка $O(h_n^3)$, но с меньшей константой;
- их разность (43) имеет тот же масштаб и может использоваться как оценка.

B2.4. Вывод оценки Ричардсона для CN (порядок 2)

Пусть U^{n+1}_* — точное решение полудискретной системы (41) в точке ξ_{n+1} (при точной арифметике и точном решении линейных систем). Тогда для метода второго порядка можно записать асимптотически:

$$U^{n+1}_{\text{full}} = U^{n+1}_* + C\, h_n^3 + O(h_n^4), \tag{44}$$

$$U^{n+1}_{\text{half}} = U^{n+1}_* + C\, \frac{h_n^3}{2^2} + O(h_n^4) = U^{n+1}_* + \frac{C}{4} h_n^3 + O(h_n^4), \tag{45}$$

где C — векторная константа (зависящая от решения и коэффициентов).

Вычитаем (44) из (45):

$$\Delta U^{n+1} = U^{n+1}_{\text{half}} - U^{n+1}_{\text{full}} = -\frac{3}{4} C\, h_n^3 + O(h_n^4).$$

Отсюда локальная ошибка более точного варианта (двух полушагов) оценивается как

$$e^{n+1}_{\text{loc}} \;\approx\; \frac{1}{2^2 - 1}\, \Delta U^{n+1} = \frac{1}{3}\, \Delta U^{n+1}. \tag{46}$$

Итоговая формула оценки. В норме получаем практическую оценку локальной ошибки:

$$\boxed{\|e_{\text{loc}}^{n+1}\| \;\approx\; \frac{1}{3}\,\|\Delta U^{n+1}\|.} \tag{47}$$

Важно. Здесь оценивается *локальная ошибка шага по* ξ для полудискретной системы. Это не полная ошибка относительно исходного уравнения Гельмгольца: модельная ошибка (параболизация) и ошибка по s здесь не исчезают, а контролируются отдельно (см. раздел A2 и дальнейшие пункты B).

B2.5. Ричардсоновская экстраполяция (улучшенный шаг)

Кроме оценки ошибки, из пары решений можно получить улучшенное значение в конце шага (опционально):

$$U_{\text{rich}}^{n+1} := U_{\text{half}}^{n+1} + \frac{U_{\text{half}}^{n+1} - U_{\text{full}}^{n+1}}{2^2 - 1} = U_{\text{half}}^{n+1} + \frac{1}{3}\Delta U^{n+1}. \tag{48}$$

Зачем это может быть полезно.

- U_{half}^{n+1} уже точнее, чем U_{full}^{n+1};
- U_{rich}^{n+1} обычно даёт ещё более точное значение на шаге;
- это полезно при стремлении к “почти машинной точности” на эталонных тестах.

Практическое замечание. В базовой реализации можно сначала использовать более простой вариант:

- оценка ошибки по (47),
- принятие шага,
- в качестве U^{n+1} брать U_{half}^{n+1}.

Ричардсоновскую экстраполяцию (48) удобно добавить после отладки базового адаптивного механизма.

B2.6. Норма для контроля ошибки на шаге

Для принятия/отклонения шага нужна *численная* норма вектора ошибки. Возможны разные варианты; ниже фиксируются два практических.

(i) Взвешенная дискретная L_2-норма по s. На слое $\xi = \xi_{n+1}$:

$$\|V\|_{2,w}^2 := \sum_{j=0}^{N_s-1} |V_j|^2\, w_j^{n+1}\, \Delta s, \qquad w_j^{n+1} := 1 + \frac{\xi_{n+1}}{\rho(s_j)}. \tag{49}$$

Тогда

$$\|V\|_{2,w} = \left(\sum_{j=0}^{N_s-1} |V_j|^2 w_j^{n+1} \Delta s \right)^{1/2}.$$

(ii) Дискретная равномерная норма (L_∞).

$$\|V\|_\infty := \max_{0\le j\le N_s-1} |V_j|. \tag{50}$$

Что использовать на практике.

- Для общего контроля энергии/средней точности удобно $\|\cdot\|_{2,w}$.
- Для защиты от локальных выбросов и фазовых "провалов" полезно дополнительно контролировать $\|\cdot\|_\infty$.

B2.7. Абсолютная и относительная точность на одном шаге

Оценка (47) сама по себе недостаточна: её нужно сравнивать с допуском. Если решение меняет масштаб, фиксированный абсолютный допуск неудобен. Поэтому используется смешанный критерий: *абсолютный + относительный*.

Скалярный критерий (через одну норму). Пусть заданы параметры:

$$\mathrm{ATOL} > 0, \qquad \mathrm{RTOL} > 0.$$

Определим масштаб решения на конце шага:

$$S_n := \max\big(\|U^{n+1}_{\mathrm{half}}\|,\ \|U^{n+1}_{\mathrm{full}}\|\big), \tag{51}$$

где $\|\cdot\|$ — выбранная норма (например, $\|\cdot\|_{2,w}$).

Тогда допустимая локальная ошибка:

$$\mathrm{TOL}_n := \mathrm{ATOL} + \mathrm{RTOL}\cdot S_n. \tag{52}$$

И нормированный индикатор шага:

$$\eta_n := \frac{\frac{1}{3}\|\Delta U^{n+1}\|}{\mathrm{TOL}_n}. \tag{53}$$

Правило принятия шага.

$$\boxed{\eta_n \le 1 \implies \text{шаг принимается}, \qquad \eta_n > 1 \implies \text{шаг отклоняется.}} \tag{54}$$

Более строгий (покомпонентный) критерий. В практической реализации часто удобно дополнительно использовать покомпонентное масштабирование:

$$\mathrm{sc}_j := \mathrm{ATOL} + \mathrm{RTOL}\cdot \max\big(|(U^{n+1}_{\mathrm{half}})_j|,\ |(U^{n+1}_{\mathrm{full}})_j|\big), \tag{55}$$

и нормированный вектор ошибки

$$Z_j := \frac{(\Delta U^{n+1})_j/3}{\mathrm{sc}_j}.$$

Тогда можно контролировать, например,

$$\|Z\|_\infty \le 1 \quad \text{или} \quad \left(\frac{1}{N_s}\sum_{j=0}^{N_s-1}|Z_j|^2\right)^{1/2} \le 1.$$

Замечание. Для диссертационного текста достаточно зафиксировать один основной критерий (например, (53)) и отдельно указать, что в коде может использоваться более жёсткий покомпонентный вариант.

B2.8. Формула выбора следующего шага по ξ

Так как локальная ошибка шага CN имеет порядок $O(h_n^3)$, новый шаг выбирается по правилу

$$h_{n+1} = h_n \left(\frac{1}{\eta_n}\right)^{1/3}. \tag{56}$$

Чтобы избежать резких скачков шага, вводится *коэффициент безопасности* $\sigma \in (0,1)$ (например, $\sigma = 0.8$ или $\sigma = 0.9$):

$$h_{n+1} = \sigma\, h_n \left(\frac{1}{\eta_n}\right)^{1/3}. \tag{57}$$

Ограничения на изменение шага. На практике полезно ограничивать слишком сильное уменьшение/увеличение:

$$h_{n+1} = h_n \cdot \min\Big(f_{\max},\ \max\big(f_{\min},\ \sigma\eta_n^{-1/3}\big)\Big), \tag{58}$$

где обычно выбирают

$$0 < f_{\min} < 1 < f_{\max},$$

например,

$$f_{\min} = 0.2, \qquad f_{\max} = 2.$$

Если шаг отклонён. Если $\eta_n > 1$, шаг не принимается, и расчёт повторяется с уменьшенным шагом (по формуле (58)), начиная с того же значения U^n.

B2.9. Как именно считать “один шаг” и “два полушага” для переменных коэффициентов

Так как оператор $\mathcal{M}(\xi)$ зависит от ξ, нужно аккуратно фиксировать точки, в которых вычисляются коэффициенты.

(i) Один полный шаг длины h_n. Используется CN-схема на интервале $[\xi_n, \xi_n + h_n]$ с оператором, взятым на середине:

$$\xi_n + \frac{h_n}{2}.$$

(ii) Два полушага длины $h_n/2$.

- Первый полушаг: от ξ_n до $\xi_n + \frac{h_n}{2}$, коэффициенты на середине

$$\xi_n + \frac{h_n}{4}.$$

- Второй полушаг: от $\xi_n + \frac{h_n}{2}$ до $\xi_n + h_n$, коэффициенты на середине

$$\xi_n + \frac{3h_n}{4}.$$

Почему это важно. Если для обоих полушагов использовать не те точки для коэффициентов, оценка ошибки может быть загрязнена не только локальной погрешностью метода, но и искусственной ошибкой из-за несогласованной аппроксимации переменных коэффициентов.

B2.10. Псевдоалгоритм одного адаптивного шага

Ниже приводится логика одного шага марша по ξ.

Вход. Известны:

$$(\xi_n,\ U^n,\ h_n).$$

Шаг 1: один полный шаг. Вычислить U^{n+1}_{full} по CN-схеме (42) на шаге h_n.

Шаг 2: два полушага.

- Выполнить CN-шаг длины $h_n/2$ от ξ_n до $\xi_n + h_n/2$;
- затем ещё один CN-шаг длины $h_n/2$ до $\xi_n + h_n$;
- получить U^{n+1}_{half}.

Шаг 3: оценка локальной ошибки.

- Вычислить ΔU^{n+1} по (43);
- вычислить η_n по (53).

Шаг 4: решение о принятии шага.

- Если $\eta_n \le 1$, шаг принимается:

  $$\xi_{n+1} = \xi_n + h_n.$$

  В качестве нового значения можно взять

  $$U^{n+1} = U^{n+1}_{\text{half}} \quad \text{или} \quad U^{n+1} = U^{n+1}_{\text{rich}}.$$

- Если $\eta_n > 1$, шаг отклоняется:

  $$U^n \text{ не меняется,}$$

  а шаг h_n уменьшается и шаг пересчитывается.

Шаг 5: выбор следующего шага. После принятого шага вычислить h_{n+1} по (58).

B2.11. Связь адаптивности по ξ с ошибкой по s

Адаптивный шаг по ξ контролирует только часть ошибки:

- он уменьшает локальную ошибку интегрирования по ξ;
- но *не контролирует* автоматически ошибку дискретизации по s.

Что это значит practically. Даже если адаптивный шаг по ξ работает идеально, общая ошибка может оставаться большой, если:

- сетка по s слишком грубая;
- решение имеет высокие пространственные частоты вдоль контура;
- коэффициенты геометрии меняются резко.

Вывод. Нужен двухуровневый протокол:

1. сначала подобрать/проверить достаточное N_s (сходимость по s);
2. затем включать адаптивность по ξ для оптимизации времени расчёта.

Именно поэтому в вычислительном разделе (дальше) отдельно строятся графики:

- ошибка vs N_s,
- ошибка vs число шагов по ξ (или средний шаг),
- ошибка vs время расчёта.

B2.12. Диагностика "машинной точности" и предела сгущения шага

Одна из ключевых целей — найти режим, где уменьшение шага уже не улучшает результат. Для этого адаптивный алгоритм нужно сопровождать диагностикой округления.

Признаки выхода на предел округления.

- шаг h_n становится очень маленьким, но ошибка относительно эталона больше не уменьшается;
- оценка локальной ошибки по step-doubling перестаёт вести себя как $O(h_n^3)$;
- результаты начинают заметно зависеть от типа арифметики (double vs long double).

Практический протокол. Для эталонного теста (например, окружность):

1. запустить расчёт с последовательным ужесточением (ATOL, RTOL);
2. строить графики:

   ошибка vs время, ошибка vs средний шаг,

3. сравнить результаты в double и long double;
4. зафиксировать диапазон допусков, где достигается "оптимум": дальнейшее ужесточение даёт только рост времени без заметного улучшения точности.

Почему это важно. Именно этот пункт превращает численный метод в *инструмент*: появляется не просто "схема", а правило, как выбирать настройки автоматически и не переплачивать за лишние вычисления.

B2.13. Вычислительная стоимость адаптивного шага

Стоимость step-doubling. Один адаптивный шаг требует:

- 1 решение системы для полного шага,
- 2 решения систем для двух полушагов.

То есть *три* решения циклической трёхдиагональной системы на одну попытку шага.

Почему это всё равно выгодно. Несмотря на удорожание одного шага, адаптивность часто уменьшает общее число шагов, потому что:

- в гладких участках можно брать шаг значительно больше;
- мелкий шаг используется только там, где он действительно нужен.

Инженерные варианты оптимизации (дальше, в реализации).

- Использовать адаптивную оценку не на каждом шаге, а периодически (например, каждые m шагов).
- Использовать step-doubling только в "режиме калибровки" для подбора шага, а затем переходить на фиксированный шаг.
- Параллелить независимые запуски по параметрам (разные k, разные начальные профили и т.д.).

B2.14. Что сохранять в логах расчёта (важно для воспроизводимости)

Для каждого принятого шага полезно сохранять:

1. номер шага и значение ξ_n;
2. шаг h_n;
3. индикатор ошибки η_n;
4. норму $\|\Delta U^{n+1}\|$;
5. тип принятия (accepted/rejected);
6. время шага;
7. при необходимости — нормы R_{PE} и R_{full} (см. A2).

Почему это важно. Эти данные позволяют:

- анализировать, где расчёт "тормозит";
- понимать, на каких участках геометрии требуется сгущение шага;
- строить воспроизводимые графики для отчёта и диссертации.

B2.15. Итог по разделу B2

В этом разделе построен полный механизм адаптивного контроля шага по ξ:

1. введена CN-схема на переменном шаге;
2. введена апостериорная оценка ошибки через “один шаг” vs “два полушага”;
3. получена оценка Ричардсона для CN:

$$\|e_{\text{loc}}\| \approx \frac{1}{3}\|\Delta U\|;$$

4. сформулированы критерии принятия/отклонения шага;
5. выведена формула автоматического выбора следующего шага;
6. зафиксированы практические правила логирования и диагностики предела точности.

Что должно быть понятно после B2. После этого раздела должно быть полностью понятно:

1. как именно оценивается локальная ошибка шага по ξ;
2. почему для CN появляется коэффициент $\frac{1}{3}$;
3. как работает адаптивный выбор шага;
4. почему адаптивность по ξ не заменяет проверку сходимости по s;
5. как практически искать режим “почти машинной точности”.

B3. Быстрый линейный алгебраический сердечник: циклическая трёхдиагональная СЛАУ, сложность и батч-режим

В этом разделе подробно описывается вычислительное “ядро” метода: решение линейной системы, возникающей на каждом шаге схемы Кранка–Николсона. Именно этот блок определяет основную стоимость расчёта и является ключевой точкой ускорения при переходе к C++/SIMD/GPU.

Зачем нужен этот раздел. В разделе B1 получена CN-схема, которая на каждом шаге по ξ требует решения циклической трёхдиагональной системы. Если решать такие системы “в лоб” (например, через плотную матрицу), стоимость становится слишком большой. Поэтому нужен специальный решатель, который:

- использует структуру матрицы,
- работает за $O(N_s)$ на шаг,
- допускает батч-режим (много правых частей),
- устойчиво работает с комплексными коэффициентами.

B3.1. Линейная система на шаге CN и её структура

Из B1 на шаге $\xi_n \to \xi_{n+1}$ схема Кранка–Николсона имеет вид

$$\left(I - \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^{n+1} = \left(I + \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^n. \tag{59}$$

Обозначим

$$A^{(n)} U^{n+1} = b^{(n)}, \tag{60}$$

где

$$A^{(n)} := I - \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}, \tag{61}$$

$$b^{(n)} := \left(I + \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^n. \tag{62}$$

Покомпонентная структура. Из формул B1 следует, что в j-й строке матрицы $A^{(n)}$ присутствуют только три "локальных" коэффициента (при $U_{j-1}^{n+1}, U_j^{n+1}, U_{j+1}^{n+1}$), но из-за периодичности по s дополнительно появляются угловые связи:

$$j = 0 \iff j = N_s - 1.$$

Поэтому $A^{(n)}$ — *циклическая трёхдиагональная матрица* (complex cyclic tridiagonal).

Почему это важно. Циклическая трёхдиагональная матрица не является обычной трёхдиагональной: стандартная прогонка (Thomas) напрямую к ней не применяется. Однако она отличается от трёхдиагональной матрицы только двумя угловыми элементами, и это позволяет решить задачу быстро через low-rank поправку.

B3.2. Обозначения для коэффициентов матрицы

Для текущего шага n введём компактные обозначения коэффициентов матрицы $A^{(n)}$.

Локальные коэффициенты (без циклических углов). Пусть

$$a_j^{(n)} := -\frac{h_n}{2}\, p_j^{n+\frac{1}{2}}, \tag{63}$$

$$d_j^{(n)} := 1 - \frac{h_n}{2}\, q_j^{n+\frac{1}{2}}, \tag{64}$$

$$c_j^{(n)} := -\frac{h_n}{2}\, r_j^{n+\frac{1}{2}}, \tag{65}$$

где $p_j^{n+\frac{1}{2}}, q_j^{n+\frac{1}{2}}, r_j^{n+\frac{1}{2}}$ определены в B1.

Тогда (если временно забыть о периодичности) матрица была бы обычной трёхдиагональной:

$$\mathrm{tridiag}(a_j, d_j, c_j).$$

Угловые (циклические) коэффициенты. Из периодических индексов появляются два дополнительных элемента:

- в первой строке — коэффициент при $U_{N_s-1}^{n+1}$,
- в последней строке — коэффициент при U_0^{n+1}.

Обозначим их:

$$\alpha_c^{(n)} := a_0^{(n)}, \qquad \beta_c^{(n)} := c_{N_s-1}^{(n)}. \tag{66}$$

(Это просто переименование угловых коэффициентов, чтобы отдельно выделить циклические связи.)

Матрица в явном виде. С учётом этих обозначений:

$$A^{(n)} = \begin{pmatrix} d_0 & c_0 & 0 & \cdots & 0 & \alpha_c \\ a_1 & d_1 & c_1 & \ddots & & 0 \\ 0 & a_2 & d_2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & c_{N_s-2} & 0 \\ 0 & & \ddots & a_{N_s-1} & d_{N_s-1} & \\ \beta_c & 0 & \cdots & 0 & a_{N_s-1} & d_{N_s-1} \end{pmatrix},$$

где для краткости индексы (n) опущены.

Замечание о записи. В явной матрице выше важно не визуальное расположение элементов, а структура:

- обычная трёхдиагональная часть,
- плюс два угловых элемента $(1, N_s)$ и $(N_s, 1)$.

Именно эту структуру далее используем для быстрого решения.

B3.3. Разложение "трёхдиагональная часть + rank-2 поправка"

Пусть $N := N_s$ для краткости. Введём стандартные базисные векторы в
Трёхдиагональная часть. Пусть $T^{(n)}$ — матрица, полученная из $A^{(n)}$ занулением двух угловых элементов:

$$T^{(n)} := A^{(n)} - \alpha_c^{(n)} e_1 e_N^T - \beta_c^{(n)} e_N e_1^T. \tag{67}$$

Тогда $T^{(n)}$ — обычная трёхдиагональная матрица.

Rank-2 представление. Матрицу $A^{(n)}$ можно записать как

$$A^{(n)} = T^{(n)} + U^{(n)} (V^{(n)})^T, \tag{68}$$

где

$$U^{(n)} := \begin{pmatrix} \alpha_c^{(n)} e_1 & \beta_c^{(n)} e_N \end{pmatrix}, \qquad V^{(n)} := \begin{pmatrix} e_N & e_1 \end{pmatrix}. \tag{69}$$

Здесь $U^{(n)}, V^{(n)} \in$
Почему это полезно. Поправка $U^{(n)}(V^{(n)})^T$ имеет ранг не больше 2. Значит, вместо решения циклической системы можно:

- решать несколько обычных трёхдиагональных систем,
- затем сделать маленькую поправку размерности 2×2.

Это и даёт линейную сложность $O(N)$.

B3.4. Формула Вудбери (rank-2) для циклической системы

Для матрицы вида (68) применяется формула Вудбери:

$$(A^{(n)})^{-1} = (T^{(n)})^{-1} - (T^{(n)})^{-1}U^{(n)}\left(I_2 + (V^{(n)})^T(T^{(n)})^{-1}U^{(n)}\right)^{-1}(V^{(n)})^T(T^{(n)})^{-1}. \quad (70)$$

Пусть требуется решить

$$A^{(n)}x = b.$$

Тогда по (70) решение можно получить без обращения матриц.

Шаг 1: одно трёхдиагональное решение для правой части. Решить

$$T^{(n)}y = b. \quad (71)$$

Шаг 2: два трёхдиагональных решения для rank-2 поправки. Решить

$$T^{(n)}z_1 = \alpha_c^{(n)}e_1, \quad (72)$$
$$T^{(n)}z_2 = \beta_c^{(n)}e_N. \quad (73)$$

Соберём матрицу

$$Z := \begin{pmatrix} z_1 & z_2 \end{pmatrix} \in$$

Шаг 3: маленькая система 2×2. Определим

$$G := I_2 + (V^{(n)})^T Z = I_2 + \begin{pmatrix} e_N^T z_1 & e_N^T z_2 \\ e_1^T z_1 & e_1^T z_2 \end{pmatrix}. \quad (75)$$

Также вычислим

$$g := (V^{(n)})^T y = \begin{pmatrix} e_N^T y \\ e_1^T y \end{pmatrix} = \begin{pmatrix} y_N \\ y_1 \end{pmatrix}. \quad (76)$$

Затем решить маленькую систему

$$Gc = g, \qquad c \in$$

Шаг 4: финальная поправка. Тогда решение исходной циклической системы:

$$\boxed{x = y - Zc.} \quad (78)$$

Почему это быстро. Все "большие" вычисления сведены к *трём* решениям трёхдиагональной системы (одна и та же матрица $T^{(n)}$), плюс одна система 2×2. Это даёт стоимость $O(N)$ на шаг.

B3.5. Решение трёхдиагональной системы: комплексная прогонка (Thomas)

В формулах B3.4 требуется решать системы вида

$$Tx = f,$$

где T — трёхдиагональная матрица с комплексными коэффициентами:

$$T = \mathrm{tridiag}(a_j, d_j, c_j), \qquad a_j, d_j, c_j \in \acute{}$$

Почему можно использовать прогонку. Для трёхдиагональной матрицы стандартный метод Thomas работает за $O(N)$ и требует только одномерные массивы коэффициентов. Он одинаково применим для вещественных и комплексных коэффициентов.

Схема прогонки (в общем виде). Для полноты записи используем индексацию $j = 1, \dots, N$.

- Прямой ход:

$$\tilde{c}_1 = \frac{c_1}{d_1}, \tag{79}$$

$$\tilde{f}_1 = \frac{f_1}{d_1}, \tag{80}$$

  а для $j = 2, \dots, N$:

$$m_j := d_j - a_j \tilde{c}_{j-1}, \tag{81}$$

$$\tilde{c}_j := \frac{c_j}{m_j} \quad (j = 2, \dots, N-1), \tag{82}$$

$$\tilde{f}_j := \frac{f_j - a_j \tilde{f}_{j-1}}{m_j}. \tag{83}$$

- Обратный ход:

$$x_N = \tilde{f}_N, \tag{84}$$

$$x_j = \tilde{f}_j - \tilde{c}_j x_{j+1}, \qquad j = N-1, \dots, 1. \tag{85}$$

Практическая оговорка. Для устойчивой работы нужно контролировать знаменатели m_j в (81): если $|m_j|$ слишком мало, прогонка становится неустойчивой. В таких случаях требуется:

- либо более устойчивый трёхдиагональный решатель (с частичным выбором ведущего элемента),
- либо изменение шага h_n,
- либо дополнительный контроль параметров схемы.

B3.6. Численная устойчивость и практические проверки решателя

Так как матрица $A^{(n)}$ комплексная и зависит от шага h_n, важно контролировать устойчивость решения.

Минимальный набор проверок на каждом шаге.

1. Проверка малых знаменателей в прогонке:

$$|m_j| > \varepsilon_{\text{pivot}}$$

   для заранее заданного порога $\varepsilon_{\text{pivot}}$.

2. Проверка обратимости маленькой матрицы G:

$$|\det G| > \varepsilon_{2\times 2}.$$

3. Контроль невязки линейной системы: после получения x вычислить

$$r_{\text{lin}} := A^{(n)}x - b, \tag{86}$$

и контролировать

$$\frac{\|r_{\text{lin}}\|}{\|b\| + \varepsilon} \tag{87}$$

(например, в ℓ_2 или ℓ_∞-норме).

Почему это важно. Даже если сама PDE-схема корректна, ошибки линейного решателя могут испортить:

- оценку сходимости,
- адаптивный контроль шага,
- диагностику округлительной ошибки.

Поэтому невязка (87) должна логироваться отдельно.

B3.7. Сложность алгоритма и оценка памяти

Один шаг, одна правая часть. Для решения циклической системы методом B3.4 требуется:

- 1 прогонка для y (правая часть b),
- 2 прогонки для z_1, z_2 (rank-2 поправка),
- 1 решение системы 2×2,
- несколько векторных операций длины N_s.

Итого:

$$\boxed{\text{стоимость одного шага по } \xi \text{ для одной правой части} = O(N_s).} \tag{88}$$

Память. Требуется хранить:

- коэффициенты трёхдиагональной части (a_j, d_j, c_j),
- текущий вектор решения U^n,
- правую часть b,
- временные векторы y, z_1, z_2.

То есть память также линейна:

$$\boxed{\text{память} = O(N_s).} \tag{89}$$

Почему это критично. Линейная сложность и линейная память делают метод масштабируемым для больших N_s и позволяют далее переходить к батч-режиму и ускорению на C++/GPU.

B3.8. Батч-режим: много правых частей при одной и той же матрице

Во многих задачах требуется выполнять *много независимых расчётов*:

- разные начальные профили $u_0(s)$,
- разные параметры возбуждения,
- набор тестов верификации,
- sweep по параметрам (например, серия запусков).

Если на данном шаге $\xi_n \to \xi_{n+1}$ матрица $A^{(n)}$ одинакова для всех запусков (одна геометрия, один k, один и тот же шаг h_n), можно решать систему в батч-режиме.

Что можно переиспользовать. Для фиксированной матрицы $T^{(n)}$:

- коэффициенты прогонки (модифицированные коэффициенты прямого хода),
- векторы z_1, z_2 из (72)–(73),
- маленькую матрицу G и её обратную (или LU-разложение 2×2).

Что остаётся делать для каждой новой правой части b_m. Только:

1. одна прогонка $T^{(n)} y_m = b_m$,
2. вычисление $g_m = (y_{m,N}, y_{m,1})^T$,
3. решение 2×2 системы $G c_m = g_m$,
4. поправка $x_m = y_m - Z c_m$.

Итоговая стоимость батча. Пусть на шаге нужно решить систему для M правых частей. Тогда:

$$\boxed{\text{стоимость} = O(N_s)\ (\text{подготовка})\ +\ O(MN_s)\ (\text{решения}).} \tag{90}$$

Это существенно лучше, чем собирать и факторизовать матрицу заново для каждого запуска.

B3.9. SIMD/CPU-реализация: что именно ускорять

При переходе к C++ основная цель — ускорить “горячие” циклы:

- сборку коэффициентов матрицы,
- вычисление правой части $b^{(n)}$,
- прогонки (прямой и обратный ход),
- батч-операции над несколькими правыми частями.

Практические рекомендации по данным.

1. **Непрерывные массивы в памяти.** Хранить коэффициенты и векторы в непрерывных массивах:
$$a[\,],\ d[\,],\ c[\,],\ U[\,],\ b[\,].$$
2. **Комплексная арифметика.** Возможны два варианта:
   - `std::complex<double>` (удобнее, но иногда медленнее),
   - раздельные массивы для real/imag (лучше для ручного SIMD).
3. **Переиспользование буферов.** Не выделять память внутри шага: все временные массивы (y, z_1, z_2, рабочие коэффициенты прогонки) выделяются один раз.
4. **Батч-векторизация.** Если решается много правых частей, выгодно векторизовать не по индексу j, а по номеру правой части (SIMD "поперёк батча").

Почему это важно. Для больших серий тестов именно память и организация циклов часто важнее, чем формальная асимптотика: плохая локальность памяти легко съедает весь выигрыш.

B3.10. GPU/параллелизм: что можно распараллелить

Тривиальный параллелизм (между запусками). Независимые расчёты (разные параметры, разные начальные профили) полностью независимы:

$$\mathrm{run}_1,\ \mathrm{run}_2,\ \ldots,\ \mathrm{run}_M.$$

Это самый простой и надёжный источник ускорения:

- многопоточность на CPU,
- распределённые запуски,
- GPU-батч при достаточно большом числе задач.

Внутри одного шага. Сама прогонка плохо параллелится (из-за рекуррентной структуры), но есть варианты:

- parallel cyclic reduction (PCR),
- hybrid Thomas + PCR,
- batched tridiagonal solvers на GPU.

Практический приоритет. Для первой рабочей версии обычно эффективнее:

1. сделать очень быстрый CPU-решатель $O(N_s)$,
2. добавить батч-режим,
3. затем уже переносить на GPU.

Это даёт быстрый и контролируемый результат без лишней сложности.

B3.11. Частный случай окружности: дополнительное ускорение через FFT

Для окружности (см. B1.10):

$$\rho(s) \equiv R, \qquad \rho_s = 0,$$

коэффициенты не зависят от s, и оператор по s становится *с постоянными коэффициентами*. В этом случае матрица шага по s имеет циклическую Toeplitz-структуру.

Следствие. Оператор диагонализуется в базисе Фурье по s. Тогда один шаг по ξ можно выполнять:

1. FFT по s,
2. независимое обновление каждой моды,
3. обратное FFT.

Сложность. Такой шаг имеет стоимость

$$O(N_s \log N_s),$$

что асимптотически хуже, чем $O(N_s)$ прогонка, но на практике может быть очень эффективным (особенно при использовании оптимизированных библиотек FFT и/или когда нужен спектральный контроль).

Почему это важно в диссертационном контексте. FFT-вариант для окружности удобен как:

- независимая проверка корректности CN-реализации,
- высокоточный эталон для диагностики дискретизационной ошибки,
- база для сравнения “разностный метод vs спектральный метод”.

B3.12. Логирование производительности и воспроизводимость

Чтобы численный метод считался полноценным инструментом, нужно логировать не только ошибку, но и ресурсы.

Что сохранять для каждого запуска.

1. N_s, число шагов по ξ, тип шага (фиксированный/адаптивный);
2. тип арифметики (double, long double, mp);
3. тип решателя (cyclic Thomas+Woodbury, FFT, и т.д.);
4. общее время расчёта;
5. среднее/максимальное время одного шага;
6. число отклонённых шагов (если включена адаптивность);
7. нормы невязки линейного решателя (87).

Почему это важно. Без логов производительности невозможно:

- честно сравнивать численные методы,
- искать “узкие места” в реализации,
- обосновывать переход к C++/GPU.

B3.13. Минимальный псевдоалгоритм шага CN с быстрым циклическим решателем

Ниже фиксируется вычислительная логика одного шага по ξ.

Вход. Известны:

$$U^n, \quad h_n, \quad \xi_n, \quad \rho(s_j), \rho_s(s_j).$$

Шаг 1: сборка коэффициентов.

1. Вычислить коэффициенты $\alpha_j, \beta_j, \gamma_j$ на полуслое $\xi_{n+\frac{1}{2}}$.
2. Вычислить p_j, q_j, r_j.
3. Собрать коэффициенты a_j, d_j, c_j матрицы $A^{(n)}$ и правой части $b^{(n)}$.

Шаг 2: выделить трёхдиагональную часть и углы.

1. Сформировать $T^{(n)}$ (обнулить два угловых элемента).
2. Сформировать $\alpha_c^{(n)}, \beta_c^{(n)}$.

Шаг 3: подготовка rank-2 поправки (если матрица новая).

1. Решить $Tz_1 = \alpha_c e_1$, $Tz_2 = \beta_c e_N$.
2. Собрать G и его факторизацию.

Шаг 4: решение для текущей правой части.

1. Решить $Ty = b$.
2. Вычислить $g = (y_N, y_1)^T$.
3. Решить $Gc = g$.
4. Получить $U^{n+1} = y - Zc$.

Шаг 5: контроль качества линейного шага.

1. При необходимости вычислить невязку r_{lin}.
2. Записать метрики в лог.

B3.14. Итог по разделу B3

В этом разделе построен быстрый линейный алгебраический сердечник для шага CN:

1. матрица шага распознана как циклическая трёхдиагональная;
2. получено rank-2 представление через трёхдиагональную часть;
3. применена формула Вудбери (через 3 прогонки + систему 2×2);
4. показана линейная сложность $O(N_s)$ и линейная память;
5. описан батч-режим с переиспользованием факторизации;
6. зафиксированы практические требования к устойчивости и логированию.

Что должно быть понятно после B3. После этого раздела должно быть полностью понятно:

1. почему шаг CN сводится к циклической трёхдиагональной СЛАУ;
2. как именно решать эту систему за $O(N_s)$;
3. почему rank-2 поправка даёт корректное и быстрое решение;
4. что именно нужно ускорять в C++-реализации;
5. как организовать батч-вычисления и контроль устойчивости.

B4. Другие численные методы для маршевой задачи и протокол их сравнения

В этом разделе рассматриваются альтернативные численные подходы для решения маршевой задачи по ξ и фиксируется единый протокол сравнения методов. Цель раздела — не просто перечислить методы, а показать:

- какие из них применимы в данной постановке,
- как они записываются в тех же обозначениях,
- в каких режимах они могут быть лучше/хуже схемы Кранка–Николсона (CN),
- как сравнивать их корректно (по точности, устойчивости и времени).

Почему это важно. Схема CN является хорошей базовой схемой, но для полноценного вычислительного инструмента необходимо исследовать и другие подходы:

- спектральные методы (особенно для окружности и периодических задач),
- split-step / operator splitting,
- экстраполяционные и более высокие порядки по ξ,
- гибридные схемы (например, спектрально по s, неявно по ξ).

Именно это позволяет обосновать выбор “рабочей” схемы, а не просто использовать одну реализацию.

B4.1. Единая операторная запись (для всех методов)

Из B1 получена маршевая форма:

$$u_\xi = \alpha(\xi, s)\, u_{ss} + \beta(\xi, s)\, u_s + \gamma(\xi, s)\, u. \tag{91}$$

Для сравнения методов удобно ввести операторное разбиение:

$$\mathcal{M}(\xi) = \mathcal{D}(\xi) + \mathcal{C}(\xi) + \mathcal{R}(\xi), \tag{92}$$

где

$$\mathcal{D}(\xi)u := \alpha(\xi, s)\, u_{ss}, \tag{93}$$
$$\mathcal{C}(\xi)u := \beta(\xi, s)\, u_s, \tag{94}$$
$$\mathcal{R}(\xi)u := \gamma(\xi, s)\, u. \tag{95}$$

Тогда

$$u_\xi = \mathcal{M}(\xi)u. \tag{96}$$

Смысл разбиения.

- $\mathcal{D}$ — "второпроизводная" часть (обычно наиболее жёсткая);
- $\mathcal{C}$ — "конвективная" часть по s;
- $\mathcal{R}$ — локальная (реакционная/фазовая) часть.

Это разбиение естественно для построения split-step схем и IMEX-подходов.

B4.2. Базовый метод: CN как опорная схема сравнения

Схема CN уже подробно выведена в B1–B3 и здесь используется как *базовый эталон среди схем*:

$$\left(I - \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^{n+1} = \left(I + \frac{h_n}{2}\mathcal{M}^{n+\frac{1}{2}}\right) U^n. \tag{97}$$

Почему именно CN берётся как базовая.

- второй порядок по ξ;
- хорошая устойчивость для жёстких членов;
- естественная работа с комплексными коэффициентами;
- матрица шага имеет специальную структуру (циклическая трёхдиагональная), что даёт быстрый решатель (B3).

Что сравнивается относительно CN. Для каждого альтернативного метода далее сравниваются:

- точность при фиксированных ресурсах,
- скорость при заданной точности,
- устойчивость на высоких частотах и больших расстояниях марша,
- чувствительность к геометрии (переменная кривизна).

B4.3. Спектральный метод по s (Fourier) для периодической координаты

Так как s — периодическая координата вдоль замкнутого контура, естественно использовать разложение по ряду Фурье:

$$u(\xi, s) = \sum_{m=-M}^{M} \widehat{u}_m(\xi) e^{i\kappa_m s}, \qquad \kappa_m := \frac{2\pi m}{L}. \tag{98}$$

Производные в спектральном виде. Для каждой моды:

$$\partial_s \left(e^{i\kappa_m s}\right) = i\kappa_m e^{i\kappa_m s}, \tag{99}$$

$$\partial_{ss} \left(e^{i\kappa_m s}\right) = -\kappa_m^2 e^{i\kappa_m s}. \tag{100}$$

Следовательно,

$$u_s(\xi, s) = \sum_{m=-M}^{M} i\kappa_m \widehat{u}_m(\xi) e^{i\kappa_m s}, \tag{101}$$

$$u_{ss}(\xi, s) = \sum_{m=-M}^{M} (-\kappa_m^2) \widehat{u}_m(\xi) e^{i\kappa_m s}. \tag{102}$$

Почему это полезно. Спектральная аппроксимация по s:

- очень точна для гладких функций (ошибка убывает быстрее любой степени при достаточной гладкости),
- идеально подходит для периодических задач,
- особенно эффективна для окружности (коэффициенты не зависят от s).

B4.4. Спектральный метод для окружности: модовая форма и точное обновление

В случае окружности (см. B1.10):

$$\rho(s) \equiv R, \qquad \rho_s(s) = 0,$$

и уравнение упрощается до

$$u_\xi = \alpha(\xi)\, u_{ss} + \gamma(\xi)\, u, \tag{103}$$

где $\alpha(\xi)$ и $\gamma(\xi)$ не зависят от s.

Подставим ряд Фурье (98). Для каждой моды m получаем ОДУ:

$$\frac{d\widehat{u}_m}{d\xi} = \lambda_m(\xi)\, \widehat{u}_m(\xi), \qquad \lambda_m(\xi) := -\alpha(\xi)\kappa_m^2 + \gamma(\xi). \tag{104}$$

Точное решение моды (в непрерывном виде). На отрезке $[\xi_n, \xi_{n+1}]$:

$$\widehat{u}_m(\xi_{n+1}) = \exp\left(\int_{\xi_n}^{\xi_{n+1}} \lambda_m(\tau)\, d\tau\right) \widehat{u}_m(\xi_n). \tag{105}$$

Практическая формула второго порядка (серединная квадратура). Если интеграл в (105) не брать аналитически, то можно использовать серединную квадратуру:

$$\int_{\xi_n}^{\xi_{n+1}} \lambda_m(\tau)\, d\tau = h_n\, \lambda_m(\xi_{n+\frac{1}{2}}) + O(h_n^3),$$

откуда

$$\widehat{u}_m^{n+1} = \exp\bigl(h_n \lambda_m(\xi_{n+\frac{1}{2}})\bigr)\, \widehat{u}_m^n + O(h_n^3). \tag{106}$$

Почему это очень важный метод. Для окружности формула (106) даёт:

- очень точный и быстрый шаг,
- независимый контроль качества CN-схемы,
- удобный спектральный эталон (кроме "точного" решения через ряд/специальные функции).

Сложность шага. Один шаг по ξ через FFT:

1. FFT: $u^n(s_j) \mapsto \widehat{u}_m^n$,
2. модовое обновление (106),
3. обратное FFT.

Итоговая стоимость:

$$O(N_s \log N_s). \tag{107}$$

B4.5. Псевдоспектральный подход для переменной кривизны

Для общей геометрии коэффициенты $\alpha(\xi, s), \beta(\xi, s), \gamma(\xi, s)$ зависят от s, поэтому оператор не диагонализуется в базисе Фурье. Однако спектральные производные по s всё равно можно использовать:

$$u_s \approx \mathcal{F}^{-1}\bigl(i\kappa_m \widehat{u}_m\bigr), \tag{108}$$

$$u_{ss} \approx \mathcal{F}^{-1}\bigl(-\kappa_m^2 \widehat{u}_m\bigr). \tag{109}$$

Идея псевдоспектрального шага. На каждом слое:

1. по текущему U вычислить u_s и u_{ss} через FFT/обратное FFT;
2. умножить на коэффициенты $\alpha(\xi, s_j), \beta(\xi, s_j), \gamma(\xi, s_j)$ в физическом пространстве;
3. собрать правую часть или оператор для шага по ξ.

Преимущества.

- высокая точность по s для гладких решений;
- удобно для гладких контуров и периодических данных;
- полезно как "вторая семья" методов для сравнения с разностным CN.

Ограничения.

- при переменных коэффициентах не получается простая диагональная формула для шага;
- неявные схемы могут приводить к плотным матрицам (или требуют итерационных методов);
- при негладких профилях возможны спектральные артефакты (эффект Гиббса).

Вывод. Псевдоспектральный подход особенно полезен:

- как высокоточный референс по s для гладких тестов,
- как сравнение с разностной аппроксимацией по s,
- как база для гибридных схем (см. ниже).

B4.6. Split-step (operator splitting): общая идея

Вместо решения полного оператора $\mathcal{M} = \mathcal{D} + \mathcal{C} + \mathcal{R}$ за один шаг можно расщепить шаг на несколько подшагов:

- отдельно для "диффузионно-дисперсионной" части $\mathcal{D}$,
- отдельно для "конвективно-реакционной" части $\mathcal{C} + \mathcal{R}$.

Для фиксированного слоя (коэффициенты "заморожены" на полушаге $\xi_{n+\frac{1}{2}}$) обозначим:

$$L_n := \mathcal{D}^{n+\frac{1}{2}}, \qquad N_n := \mathcal{C}^{n+\frac{1}{2}} + \mathcal{R}^{n+\frac{1}{2}}. \tag{110}$$

Тогда на шаге:

$$u_\xi = (L_n + N_n)u.$$

Почему разделение может быть выгодно.

- часть L_n часто "жёсткая" и требует неявного шага;
- часть N_n может считаться быстрее (или даже аналитически в простых случаях);
- для окружности split-step можно реализовать очень эффективно через FFT.

B4.7. Lie splitting и Strang splitting (формулы шага)

Lie splitting (1-й порядок по ξ). На шаге длины h_n:

$$U^{n+1} \approx e^{h_n L_n}\, e^{h_n N_n}\, U^n \quad \text{или} \quad e^{h_n N_n}\, e^{h_n L_n}\, U^n. \tag{111}$$

Локальная ошибка — порядка $O(h_n^2)$, глобальная — порядка $O(h_n)$.

Strang splitting (2-й порядок по ξ).

$$\boxed{U^{n+1} \approx e^{\frac{h_n}{2} L_n}\, e^{h_n N_n}\, e^{\frac{h_n}{2} L_n}\, U^n.} \tag{112}$$

Локальная ошибка — порядка $O(h_n^3)$, глобальная — порядка $O(h_n^2)$.

Почему Strang важнее. Поскольку CN имеет второй порядок, сравнивать разумно именно с методом второго порядка. Поэтому в вычислительных тестах основной split-step кандидат — это Strang (112).

B4.8. Как реализовать подшаги в split-step

Формулы (111)–(112) записаны через операторные экспоненты. На практике их нужно аппроксимировать.

Подшаг для $L_n = \mathcal{D}^{n+\frac{1}{2}}$ (вторая производная). Можно использовать:

- **неявный CN-подшаг** только для L_n:

$$\left(I - \frac{\tau}{2}L_n\right) V^+ = \left(I + \frac{\tau}{2}L_n\right) V, \tag{113}$$

  где $\tau = h_n$ или $\tau = h_n/2$;

- **FFT-подшаг** (если коэффициенты по s постоянны или почти постоянны).

Подшаг для $N_n = \mathcal{C}^{n+\frac{1}{2}} + \mathcal{R}^{n+\frac{1}{2}}$. Возможные варианты:

- явный шаг (быстро, но менее устойчиво),
- неявный/полунеявный шаг,
- отдельное расщепление $\mathcal{C} + \mathcal{R}$,
- для окружности: так как $\beta \equiv 0$, остаётся только $\mathcal{R}$, и подшаг реализуется *точно* как умножение:

$$V^+ = e^{\tau\gamma(\xi_{n+\frac{1}{2}})} V. \tag{114}$$

Практически важный частный случай (окружность). Для окружности split-step Strang особенно прост:

$$U^{n+1} \approx e^{\frac{h_n}{2}\mathcal{D}_n}\, e^{h_n\mathcal{R}_n}\, e^{\frac{h_n}{2}\mathcal{D}_n} U^n, \tag{115}$$

причём:

- шаги с $\mathcal{D}_n$ удобны в Фурье-пространстве,
- шаг с $\mathcal{R}_n$ — просто умножение на скаляр.

Это делает split-step очень сильным конкурентом для окружности.

B4.9. Экстраполяционные методы на базе CN (повышение порядка по ξ)

В B2 уже использовалась идея “один шаг vs два полушага” для оценки ошибки. Эту же идею можно использовать для *повышения порядка* (Richardson extrapolation).

Обозначения. Пусть:

- U^{n+1}_{full} — один CN-шаг длины h_n;
- U^{n+1}_{half} — два CN-полушага длины $h_n/2$.

Экстраполированное значение (порядок выше). Для метода второго порядка по ξ:

$$\boxed{U_{\text{ext}}^{n+1} = U_{\text{half}}^{n+1} + \frac{U_{\text{half}}^{n+1} - U_{\text{full}}^{n+1}}{2^2 - 1} = U_{\text{half}}^{n+1} + \frac{1}{3}\Delta U^{n+1}.} \tag{116}$$

Что это даёт. При достаточной гладкости:

- U_{ext}^{n+1} имеет более высокий порядок по ξ, чем обычный CN-шаг;
- одновременно остаётся встроенная оценка ошибки (из B2);
- метод удобно использовать на эталонных задачах, где важна "почти машинная точность".

Цена. Экстраполяция требует двух полушагов + одного полного шага, то есть примерно втрое дороже обычного шага (как и в B2). Поэтому её разумно применять:

- либо в режиме верификации,
- либо адаптивно только на "сложных" участках.

B4.10. IMEX-подходы как промежуточный вариант

Ещё один класс методов — IMEX (implicit-explicit), где "жёсткая" часть берётся неявно, а "нежёсткая" — явно.

Естественное разбиение. В данной задаче можно взять:

$$\mathcal{D}u = \alpha u_{ss} \quad \text{(неявно)}, \qquad (\mathcal{C} + \mathcal{R})u = \beta u_s + \gamma u \quad \text{(явно)}.$$

Простейшая IMEX-CN/AB-схема (пример). Один из возможных вариантов:

$$\frac{U^{n+1} - U^n}{h_n} = \frac{1}{2}L_n(U^{n+1} + U^n) + \left(\frac{3}{2}N_n U^n - \frac{1}{2}N_{n-1}U^{n-1}\right), \tag{117}$$

где $L_n \approx \mathcal{D}^{n+\frac{1}{2}}$, $N_n \approx (\mathcal{C} + \mathcal{R})^{n+\frac{1}{2}}$.

Зачем этот класс методов рассматривать.

- может уменьшить стоимость шага (не весь оператор идёт в неявную часть);
- сохраняет устойчивость по "жёсткой" второй производной;
- полезен при переходе к более сложным 3D-задачам.

Почему это не первая схема для реализации.

- появляется более сложный анализ устойчивости;
- нужно хранить дополнительные слои (U^{n-1});
- для текущего этапа (2D + верификация) CN и спектральные методы проще и надёжнее.

B4.11. Какие методы реально сравнивать на текущем этапе (2D)

Чтобы не раздувать объём вычислительной части, на этапе 2D целесообразно сравнивать следующий набор.

Обязательный набор (ядро сравнения).

1. **CN + разностная аппроксимация по** s (база, B1–B3).
2. **CN + Richardson/адаптивность** (B2) — как "точная" версия той же схемы.
3. **Спектральный метод по Фурье для окружности** (B4.4) — как независимый референс.
4. **Strang split-step для окружности** (B4.8) — как альтернативная быстрая схема.

Дополнительный набор (после базовой верификации).

1. Псевдоспектральный метод для "почти окружности" / гладких контуров.
2. Один IMEX-вариант как исследовательский эксперимент.

Почему такой порядок. Сначала сравниваются методы, которые:

- проще реализовать и проверить,
- имеют ясную интерпретацию,
- дают сильный контроль точности на окружности.

Уже потом добавляются более сложные гибридные варианты.

B4.12. Единые метрики сравнения (точность + устойчивость + ресурсы)

Для всех методов используется единый набор метрик (в согласии с A2).

Точность относительно эталона.

$$\varepsilon_{L_2} = \frac{\|E_{\text{num}} - E_{\text{ref}}\|_{L_2(\Omega)}}{\|E_{\text{ref}}\|_{L_2(\Omega)}}, \tag{118}$$

$$\varepsilon_{L_\infty} = \frac{\|E_{\text{num}} - E_{\text{ref}}\|_{L_\infty(\Omega)}}{\|E_{\text{ref}}\|_{L_\infty(\Omega)}}. \tag{119}$$

Фазовая ошибка.

$$\Delta\phi(\xi, s) = \operatorname{Arg}\big(E_{\text{num}}\,\overline{E_{\text{ref}}}\big), \qquad \varepsilon_{\phi,\infty} = \|\Delta\phi\|_{L_\infty(\Omega_*)}. \tag{120}$$

Невязка уравнения.

$$\|R_{\text{PE}}[u_{\text{num}}]\|, \qquad \|R_{\text{full}}[u_{\text{num}}]\|. \tag{121}$$

Метрики вычислительных ресурсов.

- общее время расчёта;
- число шагов по ξ (и число отклонённых шагов для адаптивного режима);
- время одного шага;
- память;
- тип арифметики (double/long double).

Метрики устойчивости.

- наличие/отсутствие взрывного роста нормы;
- чувствительность к уменьшению шага;
- чувствительность к типу арифметики;
- невязка линейного решателя (для неявных методов).

B4.13. Протокол сравнения методов (что именно запускать)

Для каждого теста (окружность, почти окружность, переменная кривизна) рекомендуется одинаковый протокол.

Шаг 1. Фиксация эталона.

- Для окружности: использовать высокоточный эталон (аналитический/спектральный/табл
- Для более сложных контуров: использовать "дорогой" расчёт как reference (очень мелкая сетка, строгие допуски, long double/mp).

Шаг 2. Сходимость по s. Для каждого метода сначала провести серию расчётов с ростом N_s:

$$N_s = N_0,\ 2N_0,\ 4N_0,\ \dots$$

при достаточно малом шаге по ξ. Цель — отделить ошибку дискретизации по s.

Шаг 3. Сходимость по ξ. При фиксированном "достаточном" N_s исследовать:

- фиксированные шаги h,
- адаптивный режим (если поддерживается).

Строить log–log графики:

$$\varepsilon \text{ vs } h, \qquad \varepsilon \text{ vs число шагов}, \qquad \varepsilon \text{ vs время}.$$

Шаг 4. Проверка устойчивости на параметрах. Отдельно менять:

- волновое число k (высокочастотный режим),
- длину марша Ξ,
- форму начального профиля $u_0(s)$ (гладкий/более резкий).

Шаг 5. Диагностика округления. Для лучших конфигураций:

- сравнить double vs long double;
- проверить наличие “плато” ошибки при сгущении шага;
- зафиксировать область практического оптимума.

B4.14. Как интерпретировать результаты сравнения

Важно заранее зафиксировать, какие выводы считаются существенными.

Если CN стабильно выигрывает. Это означает, что:

- структура матрицы и быстрый решатель (B3) дают оптимальный баланс,
- метод достаточно универсален для реальных контуров,
- альтернативные методы можно оставить как вспомогательные (референс/проверка).

Если спектральный/FFT-метод выигрывает на окружности. Это нормальный и ожидаемый результат:

- окружность — “идеальная” геометрия для Фурье-подхода;
- такой метод нужно использовать как эталон и калибровку;
- это не означает автоматического выигрыша на переменной кривизне.

Если split-step даёт ту же точность быстрее. Тогда split-step можно сделать основным методом для некоторых классов задач (например, близких к окружности), а CN оставить как более универсальный.

Если методы расходятся по фазе, но не по амплитуде. Это отдельный важный сигнал:

- амплитудная ошибка может быть малой, но фазовая — существенной;
- в высокочастотных задачах фазу нужно контролировать обязательно;
- при сравнении всегда хранить и амплитудные, и фазовые метрики.

B4.15. Что включить в реализацию на текущем этапе (практический план)

Чтобы не распыляться, на текущем этапе (2D) целесообразно реализовать по порядку:

1. **CN (уже есть) + быстрый циклический решатель** — базовый рабочий инструмент.
2. **Адаптивность и Richardson** — для контроля точности и поиска оптимального шага.
3. **Спектральный метод для окружности** — независимый высокоточный референс.
4. **Strang split-step для окружности** — альтернативная быстрая схема.
5. **(Опционально) псевдоспектральный вариант для гладких контуров** — исследовательский шаг.

Почему именно так. Такой порядок:

- даёт быстрый прогресс в верификации,
- не требует сразу сложной инфраструктуры,
- позволяет честно сравнить методы на одном и том же наборе тестов.

B4.16. Итог по разделу B4

В этом разделе:

1. введена единая операторная запись для всех схем;
2. зафиксирован базовый метод CN как опорная схема сравнения;
3. подробно выписан спектральный метод по Фурье (особенно важный для окружности);
4. введены split-step схемы (Lie, Strang) и показано, как их реализовать;
5. описаны экстраполяционные и IMEX-подходы как дополнительные кандидаты;
6. зафиксирован единый протокол сравнения по точности, устойчивости и ресурсам.

Что должно быть понятно после B4. После этого раздела должно быть полностью понятно:

1. какие альтернативные методы реально уместны в пункте B;
2. как они записываются в тех же обозначениях;
3. какие методы нужно сравнивать в первую очередь;
4. по каким метрикам делать сравнение;
5. как использовать окружность как “полигон” для выбора лучшей схемы.

B5. Протокол вычислительных экспериментов, формат данных и шаблон представления результатов

В этом разделе фиксируется единый протокол вычислительных экспериментов для всех методов, рассмотренных в пункте B. Цель — сделать вычислительную часть воспроизводимой, сравнимой и пригодной для дальнейшей верификации (раздел C).

Зачем нужен этот раздел. Без строгого протокола легко получить “несравнимые” результаты:

- разные методы запускаются на разных параметрах;
- метрики считаются в разных нормах;
- часть данных не сохраняется (невозможно повторить графики);
- трудно понять, где ошибка метода, а где ошибка реализации.

Поэтому ниже фиксируется:

1. набор обязательных тестов;
2. набор обязательных параметров запуска;
3. формат сохранения результатов;
4. набор обязательных графиков и таблиц;
5. правила интерпретации результатов.

B5.1. Набор обязательных тестовых задач (2D)

На текущем этапе (2D) для верификации и сравнения методов рекомендуется использовать не менее трёх задач.

Тест 1: окружность (базовый эталон).

$$\rho(s) \equiv R = \text{const}. \tag{122}$$

Это основной эталонный тест, поскольку:

- коэффициенты упрощаются;
- существует точное/высокоточное решение;
- возможен независимый спектральный контроль (FFT/модовый подход).

Тест 2: "почти окружность" (слабая переменная кривизна). Например, в виде гладкого возмущения:

$$\rho(s) = R\big(1 + \varepsilon \cos(ms)\big), \qquad 0 < \varepsilon \ll 1. \tag{123}$$

Этот тест нужен для проверки:

- корректной работы геометрических коэффициентов;
- чувствительности метода к переменной кривизне;
- устойчивости при наличии члена $\beta(\xi, s)u_s$.

Тест 3: гладкий контур с переменной кривизной (без аналитического решения). Здесь используется "дорогой" reference-расчёт:

- очень густая сетка по s,
- строгие допуски по ξ,
- при необходимости более высокая точность арифметики (long double / mp).

Этот тест нужен, чтобы показать работоспособность метода вне окружности.

Замечание. На этапе раздела B можно ограничиться тестами 1–2, а тест 3 частично вынести в раздел C (где будет более строгая верификация и анализ ошибок относительно reference).

B5.2. Набор параметров, которые обязательно фиксировать в каждом запуске

Для каждого запуска (run) должны быть сохранены *все* параметры, влияющие на результат.

(i) Параметры физической/математической задачи.

1. Волновое число k.
2. Длина контура L (или параметры геометрии, из которых она определяется).
3. Диапазон марша:
$$0 \le \xi \le \Xi.$$
4. Параметры геометрии (например, R, ε, номер моды возмущения и т.п.).
5. Тип начального профиля $u_0(s)$ и его параметры.

(ii) Параметры численного метода.

1. Тип схемы (CN, CN+adapt, CN+Richardson, FFT, Strang split-step и т.д.).
2. Число узлов по s:
$$N_s, \qquad \Delta s = L/N_s.$$
3. Режим шага по ξ:
   - фиксированный шаг ($h = \text{const}$),
   - адаптивный шаг (с указанием ATOL, RTOL).
4. Максимальный и минимальный шаг (если адаптивный режим):
$$h_{\min},\ h_{\max}.$$
5. Тип дискретизации по s (разностная/спектральная).
6. Тип линейного решателя (cyclic tridiagonal + Woodbury, FFT-обновление, и т.д.).

(iii) Параметры реализации и воспроизводимости.

1. Тип арифметики (double, long double, arbitrary precision).
2. Версия кода / git commit hash.
3. Версия компилятора / Python / библиотек (NumPy, FFTW и т.д.).
4. Платформа (CPU/GPU, число потоков).
5. Seed (если где-то используется случайность; обычно не требуется, но лучше фиксировать).

Почему это критично. Даже небольшая разница (например, другой тип арифметики или другая FFT-библиотека) может заметно изменить результат в режимах, близких к машинной точности.

B5.3. Что считать “одним экспериментом” и как организовать серии запусков

Определение. Под *одним экспериментом* будем понимать набор запусков, в котором меняется ровно один ключевой параметр, а все остальные фиксированы.

Примеры корректных серий.

1. **Сходимость по N_s:**
$$N_s = N_0,\ 2N_0,\ 4N_0, \dots$$
при фиксированном малом шаге по ξ.

2. **Сходимость по шагу по ξ:**
$$h = h_0,\ \frac{h_0}{2},\ \frac{h_0}{4}, \dots$$
при фиксированном достаточно большом N_s.

3. **Сравнение методов:** меняется только метод (CN vs FFT vs Strang), а геометрия, k, N_s, диапазон Ξ и критерии точности — одинаковые.

Некорректный эксперимент. Нельзя одновременно менять и N_s, и шаг h, и метод, а потом делать вывод, какой именно фактор повлиял на ошибку.

Вывод. Для каждой цели (сходимость, устойчивость, производительность, сравнение методов) должна быть отдельная серия запусков с чётко фиксированным набором параметров.

B5.4. Метрики, которые должны вычисляться в каждом запуске

Метрики берутся в соответствии с разделами A2 и B4, но здесь фиксируется минимальный обязательный набор для сохранения в файлы.

(A) Точность относительно эталона. Если доступен эталон E_{ref}, обязательно сохраняются:

$$\varepsilon_{L_2} = \frac{\|E_{\mathrm{num}} - E_{\mathrm{ref}}\|_{L_2}}{\|E_{\mathrm{ref}}\|_{L_2}}, \tag{124}$$

$$\varepsilon_{L_\infty} = \frac{\|E_{\mathrm{num}} - E_{\mathrm{ref}}\|_{L_\infty}}{\|E_{\mathrm{ref}}\|_{L_\infty}}, \tag{125}$$

а также (если возможно) фазовая ошибка:

$$\varepsilon_{\phi,\infty} = \|\operatorname{Arg}(E_{\mathrm{num}}\overline{E_{\mathrm{ref}}})\|_{L_\infty(\Omega_*)}. \tag{126}$$

(B) Невязки уравнений. Даже если эталона нет, нужно сохранять:

$$\|R_{\mathrm{PE}}[u_{\mathrm{num}}]\|, \qquad \|R_{\mathrm{full}}[u_{\mathrm{num}}]\| \quad (\text{если вычисляется}). \tag{127}$$

(C) Метрики шага (для адаптивного режима). Для адаптивного шага по ξ сохраняются:

$$\eta_n, \quad h_n, \quad \|\Delta U^{n+1}\|, \quad \text{accepted/rejected}. \tag{128}$$

(D) Метрики линейного решателя. Для неявных схем:

$$\frac{\|A^{(n)}U^{n+1} - b^{(n)}\|}{\|b^{(n)}\| + \varepsilon}. \tag{129}$$

(E) Метрики производительности.

1. Общее время расчёта.
2. Время на шаг (среднее, максимум).
3. Число шагов по ξ.
4. Число отклонённых шагов (если адаптивный режим).
5. Пиковая память (если есть возможность измерить).

B5.5. Формат сохранения результатов: summary CSV и step-log CSV

Для воспроизводимости и удобства построения графиков рекомендуется сохранять результаты в двух уровнях файлов.

(i) Сводный файл по запускам: `summary.csv`. Одна строка = один запуск (run). Рекомендуемые поля:

```
run_id, method, geometry, k, Xi, Ns, ds, step_mode, h0, atol, rtol,
    n_steps, n_reject, eps_L2, eps_Linf, eps_phi, res_PE, res_full,
linres_max, time_total, time_step_mean, time_step_max, arithmetic, code_version.
```
(130)

Почему нужен именно summary. Этот файл используется для:

- построения большинства итоговых графиков;
- сравнительных таблиц по методам;
- фильтрации запусков по параметрам.

(ii) Подробный лог шага: `run_<id>_steps.csv`. Одна строка = один *попытка шага* (включая отклонённые). Рекомендуемые поля:

```
step_idx, xi_start, h, accepted, eta, deltaU_norm, linres, time_step.
```

Если нужно логировать больше:

```
xi_end, h_new, U_norm, resPE_local, resFull_local.
```

Почему важен step-log. Без него невозможно:

- анализировать поведение адаптивного шага;
- понять, где метод теряет устойчивость;
- доказать, что адаптивность действительно работает “осмысленно”.

B5.6. Формат сохранения эталона (reference) и метаданных

Для эталонных решений (особенно окружность) рекомендуется отдельный формат хранения:

- массивы значений поля (или огибающей) на фиксированной сетке;
- отдельный файл метаданных.

Что должно быть в метаданных эталона.

1. Геометрия (например, R для окружности).
2. Волновое число k.
3. Сетка, на которой сохранён эталон:
$$N_s^{\mathrm{ref}}, \quad \{\xi_m^{\mathrm{ref}}\}, \quad \{s_j^{\mathrm{ref}}\}.$$
4. Метод получения эталона (аналитика / FFT / arbitrary precision / очень густая сетка).
5. Оценка точности эталона (невязка, число мод, тип арифметики).
6. Версия формулы/кода.

Почему это важно. Иначе через некоторое время невозможно понять, *насколько* эталон был точным и можно ли ему доверять при сравнении методов.

B5.7. Набор обязательных графиков для раздела B (и перехода к C)

Ниже фиксируется минимальный набор графиков, которые должны быть построены для каждого ключевого теста (в первую очередь — для окружности).

График 1. Сходимость по N_s (log–log). По оси x:
$$N_s \quad \text{или} \quad \Delta s = L/N_s,$$
по оси y:
$$\varepsilon_{L_2}, \ \varepsilon_{L_\infty}.$$
Цель:

- увидеть порядок по s для разностной схемы;
- увидеть область, где начинает доминировать другая ошибка (по ξ или округление).

График 2. Сходимость по шагу по ξ (log–log). По оси x:
$$h \quad \text{(или средний шаг в адаптивном режиме)},$$
по оси y:
$$\varepsilon_{L_2}, \ \varepsilon_{L_\infty}.$$
Цель:

- подтвердить ожидаемый порядок по ξ (для CN — второй порядок);
- увидеть “перелом” при выходе на предел округления.

График 3. Ошибка vs время расчёта. По оси x:

$$\text{time\_total},$$

по оси y:

$$\varepsilon_{L_2} \text{ или } \varepsilon_{L_\infty}.$$

Цель:

- сравнить методы как *инструменты*, а не только как схемы;
- выбрать практический "лучший" метод при заданном бюджете времени.

График 4. Невязка vs ξ (или итоговая невязка vs параметры). Либо:

$$\|R_{\text{PE}}\|(\xi_n),$$

либо summary-график по запускам. Цель:

- контролировать качество решения независимо от эталона;
- видеть накопление ошибки вдоль марша.

График 5. Адаптивный шаг h_n как функция ξ_n (для adaptive-CN). По оси x:

$$\xi_n,$$

по оси y:

$$h_n.$$

Цель:

- показать, что шаг действительно адаптируется;
- понять, где геометрия/решение требуют сгущения.

График 6. Индикатор шага η_n (для adaptive-CN). По оси x:

$$\xi_n,$$

по оси y:

$$\eta_n.$$

Цель:

- проверить корректность настройки ATOL, RTOL;
- увидеть, нет ли систематических отклонений/перестраховки.

График 7. Double vs long double (диагностика округления). По оси x:

$$h \text{ или } N_s,$$

по оси y:

$$\varepsilon.$$

На одном графике — две кривые: double и long double. Цель:

- отделить дискретизационную ошибку от округлительной;
- определить практический предел ужесточения параметров.

B5.8. Шаблон таблиц для отчёта и диссертации

Ниже фиксируются шаблоны таблиц, которые удобно использовать в тексте.

Таблица 1. Сходимость по N_s (один метод, один тест).

$$\{N_s,\ \Delta s,\ \varepsilon_{L_2},\ \varepsilon_{L_\infty},\ \text{observed order},\ \text{time}\}. \tag{131}$$

Таблица 2. Сходимость по шагу по ξ (один метод, один тест).

$$\{h,\ N_\xi,\ \varepsilon_{L_2},\ \varepsilon_{L_\infty},\ \text{observed order},\ \text{time}\}. \tag{132}$$

Таблица 3. Сравнение методов при одинаковой целевой точности.

$$\{\text{method},\ \varepsilon_{L_2},\ \varepsilon_{L_\infty},\ \text{time},\ \text{memory},\ \text{comments}\}. \tag{133}$$

Таблица 4. Адаптивный режим (сводка).

$$\{\text{ATOL}, \text{RTOL},\ n_{\text{steps}},\ n_{\text{reject}},\ \varepsilon,\ \text{time}\}. \tag{134}$$

Практическое замечание. В таблицах всегда полезно добавлять:

- тип арифметики (double/long double),
- номер версии кода,
- ссылку на run_id (чтобы можно было быстро найти исходные логи).

B5.9. Как считать “наблюдаемый порядок” и как его записывать

При наличии серии запусков с последовательным уменьшением шага/увеличением числа узлов наблюдаемый порядок полезно вычислять автоматически.

По шагу по ξ. Если $h_{m+1} = h_m/2$, то наблюдаемый порядок:

$$p_\xi^{\text{obs}} = \log_2\left(\frac{\varepsilon(h_m)}{\varepsilon(h_{m+1})}\right). \tag{135}$$

Для CN ожидается:

$$p_\xi^{\text{obs}} \approx 2$$

(до выхода на доминирование других ошибок).

По сетке по s. Если $N_{s,m+1} = 2N_{s,m}$ (то есть Δs уменьшается в 2 раза):

$$p_s^{\text{obs}} = \log_2\left(\frac{\varepsilon(\Delta s_m)}{\varepsilon(\Delta s_{m+1})}\right). \tag{136}$$

Для центральных разностей (B1) ожидается:

$$p_s^{\text{obs}} \approx 2$$

при гладком решении и достаточно малом шаге по ξ.

Почему порядок может “ломаться”. Если наблюдаемый порядок заметно уходит от ожидаемого, возможны причины:

- доминирует ошибка по другой переменной (например, по ξ вместо s);
- доминирует модельная ошибка (параболизация);
- началось влияние округления;
- есть ошибка реализации (неверные коэффициенты, неправильные индексы, ошибки в периодичности).

B5.10. Протокол для сравнения double / long double / mp

Чтобы отделить дискретизационную ошибку от округлительной, нужен отдельный диагностический протокол.

Шаг 1. Выбрать один эталонный тест. Лучше всего — окружность с фиксированными параметрами.

Шаг 2. Зафиксировать метод. Например, CN или CN+Richardson.

Шаг 3. Провести одинаковую серию запусков в разных типах арифметики.

- серия в `double`,
- серия в `long double`,
- (по возможности) 1–2 контрольные точки в arbitrary precision.

Шаг 4. Сравнить графики ошибок. Если при уменьшении шага:

- в `double` ошибка выходит на плато,
- а в `long double` продолжает уменьшаться,

то плато вызвано округлением, а не дискретизацией.

Вывод. Именно этот протокол позволяет обосновать фразу “достигнут режим, близкий к машинной точности”.

B5.11. Контроль корректности реализации (sanity checks)

Перед массовыми сериями запусков нужно пройти минимальный набор проверок корректности.

Обязательные проверки.

1. **Проверка периодичности по s:**

$$U_0^n \leftrightarrow U_{N_s}^n \quad \text{(индексация по модулю)}.$$

2. **Проверка знаков в коэффициентах** p_j, q_j, r_j: особенно знак при первой производной (β-члене).

3. **Проверка частного случая окружности:** при $\rho_s = 0$ должно получаться

$$\beta(\xi, s) \equiv 0.$$

4. **Проверка линейного решателя:** норма невязки должна быть существенно меньше нормы решения (в разумных пределах машинной точности).

5. **Проверка сходимости на "малой" задаче:** при уменьшении шага ошибка должна убывать с ожидаемым порядком.

Почему это важно. Если не пройти эти проверки заранее, можно потратить много времени на построение графиков, которые отражают не свойства метода, а ошибки в коде.

B5.12. Именование файлов и структура каталога результатов

Для большого числа запусков важно сразу ввести понятную структуру каталогов.

Рекомендуемая структура.

- `results/`
  - `summary.csv`
  - `refs/` — эталоны и их метаданные
  - `runs/`
    - `run_0001/`
      - `config.json`
      - `steps.csv`
      - `field_final.npz`
      - `metrics.json`
    - `run_0002/`, ...
  - `plots/`

Почему это удобно.

- каждый run самодостаточен;
- легко пересобрать summary;
- легко автоматизировать построение графиков;
- удобно прикладывать к отчёту/репозиторию.

B5.13. Минимальный набор автоматических regression-tests

Чтобы код оставался стабильным при доработках, нужны короткие автоматические тесты.

Что включить в regression-tests.

1. Тест на окружности с маленькой сеткой: сравнение с заранее сохранённым результатом (до заданного допуска).
2. Тест на одну итерацию шага: проверка, что линейная невязка меньше порога.
3. Тест на адаптивный шаг: проверка, что при слишком грубом начальном шаге есть хотя бы одно отклонение.
4. Тест на сходимость: на двух сетках ошибка на более густой сетке меньше.

Почему это важно. Эти тесты нужны не только “для программистов”: в научной работе они являются частью доказательства воспроизводимости.

B5.14. Связь раздела B5 с будущим разделом C (верификация)

Раздел B5 задаёт инфраструктуру и правила экспериментов, а раздел C будет использовать их для строгой верификации.

Что переходит из B в C напрямую.

- эталоны и их метаданные;
- summary-таблицы и step-логи;
- графики сходимости и графики “ошибка–время”;
- диагностика округления (double vs long double);
- наблюдаемые порядки сходимости.

Что добавится в C.

- формальный протокол верификации (критерии “принят/не принят”);
- более строгий анализ разделения ошибок (модельная / дискретизационная / округление);
- правила выбора параметров “по умолчанию” для практического использования метода.

Вывод. Таким образом, B5 — это мост между построением методов (B1–B4) и строгой верификацией/валидацией (раздел C).

B5.15. Итог по разделу B5

В этом разделе:

1. зафиксирован набор обязательных тестовых задач (окружность, почти окружность, гладкий контур);
2. перечислены все параметры, которые нужно сохранять для воспроизводимости;
3. введён двухуровневый формат хранения результатов (summary + step-log);
4. зафиксирован обязательный набор графиков и таблиц;
5. введены формулы для наблюдаемых порядков сходимости;
6. описан протокол диагностики округлительной ошибки;
7. задана структура каталогов и минимальные regression-tests.

Что должно быть понятно после B5. После этого раздела должно быть полностью понятно:

1. как именно запускать вычислительные эксперименты;
2. какие данные нужно сохранять;
3. какие графики и таблицы обязательно строить;
4. как отделять дискретизационную ошибку от округлительной;
5. как подготовить базу для раздела C (верификация и “машинная точность”).

B6. Практический алгоритм выбора сетки и параметров расчёта: точность, устойчивость и стоимость

В этом разделе формулируется практическая процедура выбора численных параметров для маршевого расчёта:

- числа узлов по периодической координате s (то есть N_s),
- шага по маршевой координате ξ (фиксированного или адаптивного),
- допусков адаптивного контроля,
- типа арифметики (double / long double).

Цель — получить решение с заранее заданной точностью и при этом не расходовать вычислительные ресурсы избыточно.

Главная идея. Точность решения определяется не одной ошибкой, а суммой нескольких вкладов:

$$\varepsilon_{\text{tot}} \lesssim \varepsilon_{\text{model}} + \varepsilon_s + \varepsilon_\xi + \varepsilon_{\text{round}}, \tag{137}$$

где

- $\varepsilon_{\text{model}}$ — модельная ошибка (параболизация),
- ε_s — ошибка дискретизации по s,
- ε_ξ — ошибка шага по ξ,
- $\varepsilon_{\text{round}}$ — ошибка округления.

Практический выбор параметров состоит в том, чтобы:

- сделать ε_s и ε_ξ контролируемыми,
- не уходить в режим, где доминирует $\varepsilon_{\text{round}}$,
- не тратить время на слишком мелкие сетки, если уже доминирует $\varepsilon_{\text{model}}$.

B6.1. Постановка практической цели расчёта

Перед запуском серии расчётов необходимо сформулировать *целевую точность*. Без этого невозможно осмысленно выбирать N_s и шаг по ξ.

Целевая метрика. Для задач верификации обычно выбирается одна из норм:

$$\varepsilon_{\text{target}} \in \{\varepsilon_{L_2},\ \varepsilon_{L_\infty},\ \varepsilon_{\phi,\infty}\}. \tag{138}$$

На практике рекомендуется фиксировать *две* цели:

- основную (например, ε_{L_2}),
- контрольную (например, ε_{L_∞} или фазовую ошибку).

Пример целевой постановки. Например:

$$\varepsilon_{L_2} \le 10^{-8}, \qquad \varepsilon_{L_\infty} \le 10^{-6}.$$

Такая постановка сразу задаёт численной схеме осмысленный “режим качества”.

Почему это важно. Без заранее заданного уровня точности легко:

- либо недосчитать (ошибка слишком большая),
- либо “пересчитать” (время потрачено, а точность уже ограничена модельной ошибкой или округлением).

B6.2. Разделение параметров на “пространственные” и “маршевые”

В задаче есть два независимых класса численных параметров:

- параметры дискретизации по s:

$$N_s, \qquad \Delta s = \frac{L}{N_s},$$

- параметры интегрирования по ξ:

$$h \text{ (фиксированный шаг)} \quad \text{или} \quad \{h_n\} \text{ (адаптивный шаг)}.$$

Ключевой принцип выбора. Чтобы корректно настраивать параметры, нельзя подбирать всё сразу. Нужно делать это поэтапно:

1. сначала подобрать "достаточный" N_s (чтобы ошибка по s была под контролем),
2. затем подбирать шаг по ξ,
3. затем проверить округление,
4. затем зафиксировать рабочую конфигурацию.

Почему именно так. Если одновременно менять N_s и h, невозможно понять:

- что именно дало улучшение,
- что именно ограничивает точность,
- где находится оптимум по времени.

B6.3. Этап 1: выбор числа узлов N_s (дискретизация по s)

Сначала необходимо выбрать N_s так, чтобы ошибка по s не доминировала при разумном шаге по ξ.

Практический протокол. Пусть выбран тест и фиксирован "достаточно маленький" шаг по ξ (или очень строгий adaptive-CN), чтобы вклад ε_ξ был мал. Запускается серия:

$$N_s = N_0,\ 2N_0,\ 4N_0,\ \dots \tag{139}$$

и строятся ошибки

$$\varepsilon_{L_2}(N_s), \qquad \varepsilon_{L_\infty}(N_s).$$

Критерий выбора N_s. Выбирается минимальное N_s, при котором выполняется хотя бы один из критериев:

1. **Целевой критерий:**
$$\varepsilon_s(N_s) \lesssim \theta_s\, \varepsilon_{\text{target}}, \tag{140}$$
где $\theta_s \in (0,1)$ — доля бюджета ошибки (например, $\theta_s = 0.2$ или 0.3).

2. **Критерий насыщения (plateau):** при удвоении N_s ошибка почти не меняется:
$$\frac{\varepsilon(2N_s)}{\varepsilon(N_s)} \approx 1, \tag{141}$$
то есть дальнейшее сгущение сетки по s не даёт заметного выигрыша.

Наблюдаемый порядок. Для разностной аппроксимации (центральные разности из B1) ожидается:

$$p_s^{\text{obs}} = \log_2\left(\frac{\varepsilon(N_s)}{\varepsilon(2N_s)}\right) \approx 2 \tag{142}$$

до тех пор, пока не начнут доминировать другие ошибки.

Практическое замечание. Для высокочастотных задач полезно контролировать не только абсолютную ошибку, но и "разрешение волны" вдоль s. Даже без строгой формулы разумно проверять, что увеличение k требует увеличения N_s: это должно быть видно по графикам ε vs N_s.

B6.4. Этап 2: выбор шага по ξ при фиксированном N_s

После выбора N_s настраивается шаг по ξ.

Случай фиксированного шага. При выбранном N_s запускается серия

$$h = h_0,\ \frac{h_0}{2},\ \frac{h_0}{4},\ \dots \tag{143}$$

и строится график ошибки:

$$\varepsilon(h).$$

Ожидаемое поведение для CN. Для схемы Кранка–Николсона:

$$\varepsilon_\xi(h) \approx C_\xi h^2 \tag{144}$$

(глобальная ошибка второго порядка по ξ) до тех пор, пока не начнут доминировать ε_s или $\varepsilon_{\text{round}}$.

Наблюдаемый порядок. Если шаг уменьшается в 2 раза, то:

$$p_\xi^{\text{obs}} = \log_2\left(\frac{\varepsilon(h)}{\varepsilon(h/2)}\right) \approx 2. \tag{145}$$

Критерий выбора фиксированного шага. Выбирается наибольший (то есть самый выгодный по времени) шаг h, для которого:

$$\varepsilon_\xi(h) \lesssim \theta_\xi\, \varepsilon_{\text{target}}, \tag{146}$$

где θ_ξ — доля бюджета ошибки (например, $\theta_\xi = 0.2$ или 0.3).

Почему выбирается именно *наибольший* допустимый шаг. Потому что уменьшение шага ниже необходимого:

- увеличивает число шагов,
- увеличивает время расчёта,
- может усилить накопление округлительной ошибки,
- не улучшает качество, если уже доминируют другие ошибки.

B6.5. Этап 3: переход к адаптивному шагу по ξ (рабочий режим)

После калибровки N_s и фиксированного шага полезно перейти к адаптивному режиму (B2), чтобы автоматически подстраивать h_n вдоль марша.

Основной принцип. Адаптивный шаг управляет локальной ошибкой на основе оценки

$$\Delta U^{n+1} = U^{n+1}_{\text{half}} - U^{n+1}_{\text{full}}$$

и нормированного индикатора η_n (см. B2).

Рекомендуемые настройки адаптивности. Для старта задаются:

$$\text{ATOL},\ \text{RTOL},\ h_0,\ h_{\min},\ h_{\max}, \tag{147}$$

где h_0 можно брать из калибровки фиксированного шага (этап 2).

Как выбрать ATOL **и** RTOL**.** Практически удобно привязать их к целевой точности:

$$\text{RTOL} \sim c_r\, \varepsilon_{\text{target}}, \qquad \text{ATOL} \sim c_a\, \varepsilon_{\text{target}}, \tag{148}$$

где c_r, c_a — константы порядка 0.1–1 (подбираются на пилотных запусках).

Почему нужны два допуска.

- RTOL контролирует относительную точность (когда решение не мало),
- ATOL защищает области малой амплитуды, где относительная ошибка плохо определена.

Что анализировать после адаптивного запуска. Нужно смотреть не только итоговую ошибку, но и:

- профиль $h_n(\xi_n)$,
- индикатор η_n,
- число отклонённых шагов,
- локальные невязки линейного решателя.

B6.6. Как распределять “бюджет ошибки” между s и ξ

Формула (137) показывает, что ошибка складывается из нескольких вкладов. Поэтому удобно заранее распределять допустимую ошибку по “бюджетам”.

Рекомендуемая схема распределения. Для заданной цели $\varepsilon_{\text{target}}$ можно взять:

$$\varepsilon_s \le \theta_s \varepsilon_{\text{target}}, \qquad \varepsilon_\xi \le \theta_\xi \varepsilon_{\text{target}}, \tag{149}$$

где, например,

$$\theta_s = 0.3, \qquad \theta_\xi = 0.3.$$

Оставшийся “запас” идёт на:

- модельную ошибку,
- округление,
- неточность оценивания самих ошибок.

Почему нельзя пытаться сделать ε_s и ε_ξ "максимально маленькими". Потому что:

- модельная ошибка не исчезает при сгущении сеток,
- округлительная ошибка растёт по относительной роли,
- вычислительная стоимость растёт резко.

Практически оптимум достигается, когда дискретизационные ошибки уменьшены до уровня, сравнимого с остальными вкладами, но не намного ниже.

B6.7. Диагностика режима: дискретизация или округление доминирует

Очень важная часть практической настройки — понять, что именно ограничивает точность на текущих параметрах.

Признаки доминирования дискретизационной ошибки.

- при сгущении сетки ($N_s \uparrow$) или уменьшении шага ($h \downarrow$) ошибка убывает;
- наблюдаемый порядок близок к теоретическому: $p_s^{\text{obs}} \approx 2$ и/или $p_\xi^{\text{obs}} \approx 2$.

Признаки доминирования округлительной ошибки.

- при дальнейшем уменьшении h ошибка перестаёт уменьшаться (плато),
- может начаться рост ошибки при слишком малом h,
- переход с double на long double заметно сдвигает плато вниз.

Практический тест (обязательный). Для нескольких "лучших" конфигураций выполнить сравнение:

$$\texttt{double} \quad \text{vs} \quad \texttt{long double} \tag{150}$$

при тех же N_s, h (или тех же ATOL/RTOL). Если кривые совпадают — доминирует не округление. Если расходятся — достигнут режим чувствительности к арифметике.

B6.8. Как выбирать "оптимальный" шаг: принцип пересечения ошибок

На практике полезно искать не минимально возможную ошибку, а наилучший баланс "точность/время".

Идея принципа пересечения. Пусть при уменьшении шага по ξ поведение ошибки такое:

$$\varepsilon(h) \approx C_\xi h^2 + \varepsilon_{\text{floor}}, \tag{151}$$

где $\varepsilon_{\text{floor}}$ — "пол" ошибки (из-за округления, ошибки по s и/или модельной ошибки).

Оптимальная область. Наиболее выгодный шаг находится вблизи пересечения:

$$C_\xi h^2 \sim \varepsilon_{\text{floor}}. \tag{152}$$

Именно здесь:

- дальнейшее уменьшение h почти не улучшает точность,
- но заметно увеличивает стоимость расчёта.

Как найти это practically. По log–log графику ε vs h выбирается точка, где:

- заканчивается "ровная" область второго порядка,
- начинается излом/плато.

Рабочий шаг выбирается немного *до* плато (с небольшим запасом устойчивости).

B6.9. Практический алгоритм выбора параметров (фиксированный шаг)

Ниже даётся пошаговая процедура для режима с фиксированным шагом по ξ.

Вход. Заданы:

$$\text{геометрия},\ k,\ \Xi,\ u_0(s),\ \varepsilon_{\text{target}}.$$

Шаг 1: пилотный выбор N_s.

1. Выбрать начальное N_0.
2. Выполнить серию (139) при очень малом шаге по ξ.
3. Построить $\varepsilon(N_s)$.
4. Выбрать минимальное N_s, удовлетворяющее (140) или критерию насыщения (141).

Шаг 2: пилотный выбор шага h.

1. Зафиксировать найденное N_s.
2. Выполнить серию (143).
3. Проверить наблюдаемый порядок (145).
4. Выбрать наибольший шаг h, удовлетворяющий (146).

Шаг 3: проверка округления.

1. Повторить 1–2 точки в `long double`.
2. Если разницы нет — оставить `double`.
3. Если разница есть и целевая точность важна — рассмотреть:
   - `long double`,
   - менее агрессивный шаг,
   - улучшение суммирования (например, compensated summation в эталоне).

Шаг 4: production-запуск.

1. Зафиксировать рабочую конфигурацию:

$$(N_s,\ h,\ \text{arithmetic})$$

2. Выполнить основной расчёт.
3. Сохранить run-конфигурацию и метрики в форматах B5.

B6.10. Практический алгоритм выбора параметров (адаптивный шаг)

Ниже — рекомендуемая процедура для рабочего режима с адаптивным шагом.

Шаг 1: калибровка N_s. Полностью совпадает с B6.9 (сначала выбирается N_s независимо от адаптивности).

Шаг 2: начальная оценка шага. В качестве h_0 удобно взять шаг, найденный в B6.9 для фиксированного режима, или шаг чуть крупнее (адаптивность сама скорректирует).

Шаг 3: стартовые допуски. Выбрать начальные

$$\text{ATOL},\ \text{RTOL}$$

по правилу (148).

Шаг 4: пилотный adaptive-запуск. Сохранить:

$$\varepsilon_{\text{final}},\quad n_{\text{steps}},\quad n_{\text{reject}},\quad h_n(\xi_n),\quad \eta_n.$$

Шаг 5: корректировка допусков.

- Если итоговая ошибка существенно больше цели: уменьшить RTOL (и при необходимости ATOL).
- Если итоговая ошибка гораздо меньше цели, а времени ушло много: ослабить допуски.
- Если слишком много отклонённых шагов: скорректировать h_0, фактор безопасности, или ограничения $h_{\min}, h_{\max}$.

Шаг 6: фиксация рабочего профиля допусков. После 2–3 пилотных запусков фиксируется набор:

$$(N_s,\ \text{ATOL},\ \text{RTOL},\ h_0,\ h_{\min},\ h_{\max})$$

который используется в серии экспериментов.

B6.11. Эмпирическая модель стоимости и как по ней выбирать параметры

Для практического использования полезно иметь хотя бы грубую модель стоимости расчёта.

Базовая оценка стоимости для CN. Из B3:

$$\text{стоимость одного шага} \sim C_{\text{step}}\, N_s, \tag{153}$$

а общее число шагов

$$N_\xi \sim \frac{\Xi}{h}$$

(в среднем, для фиксированного шага). Тогда:

$$T_{\text{CPU}} \sim C\, N_s\, \frac{\Xi}{h}. \tag{154}$$

Следствие.

- увеличение N_s в 2 раза примерно удваивает время;
- уменьшение h в 2 раза примерно тоже удваивает время;
- одновременное ужесточение обоих параметров быстро делает расчёт дорогим.

Практический вывод. Если целевая точность уже достигнута, дополнительное сгущение N_s или уменьшение h не имеет смысла, если не требуется специальная сверхточная верификация.

B6.12. Как выбирать параметры для разных целей: верификация vs рабочий расчёт

Один и тот же код используется в двух режимах, и параметры должны отличаться.

Режим 1: верификация (раздел C). Цель — исследовать свойства метода, а не просто получить ответ. Поэтому:

- используются серии сеток и шагов,
- иногда включается `long double`,
- допускается большой расход времени,
- сохраняются подробные step-логи.

Режим 2: рабочий (production) расчёт. Цель — получить надёжный результат за разумное время. Поэтому:

- используется заранее калиброванная конфигурация,
- шаг по ξ обычно адаптивный,
- выбирается минимально достаточный N_s,
- подробные логи можно отключить или сократить.

Почему это нужно явно разделять. Частая ошибка — использовать "верификационные" настройки в рабочих расчётах: это даёт красивую точность, но делает метод слишком медленным без практической пользы.

B6.13. Практические рекомендации по стартовым настройкам (пилотный запуск)

Ниже приводится безопасный шаблон первого запуска для новой геометрии/параметров.

Рекомендуемая последовательность.

1. Выбрать умеренное N_s (не минимальное и не экстремально большое).
2. Запустить CN с фиксированным шагом (без адаптивности) на коротком диапазоне ξ.
3. Проверить:
   - отсутствие взрывного роста,
   - невязку линейного решателя,
   - разумность амплитуды/фазы.
4. После этого включать:
   - либо серию по N_s,
   - либо адаптивность.

Почему не стоит сразу запускать "максимально точно". Потому что при ошибке в коэффициентах/индексации/геометрии:

- дорогой запуск просто дольше покажет неверный результат,
- будет сложнее понять источник проблемы.

B6.14. Признаки неправильного выбора параметров (диагностический список)

Ниже перечислены типичные симптомы и их интерпретация.

Симптом 1: ошибка не уменьшается при уменьшении шага по ξ. Возможные причины:

- доминирует ошибка по s,
- доминирует модельная ошибка,
- достигнут предел округления,
- ошибка в реализации.

Симптом 2: наблюдаемый порядок сильно меньше 2 для CN. Возможные причины:

- шаг слишком большой (неасимптотический режим),
- коэффициенты на полушаге считаются некорректно,
- ошибка в циклической индексации,
- доминирует другая ошибка (например, по s).

Симптом 3: адаптивность постоянно отклоняет шаги. Возможные причины:

- h_0 слишком большой,
- допуски слишком жёсткие,
- оценка локальной ошибки плохо масштабирована,
- участок задачи действительно "жёсткий" (нужно уменьшить $h_{\max}$).

Симптом 4: при увеличении N_s результат ухудшается. Возможные причины:

- накопление округления (особенно при слишком строгих настройках),
- ошибки индексации на больших массивах,
- проблема в линейном решателе (численная устойчивость).

Симптом 5: double и long double дают сильно разные ответы уже на грубых сетках. Это обычно признак не "тонкого" эффекта округления, а более серьёзной проблемы:

- плохая обусловленность,
- нестабильность схемы/решателя,
- ошибка в формуле коэффициентов.

B6.15. Сводный алгоритм (короткая версия, как техпроцедура)

Ниже приводится краткая техпроцедура выбора параметров, которую удобно использовать как рабочий чек-лист.

Чек-лист выбора параметров.

1. **Задать цель:** выбрать $\varepsilon_{\text{target}}$ (и норму/метрику).
2. **Подобрать N_s:** провести серию по N_s при малом шаге по ξ и выбрать минимально достаточное N_s.
3. **Подобрать шаг по ξ:** провести серию по h при фиксированном N_s и выбрать шаг вблизи области пересечения ошибок.
4. **Проверить округление:** сравнить double vs long double в нескольких точках.
5. **(Опционально) включить адаптивность:** настроить ATOL, RTOL по целевой точности и проверить логи h_n, η_n.
6. **Зафиксировать рабочую конфигурацию:** сохранить все параметры (B5) и использовать их в серии расчётов.
7. **Построить обязательные графики:** ε vs N_s, ε vs h, ε vs time, и диагностику округления.

Что даёт эта процедура. Она позволяет перейти от “подбора на глаз” к воспроизводимому режиму, в котором:

- точность предсказуема,
- стоимость расчёта контролируется,
- результаты можно защищать как научно, так и инженерно.

B6.16. Итог по разделу B6

В этом разделе:

1. введена практическая модель общей ошибки и её составляющих;
2. зафиксирован поэтапный выбор параметров ($N_s \to$ шаг по $\xi \to$ проверка округления);
3. описан принцип “пересечения” дискретизационной и округлительной ошибок;
4. дан отдельный алгоритм для fixed-step и adaptive-step режимов;
5. введены критерии диагностики неправильного выбора параметров;
6. сформулирован короткий рабочий чек-лист для воспроизводимого расчёта.

Что должно быть понятно после B6. После этого раздела должно быть полностью понятно:

1. как практически выбирать N_s и шаг по ξ;
2. как настраивать adaptive-CN под заданную точность;
3. как определять, что дальше сгущать сетку уже бессмысленно;
4. как отличать дискретизационную ошибку от округлительной;
5. как зафиксировать “production”-настройки для реальных расчётов.