

Общее для ЛР2 и ЛР3

Постановка задачи:

Цилиндр радиуса R заполнен излучающим газом, температурное поле $T(r)$ задано.

$z = \frac{r}{R}$ - безразмерная независимая величина, r - радиус слоя, для которого находим зависимые величины.

Требуется вычислить поток излучения $F(z)$ и объёмную плотность излучения $u(z)$ на промежутке $z = 0, 1$

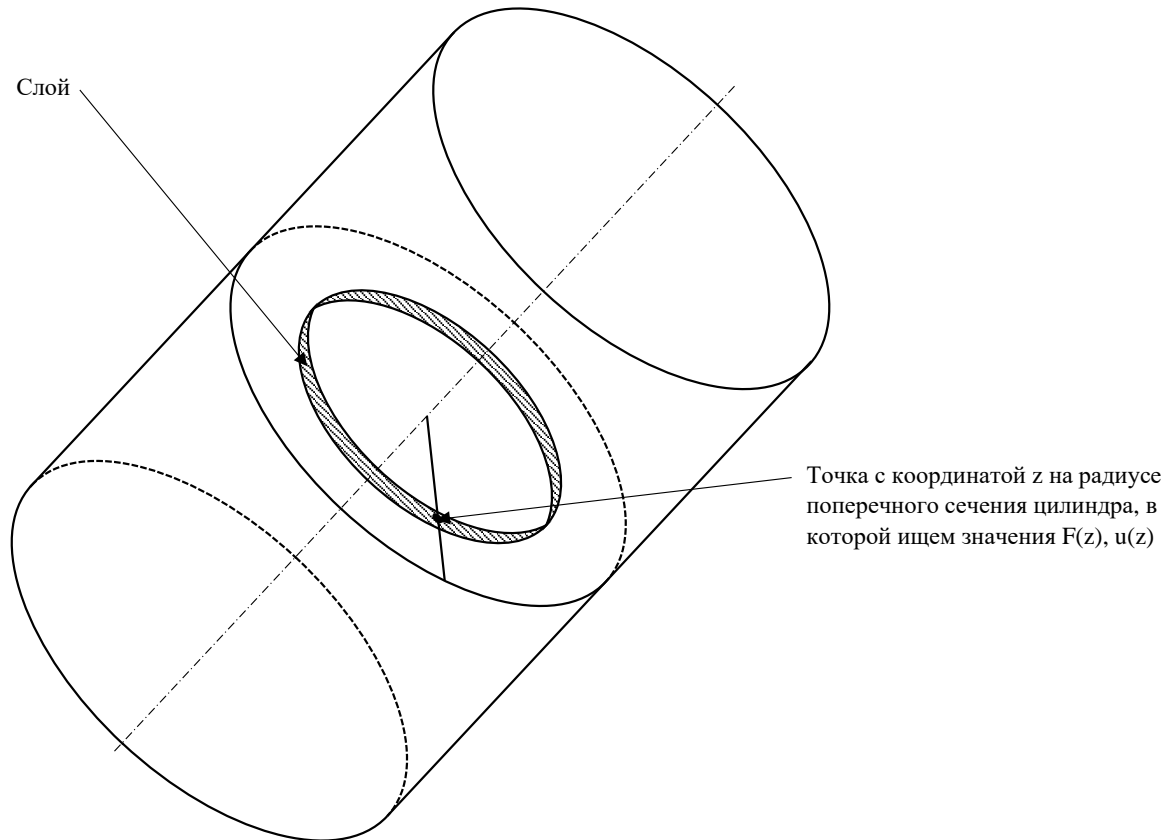


Рис. 1: Визуализация задачи

По смыслу задачи:

$F(z)$ - поток энергии через слой (от оси к стенке цилиндра);

$u(z)$ - объёмная плотность энергии излучения в слое.

Не смотря на размерности потока и объёмной плотности ($\frac{\text{Вт}}{\text{см}^2}$, $\frac{\text{Дж}}{\text{см}^3}$), задача одномерная (решается относительно z - одной независимой переменной, вдоль радиуса поперечного сечения цилиндра).

Исходная система уравнений:

$$\begin{cases} F = -\frac{c}{3Rk(T(z))} \frac{du}{dz} \\ \frac{1}{zR} \frac{d}{dz}(zF) = ck(T(z))(u_p - u) \\ T(z) = (T_w - T_0) * z^p + T_0 \\ u_p(z) = \frac{0.0003084}{e^{\frac{47990}{T(z)}} - 1} \end{cases} \quad (1)$$

Начальные условия:

$$\begin{aligned} z = 0, F(0) &= 0 \\ z = 1, F(1) &= 0.393cu(1) \\ T_w &= 2000K \\ T_0 &= 10000K \\ R &= 0.0035m \\ c &= 299792458 \frac{m}{s} \\ p &= 4 \end{aligned} \quad (2)$$

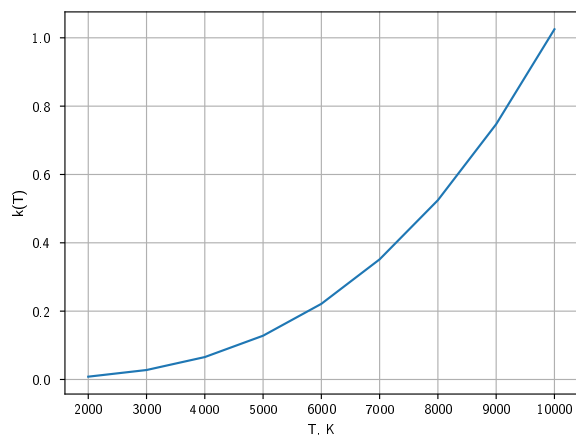
Замены:

$$\begin{aligned} a(z) &= -\frac{c}{3Rk(z)} \\ b(z) &= cRk(z) \end{aligned} \quad (3)$$

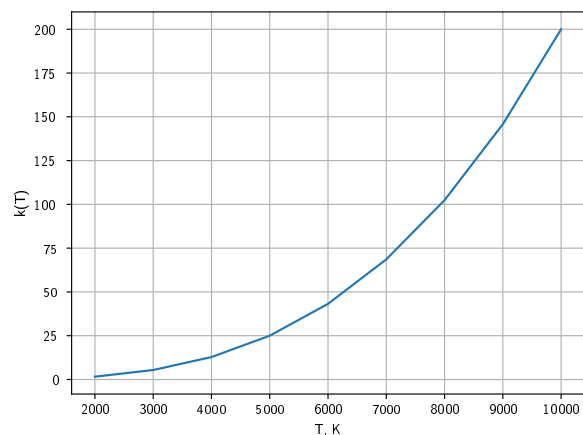
Условности нотации:

$$\begin{aligned} k(z) &\equiv k(T(z)) \\ z_n = h * n, h &- \text{шаг численного метода} \\ a(z_n) &\equiv a_n, \text{(аналогично для } b(z), k(z), u_p(z), F(z), u(z)) \end{aligned} \quad (4)$$

$k(T)$ выглядит следующим образом:



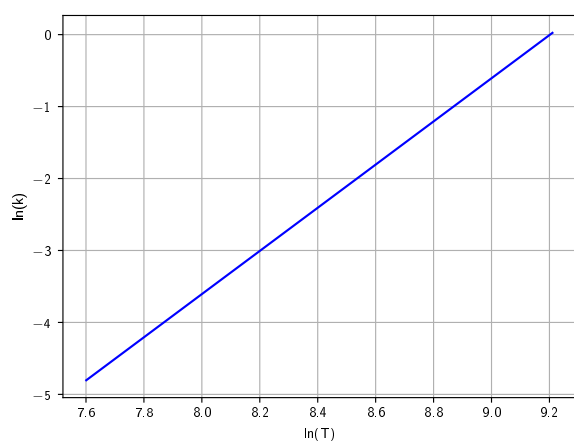
(a) $k(T)$ - Вариант 1



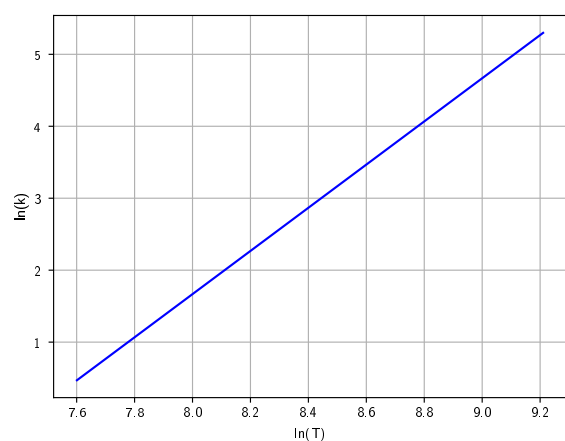
(b) $k(T)$ - Вариант 2

Если интерполировать как на графике (полиномом 1 степени), то получим некоторую ошибку.

Применяя предложенный в методе метод ($t_ln = \ln(T)$, $k_ln = \ln(k)$), получаем:



(a) $k_ln(t_ln)$ - Вариант 1



(b) $k_ln(t_ln)$ - Вариант 2

Ошибка ожидается около-нулевой, $k(T)$ вычисляется следующим образом:

$$k(T) = e^{c_1 \cdot \ln(T) + c_0}$$

* c_1 и c_0 можно посчитать руками, решив систему уравнений 1 степени для крайних значений $k(2000)$, $k(10000)$. Значения полученные с помощью numpy.polyfit: $c_1 = 2.99996105$, $c_0 = -27.60599153$ - Вариант 1, $c_1 = 3.0$, $c_0 = -22.33270375$ - Вариант 2.

ЛР2

Дифференцируем $zF : \frac{d}{dz}(zF) = F + z * \frac{dF}{dz}$

Получаем:

$$\begin{cases} \frac{du}{dz} = \frac{F}{a(z)} \\ \frac{dF}{dz} = b(z)(u_p - u) - \frac{F}{z} \end{cases} \quad (5)$$

Данная задача решается как задача Коши, если задать начальное условие для u :

$$u(0) = \mathcal{E}u_p(0)$$

Ожидая, что решение может быть найдено при начальном $\mathcal{E} \in \{0..1\}$ и что $u(1)$ монотонно зависит от \mathcal{E} , находим решение методом стрельбы + дихотомии:

```
Epsilon1 = 0.0
Epsilon2 = 1.0
Err = inf

цикл пока fabs(Err) > заданная точность

    Epsilon = (Epsilon1 + Epsilon2) / 2
    Err = Рунге-Кутты(Epsilon)

    если Err > 0.0
        Epsilon1 = Epsilon
    иначе
        Epsilon2 = Epsilon
    конец если

конец цикла
```

При этом $Err = \frac{F^{(N)}(1) - 0.393cu^{(N)}(1)}{0.393cu^{(N)}(1)} = \frac{F^{(N)}(1)}{0.393cu^{(N)}(1)} - 1$

Вычисления завершаются на N-ой итерации, когда для очередного приближения $F^{(N)}(z)$, $u^{(N)}(z)$ выполняется $|Err| \leq \delta$, где δ - заданная точность.

Для решения задачи Коши используем формулы Рунге-Кутты IV порядка точности для двух зависимых переменных:

```

F = 0
u = Epsilon * u_p(0)
z = 0

пока z <= 1

    P_1 = F / a(z)
    K_1 = b(z) * (u_p(z) - u) - F / z

    z += h / 2

    P_2 = (F + K_1 * h / 2) / a(z)
    K_2 = b(z) * (u_p(z) - u - P_1 * h / 2) - (F + K_1 * h / 2) / z

    P_3 = (F + K_2 * h / 2) / a(z)
    K_3 = b(z) * (u_p(z) - u - P_2 * h / 2) - (F + K_2 * h / 2) / z

    z += h / 2

    P_4 = (F + K_3 * h) / a(z)
    K_4 = b(z) * (u_p(z) - u - P_3 * h) - (F + K_3 * h) / z

    F += (K_1 + 2 * (K_2 + K_3) + K_4) * h / 6
    u += (P_1 + 2 * (P_2 + P_3) + P_4) * h / 6

конец пока

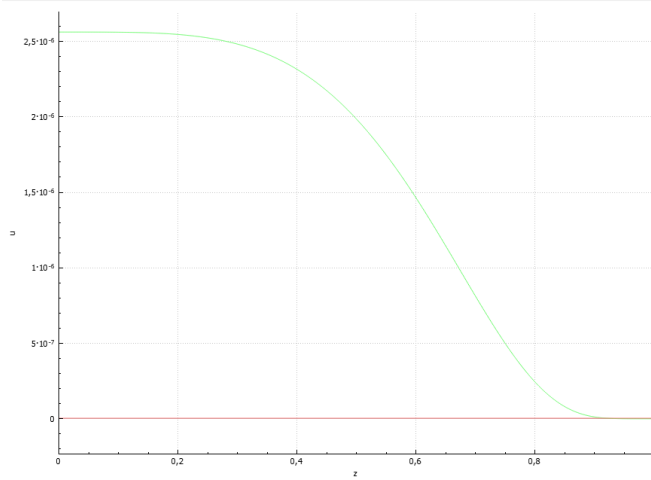
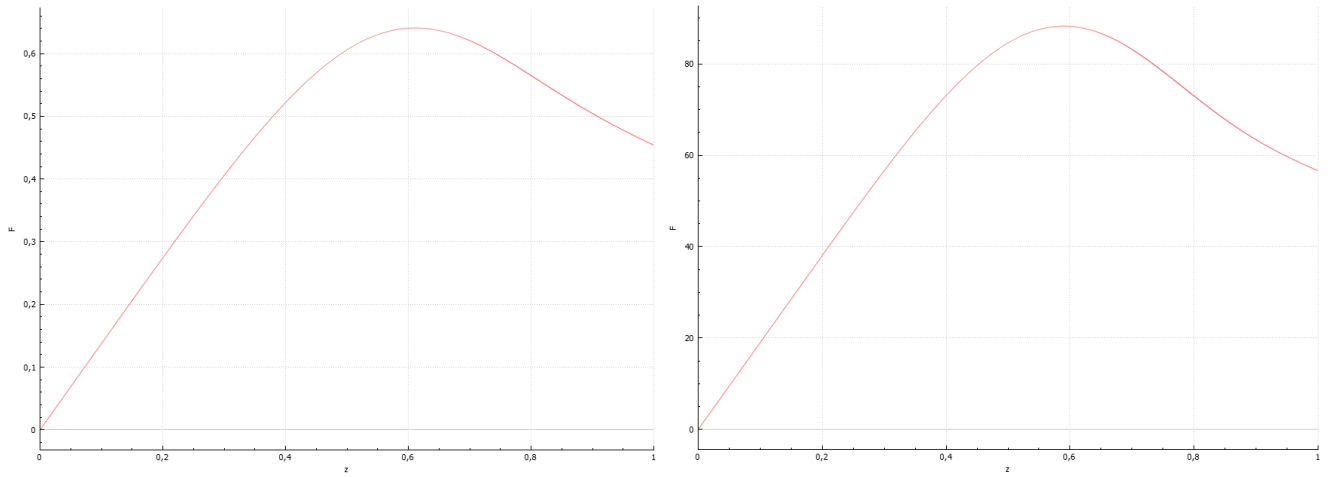
```

При этом на 1-й итерации при вычислении K_1 произойдёт деление на 0, поэтому целесообразно вынести эту итерацию из цикла и проинициализировать $K_1 := b(z) * (u_p(z) - u)$.

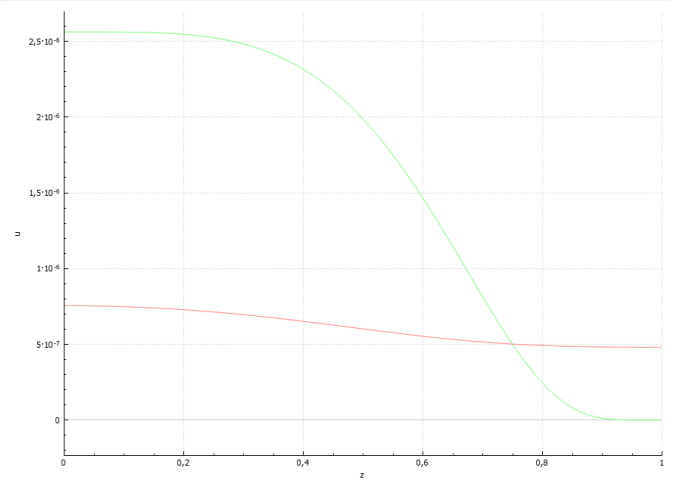
*Возможно оптимизировать путем выделения общих для K_2 и K_3 , P_2 и P_3 частей, а также общей для $a(z)$, $b(z)$ части - $Rk(z)$.

Решения

Решения с параметрами $h = 10^{-7}$, $\delta = 0.001$ (для Err) (64bit c++17):



(a) Вариант 1



(b) Вариант 2

*Зелёная кривая - функция Планка ($u_p(z)$).

Epsilon (коэфф. $u_p(0)$ для начального условия):

Вариант 1: $Epsilon = 0.00150871$

Вариант 2: $Epsilon = 0.2948$

Вопросы при защите лабораторной работы

1. Какие способы тестирования программы можете предложить?

Существует вразумительный ответ, но я его забыл)))

Точно не стоит предлагать сравнивать результаты вычислений с реальными измерениями, т.к. мы моделируем специально для того, чтобы не экспериментировать ирл.

Вероятнее всего, речь про тривиальные случаи. Потвкать параметры задачи так, чтобы решение было очевидным, и сравнить с результатом работы программы. Например: задать правое краевое условие $F(1) = 100 \frac{\text{Вт}}{\text{м}^2}$ и проверить, выполняется ли оно в результате работы программы.

2. Приведите классификацию методов решения систем ОДУ для задачи Коши

Аналитические. Можем решить - находим общее решение, используя начальное условие получаем частное решение. (такого почти не бывает)

Приближенно аналитические. Пикар. Если удаётся интегрировать правую часть много раз, получаем сходящийся к точному решению ряд. Сходится плохо. (тоже почти не бывает, но есть исключительные задачи по типу $u'(x) = x^2 + u^2$, для которых решение чисто аналитически не найти).

Численные методы:

- одношаговые (для вычисления значений в следующем узле требуется знать значения только в предыдущем узле) / многошаговые (для вычисления значений в следующем узле требуется знать значения в нескольких предыдущих узлах).
- явные ($u_n = \psi(x_{n-1}, u_{n-1}, f(x_{n-1}, u_{n-1}))$) / неявные ($u_n = \psi(x_{n-1}, u_{n-1}, f(x_n, u_n))$). В случае явных следующее значение выражается через предыдущее. В случае неявных следующее значение выражается через предыдущее И производную следующего - нельзя применить 'как есть', нужно искать решение уравнения - не всегда возможно. Как правило, неявные методы более устойчивые. Зачастую (всегда, в случае монотонных функций) явный и неявный метод сходятся к решению с разных сторон, поэтому можно использовать совместно для отыскания интервала, в котором находится точное решение.

3. Получите систему разностных уравнений для решения сформулированной задачи неявным методом Эйлера. Опишите алгоритм реализации полученных уравнений.

Неявный метод Эйлера для двух переменных:

$$\begin{cases} f_n = f_{n-1} + hf'(x_n, f_n, t_n) \\ t_n = t_{n-1} + ht'(x_n, t_n, f_n) \end{cases} \quad (6)$$

Для нашей задачи получаем:

$$\begin{cases} F_n = F_{n-1} + h(b(z_n)(u_{p_n} - u_n) - \frac{F_n}{z_n}) \\ u_n = u_{n-1} + h\frac{F_n}{a(z_n)} \end{cases} \quad (7)$$

Подставляем (6.2) в (6.1):

$$F_n = F_{n-1} + h(b(z_n)(u_{p_n} - u_{n-1} - \frac{F_n}{a(z_n)}h) - \frac{F_n}{z_n}) \quad (8)$$

Выражаем F_n из (7):

$$F_n = \frac{F_{n-1} + hb_n(u_{p_n} - u_{n-1})}{1 + \frac{h}{z_n} + \frac{h^2 b_n}{a_n}} \quad (9)$$

Система разностных уравнений:

$$\begin{cases} F_n = \frac{F_{n-1} + hb_n(u_{p_n} - u_{n-1})}{1 + \frac{h}{z_n} + \frac{h^2 b_n}{a_n}} \\ u_n = u_{n-1} + h \frac{F_n}{a(z_n)} \end{cases} \quad (10)$$

Алгоритм реализации: 1. Инициализировать F_0, u_0 2. Пока $z \leq 1$ вычисляем F_n, u_n

4. Получите систему разностных уравнений для решения сформулированной задачи неяв-ным методом трапеций. Опишите алгоритм реализации полученных уравнений.

Неявный метод трапеций для двух переменных:

$$\begin{cases} f_n = f_{n-1} + h \frac{f'(x_{n-1}, f_{n-1}, t_{n-1}) + f'(x_n, f_n, t_n)}{2} \\ t_n = t_{n-1} + h \frac{t'(x_{n-1}, t_{n-1}, f_{n-1}) + t'(x_n, t_n, f_n)}{2} \end{cases} \quad (11)$$

Для нашей задачи получаем:

$$\begin{cases} F_n = F_{n-1} + h \frac{b(z_{n-1})(u_{p_{n-1}} - u_{n-1}) - \frac{F_{n-1}}{z_{n-1}} + b(z_n)(u_{p_n} - u_n) - \frac{F_n}{z_n}}{2} \\ u_n = u_{n-1} + h \frac{\frac{F_{n-1}}{a(z_{n-1})} + \frac{F_n}{a(z_n)}}{2} \end{cases} \quad (12)$$

Подставляем (12.2) в (12.1) и выражаем F_n , получаем систему разностных уравнений:

$$\begin{cases} F_n = \frac{F_{n-1}(1 - \frac{h}{2z_{n-1}} - \frac{h^2 b_n}{4a_{n-1}}) + \frac{h}{2}(b_{n-1}u_{p_{n-1}} + b_n u_{p_n} - u_{n-1}(b_{n-1} + b_n))}{1 + \frac{h}{2z_n} + \frac{h^2 b_n}{4a_n}} \\ u_n = u_{n-1} + h \frac{\frac{F_{n-1}}{a(z_{n-1})} + \frac{F_n}{a(z_n)}}{2} \end{cases} \quad (13)$$

Алгоритм реализации: $-//-$, поскольку в (12.1) есть деление на z_{n-1} , надо на 1-й итерации не поделить на 0 (вынести за цикл)

5. Из каких соображений проводится выбор численного метода того или иного порядка точности, учитывая, что чем выше порядок точности метода, тем он более сложен и требует, как правило, больших ресурсов вычислительной системы?

Для некоторых задач не является целесообразным использовать методы более высокого порядка точности.

Если аналитически или по найденным решениям можно сделать вывод о том, что производная P -го порядка $u^{(P)}(z)$ или даже младших порядков - не существует, то метод порядка точности P - избыточен. Он будет давать точность меньшую, чем оправдана объёмом вычислений.

От метода порядка точности P мы ожидаем, что погрешность решения убывает в k^P раз при уменьшении шага в k раз. Если этого не наблюдается на больших шагах, то метод избыточен и его использовать не стоит.