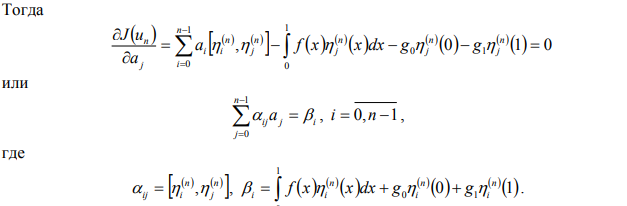
**Теория**

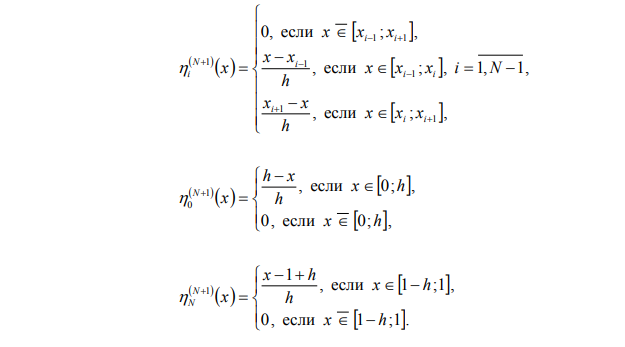
Еще одним подходом к решению данной задачи является вариационно-разностный метод. Так, решение исходной задачи эквивалентно отысканию функции , доставляющей минимум функционала

где

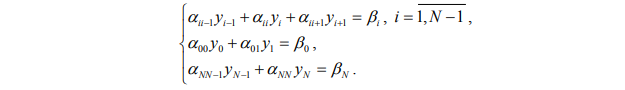
В соответствии с идеей Ритца построим последовательность конечномерных подпространств и будем искать минимум на . Данный минимум существует, так как выполняются условия на отрезке [0; 1]. Пусть размерность подпространства равна n и – базис этого подпространства, т.е. любой элемент этого подпространства представим в виде

Подставляя записанное представление в функционал, получим функцию n переменных Так как мы желаем получить минимум этой функции, то должны удовлетворять системе уравнений





При таком подборе координатных функций мы придем к следующей системе:



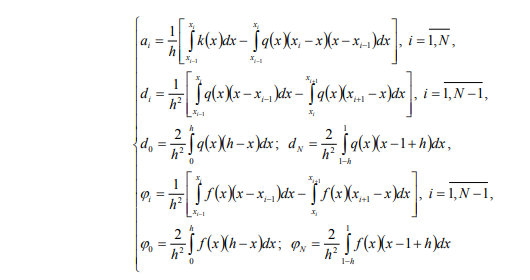
Полученную систему уравнений можно записать в стандартной для однородных консервативных схем виде:



Необходимо положить, что



При



и заменяя интегралы квадратурной формулой трапеций, приходим к следующей системе:

Опять-таки нетрудно заметить, что матрица имеет трехдиагональную структуру, следовательно, решаем систему методом разностной прогонки.

**Листинг**

**def form\_matrix\_3(N):**

**h = 1 / N # шаг сетки**

**x = [i \* h for i in range(N + 1)] # равномерная сетка**

**A = [.0] \* (N + 1) # коэффиценты при y\_i-1**

**B = [.0] \* (N + 1) # коэффиценты при y\_i**

**C = [.0] \* (N + 1) # коэффиценты при y\_i+1**

**F = [.0] \* (N + 1) # правый столбец системы**

**B[0] = ( k(x[0]) + k(x[1]) ) / (2 \* h) + 0.5 \* h \* q(x[0]) + KAPPA\_0 # нашли B[0]**

**C[0] = - ( ( k(x[0]) + k(x[1]) ) / (2 \* h) ) # нашли C[0]**

**F[0] = 0.5 \* h \* f(x[0]) + G\_0 # нашли F[0]**

**for i in range(1, N): # в цикле вычисляем A\_i, B\_i ,C\_i, F\_i**

**A[i] = ( k(x[i - 1]) + k(x[i]) ) / (2 \* h\*\*2)**

**B[i] = - ( ( k(x[i - 1]) + 2 \* k(x[i]) + k(x[i + 1]) + 2 \* q(x[i]) \* h\*\*2 ) / (2 \* h\*\*2) )**

**C[i] = ( k(x[i]) + k(x[i + 1]) ) / (2 \* h\*\*2)**

**F[i] = - f(x[i])**

**A[N] = - ( ( k(x[N - 1]) + k(x[N]) ) / (2 \* h) ) # нашли A[N]**

**B[N] = ( k(x[N - 1]) + k(x[N]) ) / (2 \* h) + 0.5 \* h \* q(x[N]) + KAPPA\_1 # нашли B[N]**

**F[N] = 0.5 \* h \* f(x[N]) + G\_1 # нашли F[N]**

**return A, B, C, F**