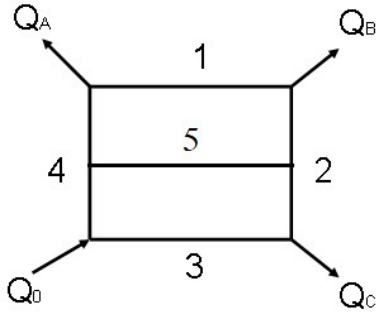
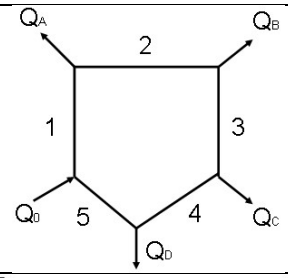
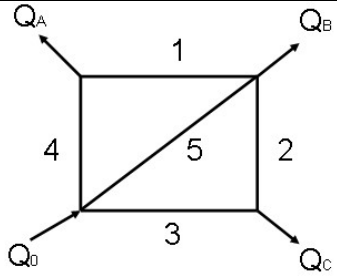
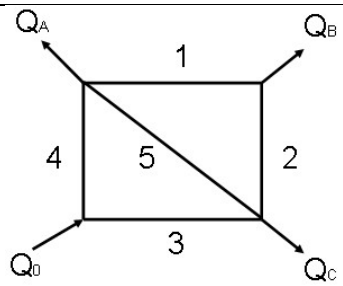
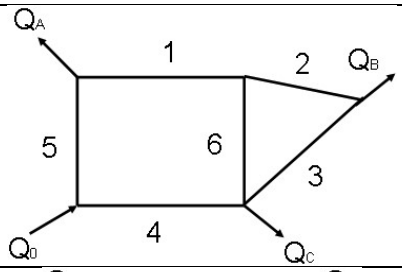
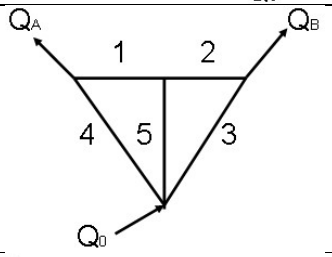
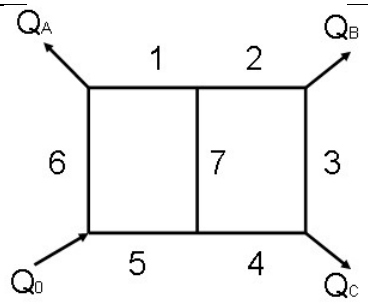
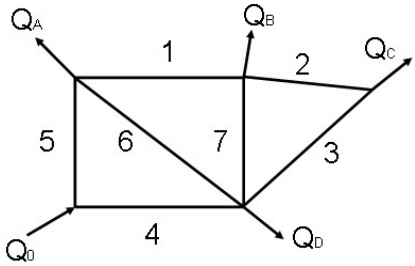
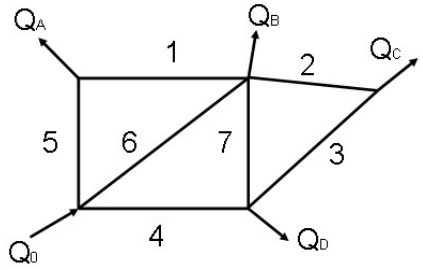
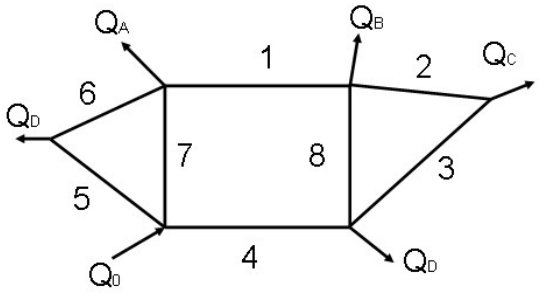
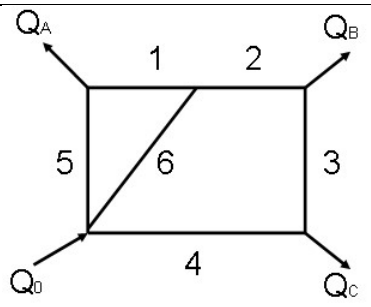
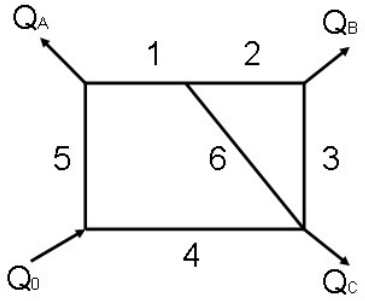
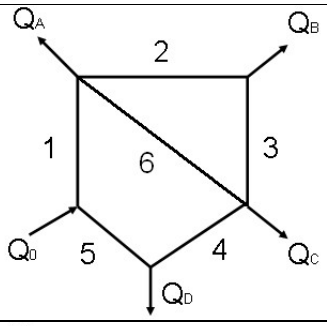
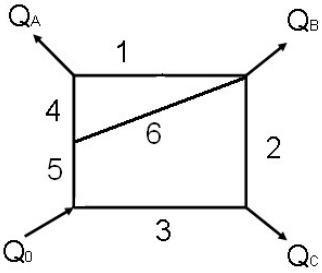
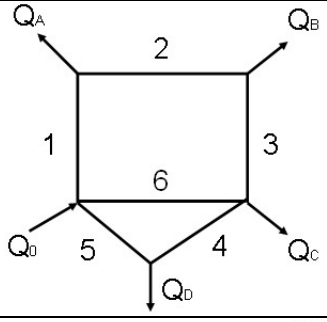
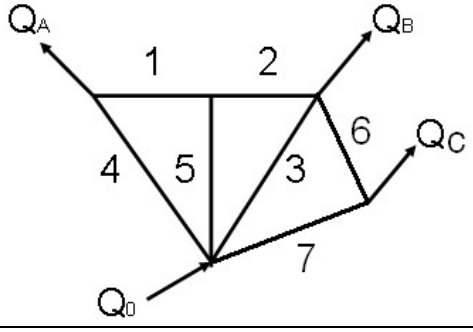
Лабораторная работа №2

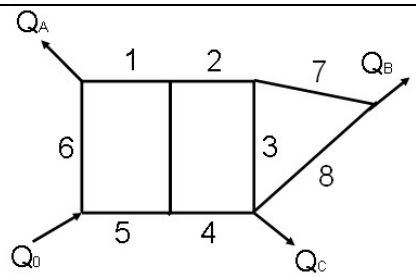
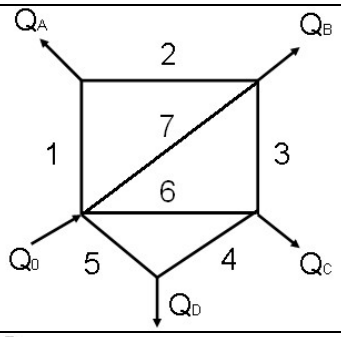
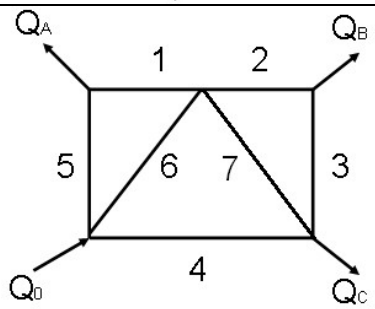
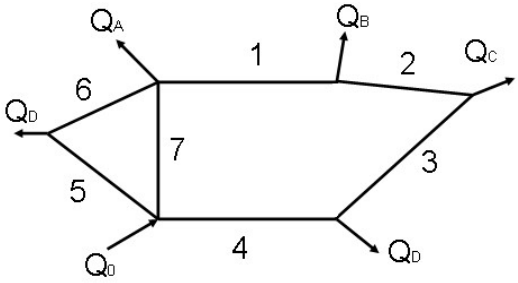
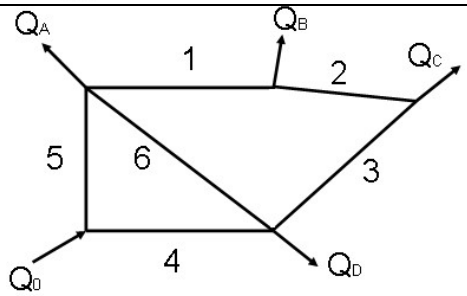
Численное решение нелинейных систем

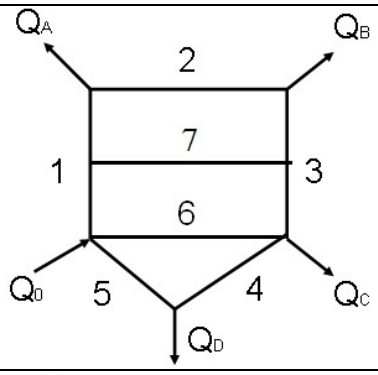
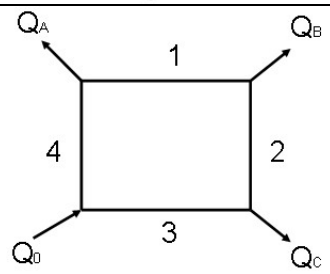
Задание. Найти объемные расходы Q_i жидкости на участках сложного трубопровода. Диаметры d и абсолютные шероховатости Δ всех труб одинаковые. Транспортируемая жидкость имеет кинематическую вязкость $\nu = 10^{-6}$ м^2/с. Варианты заданий приведены в таблице.

№ вар.	Схема трубопровода	$L_1, L_2,$ … м	d, мм Δ, мм	$\zeta_{м1}, \zeta_{м2},$ …	$Q_A, Q_B,$ … м^3/с
1		110 70 110 70 110	280 0.80	1.5 1.1 0.3 0.3	0.1 0.5 0.4
2		50 70 50 40 40	350 0.7	0.5 1 1.5 0 1	0.3 0.2 0.3 0.1
3		60 80 60 80	250 0.80	0.5 1.2 1.8 0 1	0.2 0.1 0.3
4		70 50 70 50	250 0.9	0.75 1 0 0.6 0.8	0.2 0.4 0.2

5		80 50 60 80 100 100	450 1	1 1 0 1.2 0.7	0.3 0.3 0.6
6		90 90 140 140	400 1.2	0.5 0 0.5 0.9 1.2	0.25 0.4 0.35
7		100 130 80 100 130 80 80	300 1.1	0.5 1 0 1 0.5	0.2 0.3 0.1
8		110 50 70 110 70 90 70	300 1.3	0 0.5 0.5 1 0.5 0 0	0.4 0.2 0.3
9		120 60 60 120 90 160 90	250 0.5	1 0 1.5 0 1 0 1	0.1 0.2 0.2
10		130 150 170 130 150 170 100 100	200 1	0 0 0.5 1 1.5 0 0 1	0.2 0.3 0.1 0.25

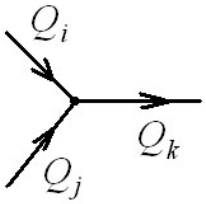
11		60 60 70 120 70 -	300 0.8	1.2 0.5 0.8 0.5 1.1 0.3	0.1 0.2 0.2
12		100 100 150 200 150 -	200 1.2	1 0.8 0.6 1.5 0.8 0.6	0.3 0.2 0.2
13		60 80 60 50 50	250 1.2	0.5 1 1.5 0 1 0.4	0.1 0.5 0.2 0.3
14		100 80 100 40 40	220 0.80	2.4 1.2 0.1 0.2 0.9 0.3	0.3 0.4 0.3
15		60 80 60 50 50	180 1.6	0.6 1.1 1.3 0.2 1.3 0.7	0.2 0.2 0.35 0.45
16		90 90 140 140 - 100 120	350 1.6	0.5 0 0.5 0.9 1.2	0.35 0.3 0.25 0.7

17		40 40 60 40 40 60 45 55	200 0.8	0.1 0.5 0.5 1.0 0.6 0 0.25 0.45	0.4 0.2 0.7
18		60 80 60 50 50 80 -	160 1.2	0.3 1.5 1.2 0. 1.0 0.3 0.5	0.3 0.25 0.15 0.50
19		60 60 70 120 70 - -	240 1.6	1.0 0.0 0.8 0.6 1.1 0.3 0.7	0.3 0.1 0.1
20		130 150 170 130 150 170 100	200 1	0.3 0.2 0.0 1.3 1.0 0 0.6	0.3 0.2 0.1 0.2
21		110 50 70 110 80 -	180 1.1	0.2 0.0 0.6 1.2 0.3 0 0.1	0.3 0.4 0.1 0.15

22		50 70 50 40 40 70 70	150 1.0	0.4 1.4 1.3 0.1 1.2 0.5 0.7	0.45 0.15 0.35 0.40
23		100 80 100 80	300 0.60	2 1 0 0.5	0.3 0.4 0.3

Указания. Расходы Q_i (м^3/с) на участках трубопровода должны удовлетворять системе уравнений:

1. Баланс расходов в узлах:

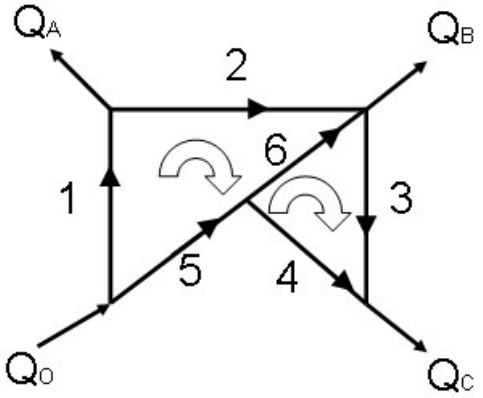


$$Q_i + Q_j = Q_k$$

Формулируем $N-1$ таких уравнений, где N – общее количество узлов.

2. Баланс напоров, т.е. равенство нулю алгебраической суммы потерь напора для каждого кольца при подсчете по направлению движения часовой стрелки или против нее. Потери напора считаются положительными при совпадении направлений обхода контура и течения жидкости.

Например, для трубопровода с $N=5$ узлами и 2 кольцами



система уравнений имеет вид

$$Q_1 - Q_2 = Q_A, \quad Q_2 + Q_6 - Q_3 = Q_B, \quad Q_4 + Q_3 = Q_C, \quad Q_5 - Q_4 - Q_6 = 0,$$

$$\Delta H_1 + \Delta H_2 - \Delta H_6 - \Delta H_5 = 0\,, \quad \Delta H_3 - \Delta H_4 + \Delta H_6 = 0\,. \tag{1}$$

Здесь $\Delta H_i = K_i Q_i |Q_i|$ – потери напора (м), $K_i = \beta_i \zeta_i$, $\beta_i = 8/(\pi^2 g d_i^4)$, $\zeta_i = \zeta_{\text{м}i} + \lambda(Q_i, d_i, \Delta_i)\dfrac{l_i}{d_i}$ – коэффициент сопротивления i-го участка; $\lambda(Q,d,\Delta) = 0.11\big(\Delta/d + 68/(cQ)\big)^{0.25}$ – коэффициент гидравлического трения, $c = 4/(\pi \nu d)$, ($cQ \equiv \mathrm{Re}$ – число Рейнольдса); $\zeta_{\text{м}i}$ – коэффициент местных сопротивлений. С учетом этого (1) принимает вид

$$\begin{bmatrix} Q_1 - Q_2 \\ Q_2 - Q_3 + Q_6 \\ Q_3 + Q_4 \\ -Q_4 + Q_5 - Q_6 \\ K_1 Q_1|Q_1| + K_2 Q_2|Q_2| - K_5 Q_5|Q_5| - K_6 Q_6|Q_6| \\ K_3 Q_3|Q_3| - K_4 Q_4|Q_4| + K_6 Q_6|Q_6| \end{bmatrix} = \begin{bmatrix} Q_A \\ Q_B \\ Q_C \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

или в матрично-векторной форме

$$A(\vec{Q}) \cdot \vec{Q} = \vec{b}\,, \tag{2}$$

где

$$A(\vec{Q}) = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & -1 \\ K_1|Q_1| & K_2|Q_2| & 0 & 0 & -K_5|Q_5| & -K_6|Q_6| \\ 0 & 0 & K_3|Q_3| & -K_4|Q_4| & 0 & K_6|Q_6| \end{bmatrix},$$

$$\vec{Q} = \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \end{bmatrix}, \quad \vec{b} = \begin{bmatrix} Q_A \\ Q_B \\ Q_C \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Для решения системы нелинейных уравнений (2) применяем метод последовательных приближений с инерцией по следующему алгоритму:

1) Задаем весовой коэффициент α ($0<\alpha<1$, рекомендуемое значение 0.7…0.8), точность ε и максимальное число итераций N_{it}.

2) Задаем расходы на участках на нулевой и первой итерациях:

$$\vec{Q}_i^{(0)} = Q_0 / N, \quad \vec{Q}_i^{(1)} = \vec{Q}_i^{(0)}, \; i = 1,...,N,$$

где $Q_0 = Q_A + Q_B + \cdots$ – расход в магистрали.

3) Для $k = 1,2,...,N_{it}$

решаем СЛАУ

$$A((1-\alpha)\vec{Q}^{(k-1)} + \alpha\vec{Q}^{(k)}) \cdot \vec{Q}^{(k+1)} = \vec{b}. \qquad (3)$$

если $\max\limits_{1\le i\le N}\left|\vec{Q}_i^{(k+1)} - \vec{Q}_i^{(k)}\right| < \varepsilon$, то приближенное решение найдено: $\vec{Q}_i \approx \vec{Q}_i^{(k)}$, STOP.

Ниже приведена программная реализация алгоритма. Для решения СЛАУ (3) используется метод Гаусса (процедура «Gauss1» модуля «matrlib»). Задание элементов матрицы $A(\vec{Q})$ системы осуществляется в процедуре «Matr», определение коэффициентов сопротивления и гидравлического трения – в процедурах «dzefun» и «alamfun» соответственно. Для проверки используется функция «root» модуля «scipy.optimize». В качестве примера рассчитывается изображенный выше трубопровод при $Q_A = 0.3$ м^3/с, $Q_B = 0.3$ м^3/с, $Q_C = 0.6$ м^3/с, $d_1 = 120$ мм, $d_2 = 100$ мм, $d_3 = 150$ мм, $d_4 = 100$ мм, $d_5 = 150$ мм, $d_6 = 200$ мм, $\Delta = 1$ мм, $l_1 = 100$ м, $l_2 = 200$ м, $l_1 = 100$ м, $l_1 = 120$ м, $l_1 = 120$ м, $l_1 = 100$ м, $\zeta_{м1} = 10$, $\zeta_{м2} = 3$, $\zeta_{м3} = 1$, $\zeta_{м4} = 4$, $\zeta_{м5} = 1$, $\zeta_{м6} = 5$.

```
from math import pi
from copy import deepcopy
from matrlib import MatrPrint,Gauss1,Resid,NRazn
import numpy as np
from scipy import optimize

def alamfun(delt,q,c):
    Re=c*q
    return 0.11*(delt+68./Re)**0.25

def dzefun(dzem,delt,al,q,c):
    return dzem+alamfun(delt,q,c)*al

def Matr(N,dzem,delt,al,c,bet,q,A):
    aK = [0]*(N+1)
    for i in range(1,N+1):
        aK[i] = bet[i]*dzefun(dzem[i],delt[i],al[i],q[i],c[i])
    A[1][1:] = [1.,-1.,0.,0.,0.,0.]
    A[2][1:] = [0.,1.,-1.,0.,0.,1.]
    A[3][1:] = [0.,0.,1.,1.,0.,0.]
    A[4][1:] = [0.,0.,0.,-1.,1.,-1.]
```

```
    A[5][1:] = [aK[1]*abs(q[1]),  aK[2]*abs(q[2]), 0., 0.,
              -aK[5]*abs(q[5]), -aK[6]*abs(q[6])]
    A[6][1:] = [0., 0., aK[3]*abs(q[3]), -aK[4]*abs(q[4]),
              0., aK[6]*abs(q[6]) ]

N=6
d,bet,c,dzem,delt,al,q,b,qq = ([0]*(N+1) for j in range(9))
A=[[0]*(N+1) for j in range(N+1)]
anu=1.e-6                          # кин. вязкость
d[1:]=[120.,100.,150.,100.,150.,200.]
d[1:]=[d[i]/1e3 for i in range(1,N+1)]
bet[1:]=[8/(9.81*pi**2*d[i]**4) for i in range(1,N+1)]
bet[1:]=[bet[i]/bet[1] for i in range(1,N+1)]
c[1:]=[4./(pi*anu*d[i]) for i in range(1,N+1)]   # c*q=Re
dzem[1:]=[10.,3.,1.,4.,1.,5.]            # у-ты местных сопротивлений
delt[1:]=[1.0e-3/d[i] for i in range(1,N+1)]   # относит. шероховатость
al[1:]=[100.,200.,100.,120.,120.,100.]
al[1:]=[al[i]/d[i] for i in range(1,N+1)]  # относит. длины
qa,qb,qc = 0.3,0.3,0.6
q0=qa+qb+qc                        # расход в магистрали
b[1:]=[qa,qb,qc,0.,0.,0.]              # вектор правых частей
q[1:]=[q0/N for i in range(1,N+1)]
qold=q.copy()
print(' q0(m^3/s)=', ' '.join(f'{qq:.3f}' for qq in q[1:]))
Matr(N,dzem,delt,al,c,bet,q,A)
print('A=')
MatrPrint(A, N, N)
Rmax=Resid(A,b,q,N)
print('Rmax=',Rmax)
eps, alfa = 1.0e-6, 0.8
it, itmax = 0, 100
while it<itmax:
    qold2=qold.copy()
    qold=q.copy()
    qq[1:]=[(1-alfa)*qold2[i]+alfa*qold[i] for i in range(1,N+1)]
    Matr(N,dzem,delt,al,c,bet,qq,A)
    b1 = deepcopy(b)
    q,det = Gauss1(A,b1,N) # решаем СЛАУ
    it=it+1
    dq = NRazn(q, qold, N)
    print(' Iter. #',it,f' dqmax={dq:.3e}')
    if dq<eps: print('dq<eps'); break
    if it==itmax: print('it=itmax'); break
print(' q(m^3/s)=', ' '.join(f'{qq:.3f}' for qq in q[1:]))
Matr(N,dzem,delt,al,c,bet,q,A)
Rmax=Resid(A,b,q,N)
print(f'Rmax={Rmax:.3e}')

# Scipy
def residual(q):
    q1=q.tolist()
    A1=[[0]*(N+1) for j in range(N+1)]
    Matr(N,dzem,delt,al,c,bet,q1,A1)
    A=np.array(A1)
    return np.dot(A,q)-b
guess = (q0/N)*np.ones(N+1)
sol = optimize.root(residual, guess, method='hybr')
```

```
print('Scipy q=', ' '.join(f'{xx:.3f}' for xx in sol.x[1:]))
print(f'Residual={abs(residual(sol.x)).max():.3e}')
```

Результаты:

```
q0(m^3/s)= 0.200 0.200 0.200 0.200 0.200 0.200
A=
 1.00000e+00-1.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00 0.00000e+00
 0.00000e+00 1.00000e+00-1.00000e+00 0.00000e+00 0.00000e+00 1.00000e+00
 0.00000e+00 0.00000e+00 1.00000e+00 1.00000e+00 0.00000e+00 0.00000e+00
 0.00000e+00 0.00000e+00 0.00000e+00-1.00000e+00 1.00000e+00-1.00000e+00
 7.54451e+00 3.01155e+01 0.00000e+00 0.00000e+00-2.14492e+00-5.09696e-01
 0.00000e+00 0.00000e+00 1.80109e+00-1.89817e+01 0.00000e+00 5.09696e-01
Rmax= 7.001080409896835
 Iter. # 1  dqmax=6.978e-01
 Iter. # 2  dqmax=3.687e-01
 Iter. # 3  dqmax=2.536e-01
 Iter. # 4  dqmax=1.164e-01
 Iter. # 5  dqmax=4.411e-02
 Iter. # 6  dqmax=1.181e-02
 Iter. # 7  dqmax=6.272e-04
 Iter. # 8  dqmax=1.858e-03
 Iter. # 9  dqmax=1.610e-03
 Iter. # 10  dqmax=9.160e-04
 Iter. # 11  dqmax=4.105e-04
 Iter. # 12  dqmax=1.451e-04
 Iter. # 13  dqmax=3.395e-05
 Iter. # 14  dqmax=1.855e-06
 Iter. # 15  dqmax=8.269e-06
 Iter. # 16  dqmax=6.239e-06
 Iter. # 17  dqmax=3.335e-06
 Iter. # 18  dqmax=1.420e-06
 Iter. # 19  dqmax=4.682e-07
dq<eps
 q(m^3/s)= 0.404 0.104 0.432 0.168 0.796 0.628
Rmax=4.106e-06
Scipy q= 0.404 0.104 0.432 0.168 0.796 0.628
Residual=9.501e-13
```

Как видим, для достижения заданной точности $\varepsilon = 10^{-6}$ потребовалось 19 итераций, расходы ($м^3/с$) жидкости на участках равны $Q_1 = 0.404$, $Q_2 = 0.104$, $Q_3 = 0.432$, $Q_4 = 0.168$, $Q_5 = 0.796$ и $Q_6 = 0.628$.