PME3332 - Mecânica dos Fluidos : Noções, Laboratório e Aplicações

T1 - Instalação de um sistema composto por condutos

Prof. Dr. Fabio Saltara

Arthur Freitas Campos - 11808575

Ivan da Cruz Nunes de Moraes - 11375225

Pedro Ruiz Toniato - 11805593

São Paulo

18 de nov. de 2022

Introdução

O presente trabalho visa simular a instalação de um sistema descrito a seguir na Figura 1, a partir dos conhecimentos desenvolvidos na matéria, PME3332 - Mecânica dos Fluidos : Noções, Laboratório e Aplicações.

Dessa forma, foi criado um programa na linguagem R, com objetivo de realizar a análise de instalação, por meio dessa metodologia numérica. Assim, no decorrer do trabalho, será relatado a base de dados da questão proposta, além do desenvolvimento da metodologia aplicada.

Dados da instalação de um condutor

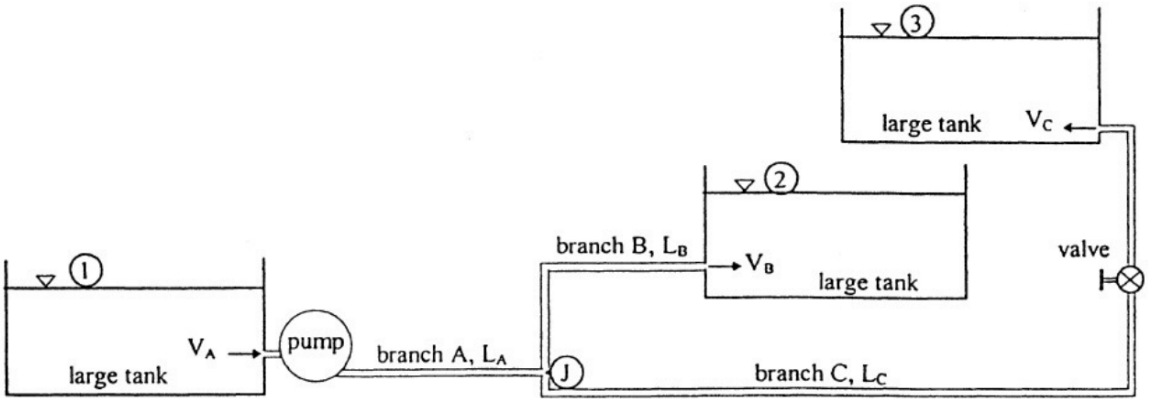

Figura 1 - Sistema composto por condutos

A seguir, constam duas tabela com dados a serem empregados para a instalação do sistema a seguir:

D_C	ε/D_C	L_a	L_b	L_c	L_{eq}	Bomba	
						ρ	v
5 cm	0,001	50m	100m	200m	50m	$10^3 kg/m^3$	$10^{-6} m^2/s$

Tabela 1 - Dados fixos do problema

z_1	z_2	z_3	H_M
6,3 m	15,3 m	35,3 m	57,3 m

Tabela 2 - Variáveis conforme o número USP do integrantes

D	Diâmetro do condutor
ε/D	Rugosidade relativa;
L_a, L_b, L_c	Comprimentos dos trechos retos de A, B e C, respectivamente;
L_{eq}	Comprimento equivalente na válvula do trecho C, que corresponde à equação: $K_s D/f$;
z_1, z_2	Reservatórios 1,2 e 3, respectivamente, de dimensões abertas;
H	Altura manométrica da bomba.

Descrição do método empregado

Inicialmente, temos as declarações que utilizaremos para o funcionamento do programa, os números USP dos integrantes do grupo e assim como os dados contidos na tabela 1. Em seguida, iniciamos as variáveis com o cálculos das variáveis que estão contidas na tabela 2.

```
#Data
d <- 5e-2
relRoughness <- 0.001
la <- 50
lb <- 100
lc <- 200
leq <- 50
rho <- 1000
vsc <- 1e-6
g <- 10
ids <- c("11808575", "11375225", "11805593")
z1 <- calcZs(ids, 1)
z2 <- calcZs(ids, 2)
z3 <- calcZs(ids, 3)
hm <- calcHm(ids, z3)
```

Figura 2 - Declaração dos dados

```
calcZs <- function(ids, n){
  result <- c()

  vectorAux <- c()
  for (i in ids){
    vectorAux <- append(vectorAux, as.double(substr(i, 3, 3)))
  }
  result <- append(result, mean(vectorAux))

  vectorAux <- c()
  for (i in ids){
    vectorAux <- append(vectorAux, as.double(substr(i, 4, 4)))
  }
  result <- append(result, sum(vectorAux) + 2)

  vectorAux <- c()
  for (i in ids){
    vectorAux <- append(vectorAux, as.double(substr(i, 5, 5)))
  }
  result <- append(result, sum(vectorAux) + 2)

  if(n==1){
    return(result[1])
  }
  else if(n==2){
    return(result[1]+result[2])
  }
  else if(n==3){
    return(result[1]+result[2]+result[3])
  }
}
```

Figura 3 - Cálculo das variáveis Z's dependentes dos NUSP's

```
calcHm <- function(ids, z3){
  result <- c()
  vectorAux <- c()
  for (i in ids){
    vectorAux <- append(vectorAux, as.double(substr(i, 6, 6)))
  }
  result <- append(result, sum(vectorAux))
  return(result[1]+z3+10)
```

Figura 4 - Cálculo da variável H_m dependente dos NUSP's

A cota Z_1 em metros foi dada pela média do terceiro dígito do número USP dos integrantes do grupo. A conta de Z_2 em metros foi realizada pela soma de Z_1 com a somatória do quarto dígito dos números USP dos integrantes mais dois metros. A cota Z_3 em metros foi expedida pela soma de Z_2 com a somatória do quinto dígito dos números USP de cada integrante mais dois metros. E por fim, a altura manométrica da bomba H_m em metros é dada pela soma de Z_3 com a somatória do sexto dígito dos números USP mais dez metros.

Assim, serão retornados os valores correspondentes aos dados da tabela dois. Com isso, tais dados servirão de base de cálculo para a carga H_J na junção; os coeficientes de perda de carga distribuída F_a, F_b e F_c; assim como as vazões Q_a, Q_b e Q_c.

z1	6.33333333333333
z2	15.3333333333333
z3	35.3333333333333

Figura 5 - Variáveis Z's obtidas

hm	57.3333333333333

Figura 6 - Variável H_m obtida

Seguindo, calculamos os $f_cv^2_c$, $Re_csqrt(f_c)$, f_c, V_c, Q_c, $f_bv^2_b$, $Re_bsqrt(f_b)$, f_b, V_b, Q_b, $f_av^2_a$, $Re_asqrt(f_a)$, f_a, V_a e Q_a utilizando as seguintes equações:

$$f_aV_a^2 = \frac{(H_m + z_1 - H_j)2gD}{L_a}$$
$$Re_a\sqrt{f_a} = \frac{D}{\upsilon}\sqrt{f_aV_a^2}$$
$$f_a = g(\varepsilon/D, Re_a\sqrt{f_a})$$
$$V_a = V_b + Vc$$
$$Re_a = \frac{V_aD}{\upsilon}$$
$$f_a = g(\varepsilon/D, Re_a)$$
$$Q_a = \pi V_aD^2/4$$

$$f_bV_b^2 = \frac{(H_j - z_2)2gD}{L_b}$$
$$Re_b\sqrt{f_b} = \frac{D}{\upsilon}\sqrt{f_bV_b^2}$$
$$f_b = g(\varepsilon/D, Re_b\sqrt{f_b})$$
$$V_b = \sqrt{f_bV_b^2/f_b}$$
$$Q_b = \pi V_bD^2/4$$

$$f_cV_c^2 = \frac{(H_j - z_3)2gD}{L_c + L_{eq}}$$
$$Re_c\sqrt{f_c} = \frac{D}{\upsilon}\sqrt{f_cV_c^2}$$
$$f_c = g(\varepsilon/D, Re_c\sqrt{f_c})$$
$$V_c = \sqrt{f_cV_c^2/f_c}$$
$$Q_c = \pi V_cD^2/4$$

Figura 7 - Equações utilizadas

Para calcularmos o H_j proveniente dos parâmetros calculados utilizamos a fórmula:

$$H_j = z_1 + H_m - f_a\frac{L_aV_a^2}{2gD}$$

Figura 8 - Equações utilizadas

Assim, criamos a função 'calc' responsável pelo cálculo dessas variáveis:

```
calc <- function(solution, hj, z1, z2, z3, d, hm, la, lb, lc, leq, vsc, relRoughness, g){
  fcVc2 <- ((hj-z3)*2*g*d)/(lc+leq)
  fbVb2 <- ((hj-z2)*2*g*d)/lb
  re2_c <- (d/vsc)*sqrt(fcVc2)
  re2_b <- (d/vsc)*sqrt(fbVb2)
  fc <- clbrk(relRoughness, re2_c)
  fb <- clbrk(relRoughness, re2_b)
  Vc <- sqrt(fcVc2/fc)
  Vb <- sqrt(fbVb2/fb)
  Va <- Vc+Vb
  re_a <- Va*d/vsc
  fa <- hlnd(relRoughness, re_a)
  newHj <- z1+hm-fa*(la/d)*((Va**2)/(2*g))
  Qa <- Va*pi*d**2/4
  Qb <- Vb*pi*d**2/4
  Qc <- Vc*pi*d**2/4
  result <- c()
  result <- append(result, hj)
  result <- append(result, fa)
  result <- append(result, fb)
  result <- append(result, fc)
  result <- append(result, Qa)
  result <- append(result, Qb)
  result <- append(result, Qc)
  result <- append(result, newHj)
  solution[nrow(solution) + 1,] <- result

  return(solution)
}
```

Figura 8 - Função ‘calc’

Para realmente solucionarmos o problema com a finalidade de encontrarmos a convergência, utilizamos a lógica acima repetidas vezes para termos interações até que se nós encontrássemos resultados novos que estivessem condizentes com os resultados anteriores dentro de um erro absoluto $1x10^{-7}$.

```
#solving
maxError <- 1e-7
HJ <- c(50)
er <- 1e10
i <- 1
hjInserido <- c(0)
fa <- c(0)
fb <- c(0)
fc <- c(0)
Qa <- c(0)
Qb <- c(0)
Qc <- c(0)
HjCalculado <- c(0)
solution <- data.frame(hjInserido, fa, fb, fc, Qa, Qb, Qc, HjCalculado)
while (i<1000 & er>maxError){
  solution <- calc(solution, HJ[i], z1, z2, z3, d, hm, la, lb, lc, leq, vsc, relRoughness, g)
  meanHj = (solution$HjCalculado[i+1]+HJ[i])/2
  HJ <- append(HJ, meanHj)
  er <- abs(HJ[i]-meanHj)
  i <- i+1
}
solution <- solution[-1,]
```

Figura 8 - Lógica responsável por chamar a funções e iterar até se encontrar o resultado satisfatório

Resultados Obtidos

Segue abaixo os resultados numéricos que obtivemos. Vale a pena notar que o H_j que obtivemos convergiu para 41,85733 a partir da décima segunda iteração.

	hjInserido	fa	fb	fc	Qa	Qb	Qc	HjCalculado
2	50.00000	0.02057043	0.02101468	0.02266740	0.011133707	0.007974888	0.003158819	30.59674
3	40.29837	0.02082511	0.02123813	0.02436700	0.008504541	0.006731903	0.001772638	44.13231
4	42.21534	0.02075473	0.02118506	0.02378601	0.009106631	0.006994327	0.002112305	41.34417
5	41.77976	0.02076935	0.02119664	0.02389729	0.008975133	0.006935534	0.002039600	41.96891
6	41.87433	0.02076611	0.02119410	0.02387229	0.009003922	0.006948340	0.002055582	41.83289
7	41.85361	0.02076682	0.02119465	0.02387772	0.008997626	0.006945536	0.002052090	41.86267
8	41.85814	0.02076666	0.02119453	0.02387653	0.008999003	0.006946149	0.002052854	41.85616
9	41.85715	0.02076670	0.02119456	0.02387679	0.008998702	0.006946015	0.002052687	41.85758
10	41.85737	0.02076669	0.02119455	0.02387674	0.008998768	0.006946044	0.002052724	41.85727
11	41.85732	0.02076669	0.02119455	0.02387675	0.008998753	0.006946038	0.002052716	41.85734
12	41.85733	0.02076669	0.02119455	0.02387675	0.008998757	0.006946039	0.002052717	41.85732
13	41.85733	0.02076669	0.02119455	0.02387675	0.008998756	0.006946039	0.002052717	41.85733
14	41.85733	0.02076669	0.02119455	0.02387675	0.008998756	0.006946039	0.002052717	41.85733
15	41.85733	0.02076669	0.02119455	0.02387675	0.008998756	0.006946039	0.002052717	41.85733

Figura 8 - Resultados em tabela

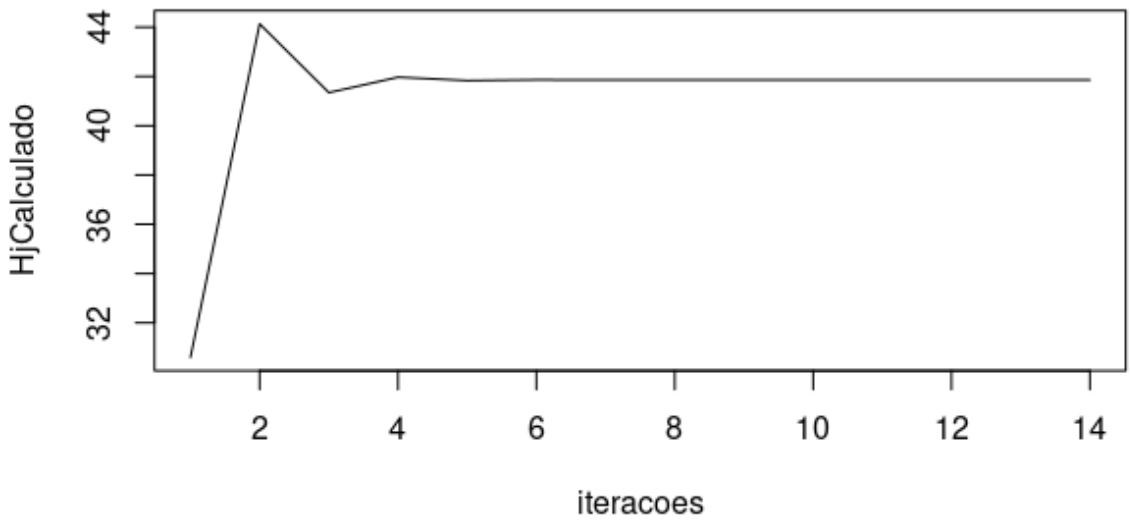

Figura 8 - Resultados de Hj em gráfico, evidenciando a convergência

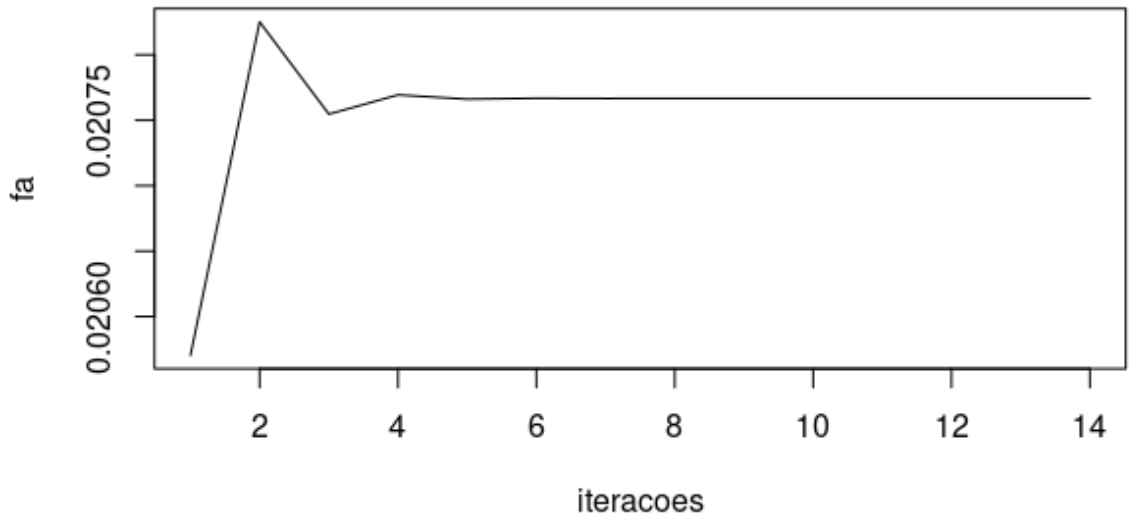

Figura 8 - Resultados de F_a em gráfico, evidenciando a convergência

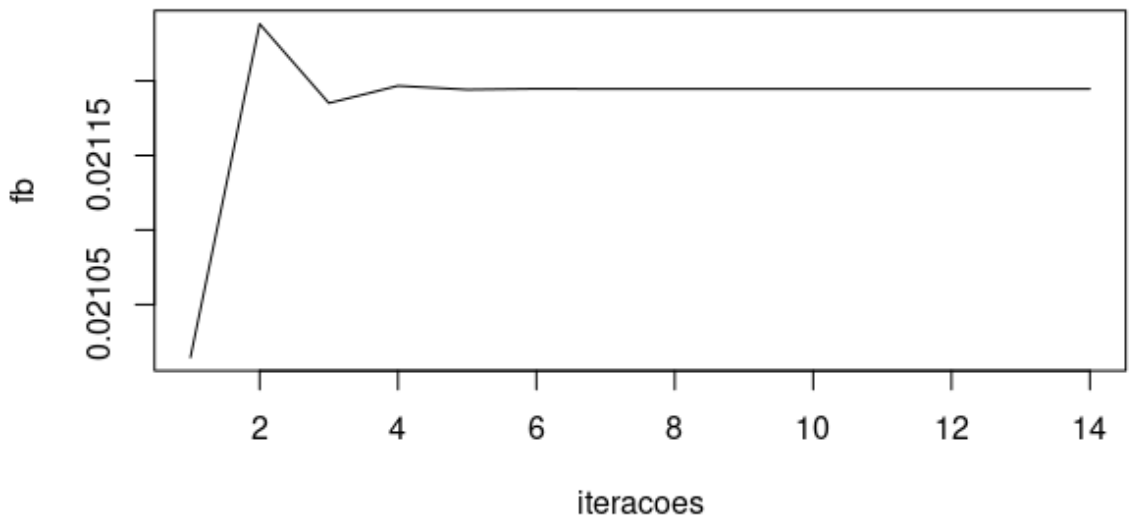

Figura 8 - Resultados de F_b em gráfico, evidenciando a convergência

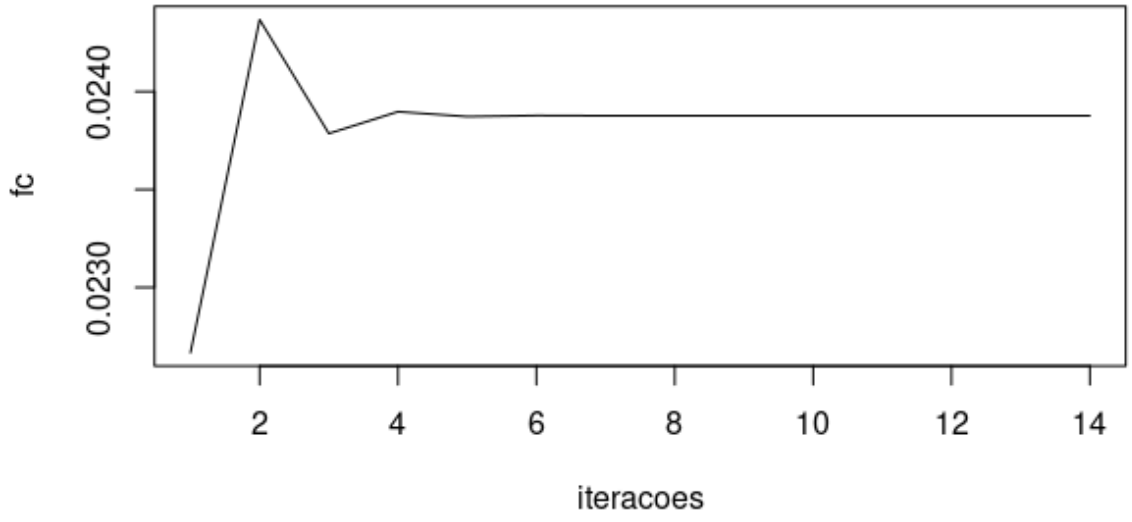

Figura 8 - Resultados de F_c em gráfico, evidenciando a convergência

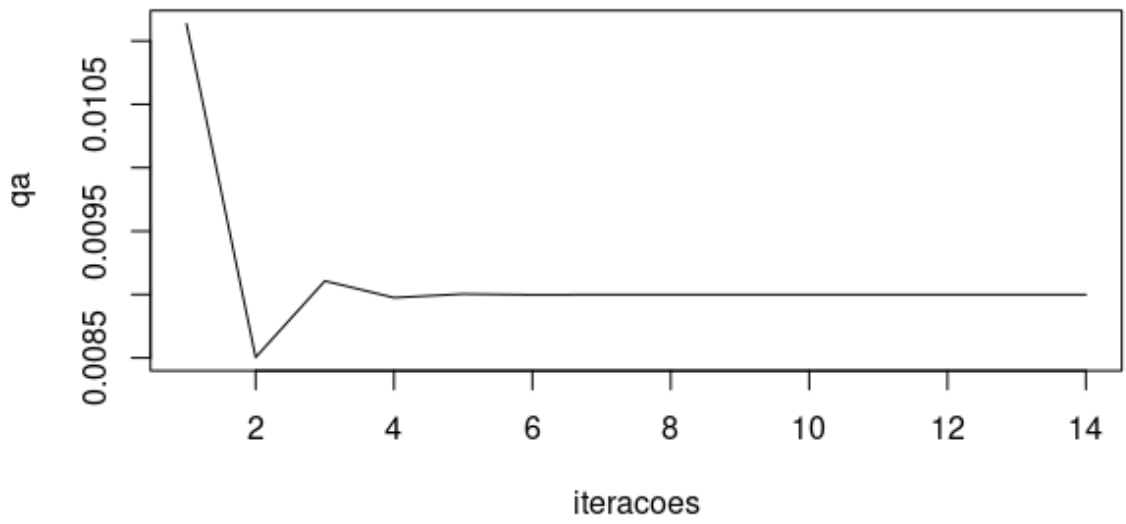

Figura 8 - Resultados de Q_a em gráfico, evidenciando a convergência

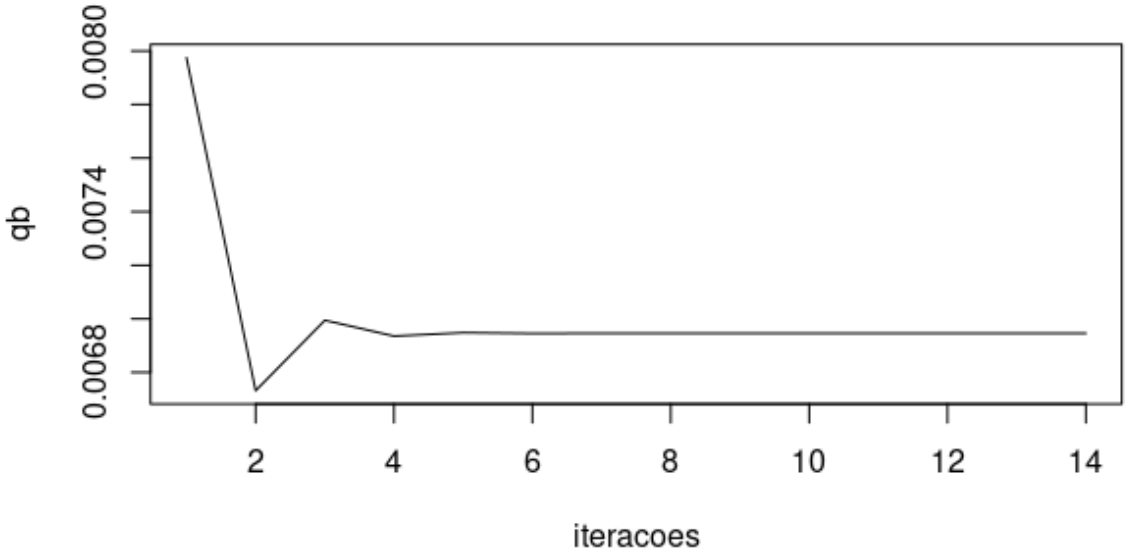

Figura 8 - Resultados de Q_b em gráfico, evidenciando a convergência

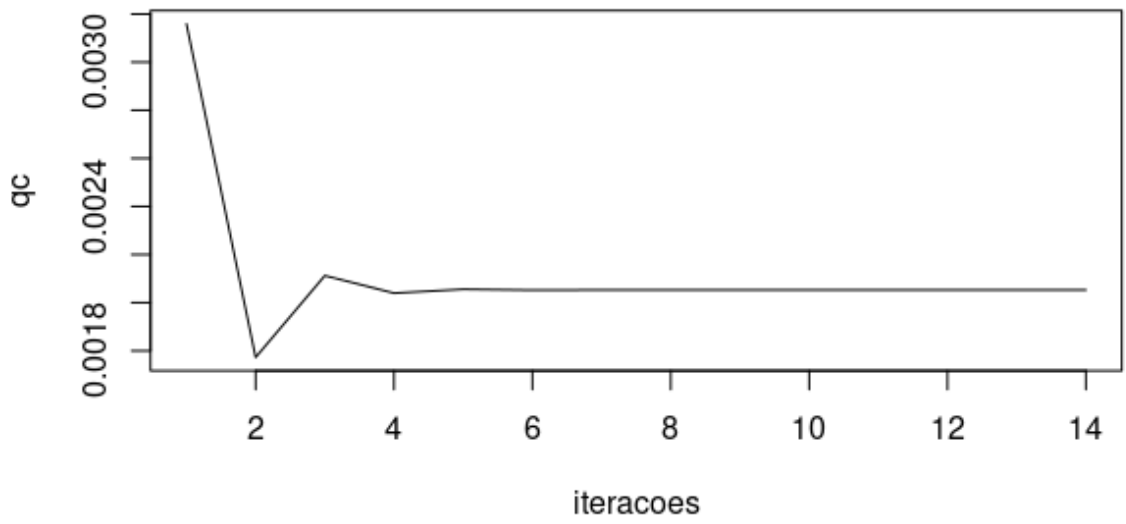

Figura 8 - Resultados de Q_c em gráfico, evidenciando a convergência

Script em R

```
# ////////////////////////////////////////////////////////////////////////////#
#                                                                             #
#                                                                             #
#    PME3332 - Mecânica dos Fluidos : Noções, Laboratório e Aplicações (2022)  #
#    Primeiro Trabalho                                                        #
#                                                                             #
#    11808575 - Arthur Freitas Campos - arthurfc@usp.br                       #
#    11375225 - Ivan da Cruz Nunes de Moraes - ivandacruz1805@usp.br           #
#    11805593 - Pedro Ruiz Toniato - pedro.toniato@usp.br                     #
#                                                                             #
#                                                                             #
# ////////////////////////////////////////////////////////////////////////////#

#Functions
calcZs <- function(ids, n){
  result <- c()

  vectorAux <- c()
  for (i in ids){
    vectorAux <- append(vectorAux, as.double(substr(i, 3, 3)))
  }
  result <- append(result, mean(vectorAux))

  vectorAux <- c()
  for (i in ids){
    vectorAux <- append(vectorAux, as.double(substr(i, 4, 4)))
  }
  result <- append(result, sum(vectorAux) + 2)

  vectorAux <- c()
  for (i in ids){
    vectorAux <- append(vectorAux, as.double(substr(i, 5, 5)))
```

```
    }
    result <- append(result, sum(vectorAux) + 2)

    if(n==1){
      return(result[1])
    }
    else if(n==2){
      return(result[1]+result[2])
    }
    else if(n==3){
      return(result[1]+result[2]+result[3])
    }
  }

  calcHm <- function(ids, z3){
    result <- c()
    vectorAux <- c()
    for (i in ids){
      vectorAux <- append(vectorAux, as.double(substr(i, 6, 6)))
    }
    result <- append(result, sum(vectorAux))
    return(result[1]+z3+10)
  }

  clbrk <- function(relRoughness, re2){
    x <- (-2*log10(relRoughness/3.7 + 2.51/re2))**(-2)
    return (x)
  }

  hlnd <- function(relRoughness, re){
    x <- (-1.8*log10((relRoughness/3.7)**1.11 + 6.9/re))**(-2)
    return (x)
  }

  calc <- function(solution, hj, z1, z2, z3, d, hm, la, lb, lc, leq, vsc, relRoughness, g){
    fcVc2 <- ((hj-z3)*2*g*d)/(lc+leq)
    fbVb2 <- ((hj-z2)*2*g*d)/lb
    re2_c <- (d/vsc)*sqrt(fcVc2)
    re2_b <- (d/vsc)*sqrt(fbVb2)
    fc <- clbrk(relRoughness, re2_c)
    fb <- clbrk(relRoughness, re2_b)
    Vc <- sqrt(fcVc2/fc)
    Vb <- sqrt(fbVb2/fb)
    Va <- Vc+Vb
    re_a <- Va*d/vsc
    fa <- hlnd(relRoughness, re_a)
    HjCalculado <- z1+hm-fa*(la/d)*((Va**2)/(2*g))
    Qa <- Va*pi*d**2/4
    Qb <- Vb*pi*d**2/4
    Qc <- Vc*pi*d**2/4
    result <- c()
    result <- append(result, hj)
    result <- append(result, fa)
    result <- append(result, fb)
    result <- append(result, fc)
    result <- append(result, Qa)
    result <- append(result, Qb)
    result <- append(result, Qc)
    result <- append(result, HjCalculado)
    solution[nrow(solution) + 1,] <- result

    return(solution)
  }
```

```
#Data
d <- 5e-2
relRoughness <- 0.001
la <- 50
lb <- 100
lc <- 200
leq <- 50
rho <- 1000
vsc <- 1e-6
g <- 10
ids <- c("11808575", "11375225", "11805593")
z1 <- calcZs(ids, 1)
z2 <- calcZs(ids, 2)
z3 <- calcZs(ids, 3)
hm <- calcHm(ids, z3)

#solving
maxError <- 1e-7
HJ <- c(50)
er <- 1e10
i <- 1
hjInserido <- c(0)
fa <- c(0)
fb <- c(0)
fc <- c(0)
Qa <- c(0)
Qb <- c(0)
Qc <- c(0)
HjCalculado <- c(0)
solution <- data.frame(hjInserido, fa, fb, fc, Qa, Qb, Qc, HjCalculado)
while (i<1000 & er>maxError){
  solution <- calc(solution, HJ[i], z1, z2, z3, d, hm, la, lb, lc, leq, vsc, relRoughness, g)
  meanHj = (solution$HjCalculado[i+1]+HJ[i])/2
  HJ <- append(HJ, meanHj)
  er <- abs(HJ[i]-meanHj)
  i <- i+1
}
solution <- solution[-1,]

#plot
iteracoes <- 1:(i-1)
HjCalculado <- solution$HjCalculado
graphHj <- data.frame(iteracoes, HjCalculado)
plottedHj <- plot(graph, type="l")
fa <- solution$fa
graphFa <- data.frame(iteracoes, fa)
plottedFa <- plot(graphFa, type="l")
fb <- solution$fb
graphFb <-data.frame(iteracoes, fb)
plottedFb <- plot(graphFb, type="l")
fc <- solution$fc
graphFc <- data.frame(iteracoes, fc)
plottedFc <- plot(graphFc, type="l")
qa <- solution$Qa
graphQa <- data.frame(iteracoes, qa)
plottedQa <- plot(graphQa, type="l")
qb <- solution$Qb
graphQb <- data.frame(iteracoes, qb)
plottedQb <- plot(graphQb, type="l")
qc <- solution$Qc
graphQc <- data.frame(iteracoes, qc)
plottedQc <- plot(graphQc, type="l")
```