

# SPIRAL OPTIMIZATION ALGORITHM

Tugas Kuliah  
SK5001 Analisis Numerik Lanjut

Mohammad Rizka Fadhli  
NIM: 20921004

17 October 2021

## PENDAHULUAN

### Bahasa yang Digunakan

Untuk membuat program *spiral optimization algorithm*, saya menggunakan bahasa **R** yang bisa dieksekusi pada versi minimal 3.5.3.

### Spiral Optimization Algorithm

*Spiral Optimization Algorithm* adalah salah satu metode *meta heuristic* yang digunakan untuk mencari minimum global dari suatu sistem persamaan.

Algoritmanya mudah dipahami dan intuitif tanpa harus memiliki latar keilmuan tertentu. Proses kerjanya adalah dengan melakukan *random number generating* pada suatu selang dan melakukan rotasi sekaligus kontraksi dengan titik paling minimum pada setiap iterasi sebagai pusatnya.

Berikut adalah algoritmanya:

```
INPUT
m >= 2 # jumlah titik
theta # sudut rotasi (0 <= theta <= 2pi)
r      # konstraksi
k_max # iterasi maksimum

PROCESS
1 generate m buah titik secara acak
    x_i

2 initial condition
    k = 0 # untuk keperluan iterasi

3 cari x_* yang memenuhi
    min(f(x_*))

4 lakukan rotasi dan konstraksi semua x_i
    x_* sebagai pusat rotasi
    k = k + 1

5 ulangi proses 3 dan 4
```

```

6 hentikan proses saat k = k_max
    output x_*

```

Berdasarkan algoritma di atas, salah satu proses yang penting adalah melakukan **rotasi** dan **konstraksi** terhadap semua titik yang telah *di-generate*.

Agar memudahkan, saya akan memberikan ilustrasi geometri beserta operasi matriks aljabar terkait kedua hal tersebut.

## Membuat Program

Untuk menyelesaikan tugas soal yang diberikan, pertama-tama saya harus membuat program *spiral optimization algorithm*. Untuk membuatnya, saya akan melakukannya perlahan-lahan dengan bantuan ilustrasi geometri. Berikut adalah langkah-langkah yang ditempuh:

1. **Pertama** saya akan membuat program yang bisa merotasi suatu titik berdasarkan suatu  $\theta$  tertentu.
2. **Kedua** saya akan memodifikasi program tersebut untuk melakukan rotasi sekaligus konstraksi dengan rasio  $r$  tertentu.
3. **Ketiga** saya akan memodifikasi program tersebut untuk melakukan rotasi sekaligus konstraksi dengan **titik pusat rotasi tertentu**.

Dari program yang terakhir, akan saya pakai untuk **membangun program spiral optimization algorithm** yang sebenarnya.

## Langkah dan Ilustrasi Geometri

### Operasi Matriks Rotasi

Misalkan saya memiliki titik  $x \in \mathbb{R}^2$ . Untuk melakukan rotasi sebesar  $\theta$ , saya bisa menggunakan suatu matriks  $A_{2 \times 2}$  berisi fungsi-fungsi trigonometri sebagai berikut:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

Berdasarkan operasi matriks di atas, saya membuat **program** di **R** dengan beberapa modifikasi. Sebagai contoh, saya akan membuat program yang bertujuan untuk melakukan rotasi suatu titik  $x \in \mathbb{R}$  sebanyak  $n$  kali:

```

# mendefinisikan program
rotasi_kan = function(x0,rot){
  # menghitung theta
  theta = 2*pi/rot

  # definisi matriks rotasi
  A = matrix(c(cos(theta),-sin(theta),
              sin(theta),cos(theta)),
             ncol = 2,byrow = T)

  # membuat template
  temp = vector("list")
  temp[[1]] = x0

  # proses rotasi
  for(i in 2:rot){
    xk = A %*% x0
    temp[[i]] = xk
    x0 = xk
  }
}

```

```

}

# membuat template data frame
final = data.frame(x = rep(NA,rot),
                    y = rep(NA,rot))

# gabung data dari list
for(i in 1:rot){
  tempura = temp[[i]]
  final$x[i] = tempura[1]
  final$y[i] = tempura[2]
}

# membuat plot
plot =
  ggplot() +
  geom_point(aes(x,y),data = final) +
  geom_point(aes(x[1],y[1]),
             data = final,
             color = "red") +
  coord_equal() +
  labs(title = "titik merah adalah titik initial")

# enrich dengan garis panah
panah = data.frame(
  x_start = final$x[1:(rot-1)],
  x_end = final$x[2:rot],
  y_start = final$y[1:(rot-1)],
  y_end = final$y[2:rot]
)
# menambahkan garis panah ke plot
plot =
  plot +
  geom_segment(aes(x = x_start,
                   xend = x_end,
                   y = y_start,
                   yend = y_end),
               data = panah,
               arrow = arrow(length = unit(.3,"cm"))
  )

# menyiapkan output
list("Grafik rotasi" = plot,
     "Titik-titik rotasi" = final)
}

```

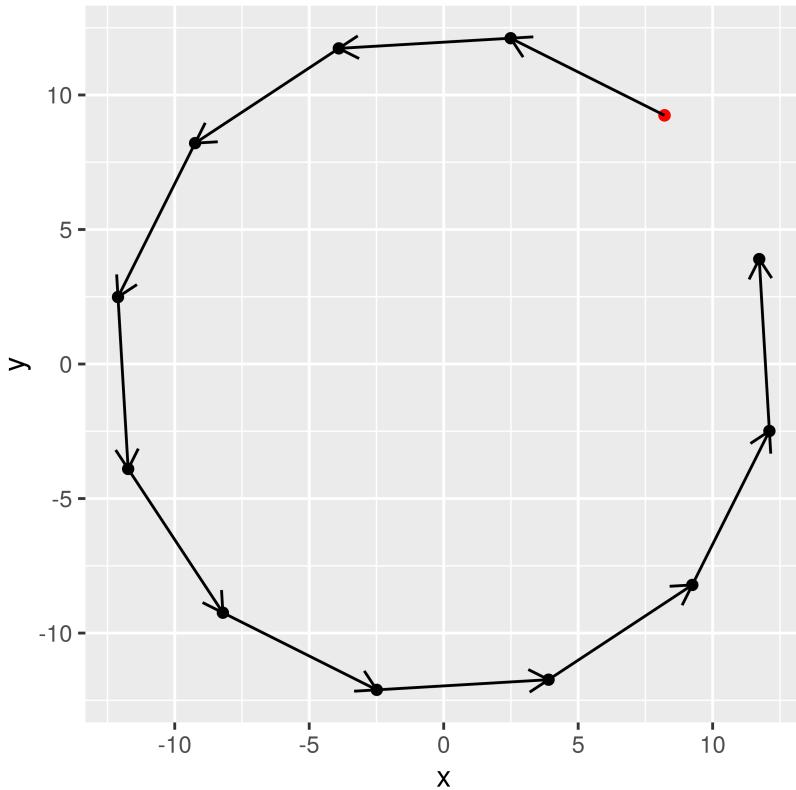
Berikut adalah uji coba dengan titik sembarang berikut ini:

```
# uji coba
rot = 12 # berapa banyak rotasi
x0 = rand_titik(0,10) # generate random titik

rotasi_kan(x0,rot)
```

```
## $`Grafik rotasi`
```

titik merah adalah titik initial



```
##
## $`Titik-titik rotasi`  

##          x          y  

## 1    8.21063   9.24304  

## 2    2.48909  12.11002  

## 3   -3.89939  11.73213  

## 4   -9.24304   8.21063  

## 5  -12.11002   2.48909  

## 6  -11.73213  -3.89939  

## 7  -8.21063  -9.24304  

## 8  -2.48909 -12.11002  

## 9   3.89939 -11.73213  

## 10  9.24304 -8.21063  

## 11 12.11002 -2.48909  

## 12 11.73213   3.89939
```

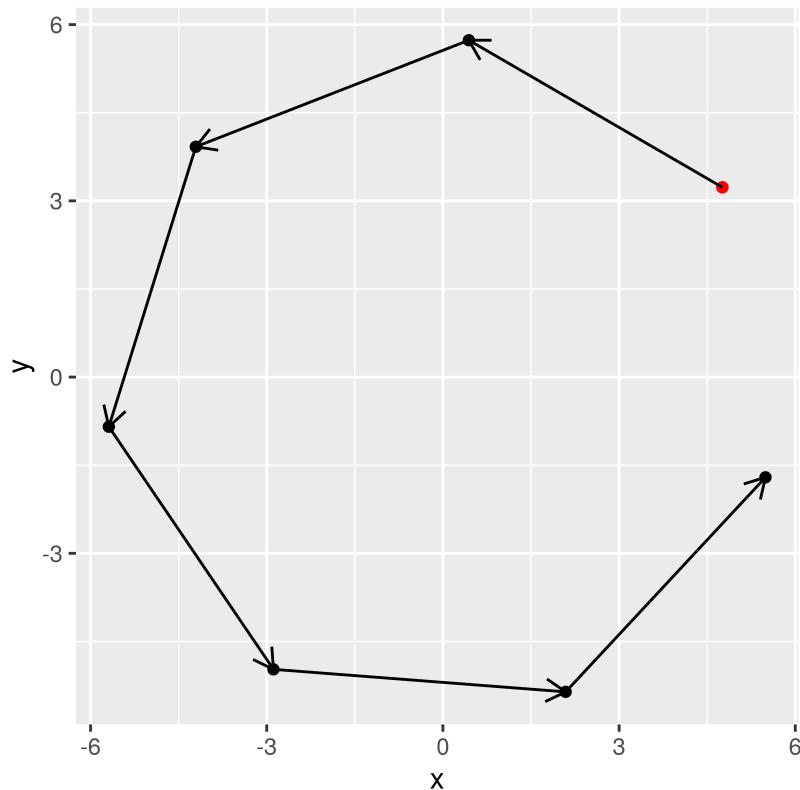
Uji coba kembali dengan titik sembarang lainnya berikut ini:

```
# uji coba
rot = 7 # berapa banyak rotasi
x0 = rand_titik(0,10) # generate random titik

rotasi_kan(x0,rot)
```

```
## $`Grafik rotasi`
```

titik merah adalah titik initial



```
##
## $`Titik-titik rotasi`  
##      x      y  
## 1  4.758599  3.230136  
## 2  0.441516  5.734379  
## 3 -4.208037  3.920518  
## 4 -5.688853 -0.845573  
## 5 -2.885846 -4.974930  
## 6  2.090262 -5.358064  
## 7  5.492360 -1.706466
```

## Operasi Matriks Rotasi dan Kontraksi

Jika pada sebelumnya saya **hanya melakukan rotasi**, kali ini saya akan memodifikasi operasi matriks agar melakukan rotasi dan konstraksi secara bersamaan. Untuk melakukan hal tersebut, saya akan definisikan  $r, 0 < r < 1$  dan melakukan operasi matriks sebagai berikut:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} r \\ r \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}$$

Oleh karena itu saya akan modifikasi program **R** sebelumnya menjadi sebagai berikut:

```
# mendefinisikan program
rotasi_konstraksi_kan = function(x0,rot,r){
  # menghitung theta
  theta = 2*pi/rot

  # definisi matriks rotasi
  A = matrix(c(cos(theta),-sin(theta),
              sin(theta),cos(theta)),
             ncol = 2,byrow = T)

  # membuat template
  temp = vector("list")
  temp[[1]] = x0

  # proses rotasi dan konstraksi
  for(i in 2:rot){
    xk = A %*% x0
    xk = r * xk
    temp[[i]] = xk
    x0 = xk
  }

  # membuat template data frame
  final = data.frame(x = rep(NA,rot),
                      y = rep(NA,rot))

  # gabung data dari list
  for(i in 1:rot){
    tempura = temp[[i]]
    final$x[i] = tempura[1]
    final$y[i] = tempura[2]
  }

  # membuat plot
  plot =
    ggplot() +
    geom_point(aes(x,y),data = final) +
    geom_point(aes(x[1],y[1]),
               data = final,
               color = "red") +
    coord_equal() +
    labs(title = "titik merah adalah titik initial")

  # enrich dengan garis panah
```

```

panah = data.frame(
  x_start = final$x[1:(rot-1)],
  x_end = final$x[2:rot],
  y_start = final$y[1:(rot-1)],
  y_end = final$y[2:rot]
)
# menambahkan garis panah ke plot
plot =
  plot +
  geom_segment(aes(x = x_start,
                    xend = x_end,
                    y = y_start,
                    yend = y_end),
               data = panah,
               arrow = arrow(length = unit(.3,"cm")))
)

# menyiapkan output
list("Grafik rotasi" = plot,
     "Titik-titik rotasi" = final)
}

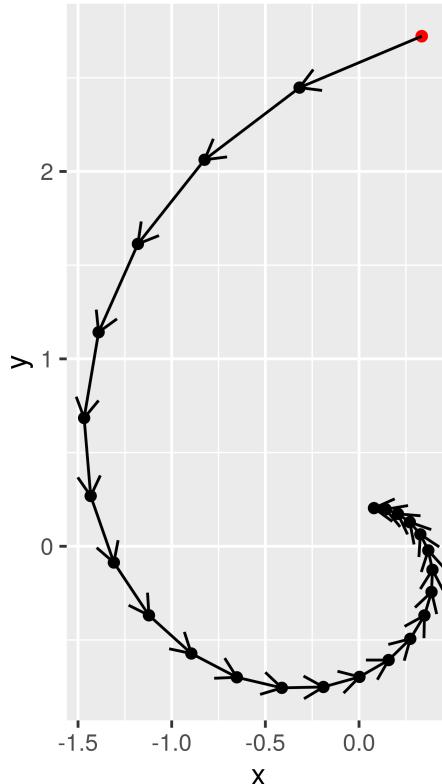
```

Saya akan uji coba untuk sembarang titik berikut ini:

```
# uji coba
rot = 25 # berapa banyak rotasi
x0 = rand_titik(0,4) # generate random titik
r = .9
rotasi_konstraksi_kan(x0,rot,r)
```

```
## $`Grafik rotasi`
```

titik merah adalah titik initial



```
##
## $`Titik-titik rotasi`
##           x         y
## 1   0.33467948  2.7218219
## 2   -0.31745220  2.4475880
## 3   -0.82455232  2.0625708
## 4   -1.18042921  1.6134422
## 5   -1.39013155  1.1422729
## 6   -1.46747677  0.6846072
## 7   -1.43246536  0.2683371
## 8   -1.30877510 -0.0866995
## 9   -1.12148661 -0.3685094
## 10  -0.89514764 -0.5722509
## 11  -0.65224073 -0.6991981
## 12  -0.41207931 -0.7554934
## 13  -0.19012455 -0.7508144
## 14  0.00231165 -0.6970574
```

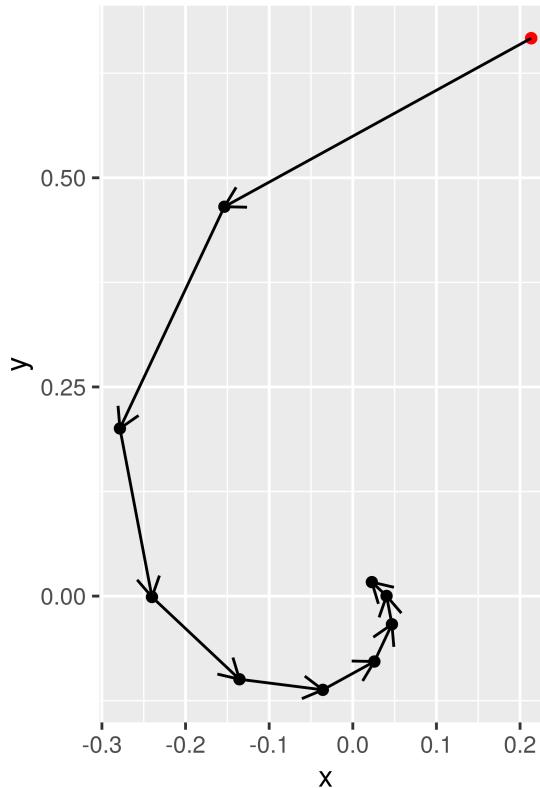
```
## 15 0.15803113 -0.6071248
## 16 0.27364689 -0.4938751
## 17 0.34908437 -0.3692753
## 18 0.38695706 -0.2437741
## 19 0.39188182 -0.1258949
## 20 0.36979102 -0.0220343
## 21 0.32728777  0.0635591
## 22 0.27107903  0.1286599
## 23 0.20750956  0.1728292
## 24 0.14220846  0.1971044
## 25 0.07985056  0.2036501
```

Saya akan uji coba kembali untuk sembarang titik lainnya berikut ini:

```
# uji coba
rot = 10 # berapa banyak rotasi
x0 = rand_titik(0,4) # generate random titik
r = .7
rotasi_konstraksi_kan(x0,rot,r)
```

```
## $`Grafik rotasi`
```

titik merah adalah titik initial



```
##
## $`Titik-titik rotasi`  
##           x          y  
## 1   0.2133223  0.66685844  
## 2   -0.1535717  0.46542127  
## 3   -0.2784669  0.20038656  
## 4   -0.2401481 -0.00109383  
## 5   -0.1355487 -0.09942832  
## 6   -0.0358531 -0.11207890  
## 7    0.0258108 -0.07822335  
## 8    0.0468019 -0.03367897  
## 9    0.0403617  0.00018384  
## 10   0.0227817  0.01671092
```

#### Catatan penting:

Terlihat bahwa semakin banyak rotasi dan konstraksi yang dilakukan akan membuat titik *initial* menuju **pusat** (0, 0).

## Operasi Matriks Rotasi dan Kontraksi dengan Titik $x^*$ Sebagai Pusatnya

Salah satu prinsip utama dari *spiral optimization algorithm* adalah menjadikan titik  $x^*$  sebagai pusat rotasi di setiap iterasinya. Operasi matriksnya adalah sebagai berikut:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} x_1^* \\ x_2^* \end{bmatrix} + \begin{bmatrix} r \\ r \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \left( \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} - \begin{bmatrix} x_1^* \\ x_2^* \end{bmatrix} \right)$$

Oleh karena itu kita akan modifikasi program bagian sebelumnya menjadi seperti ini:

```
# mendefinisikan program
rotasi_konstraksi_pusat_kan = function(x0,rot,r,x_bin){
  # pusat rotasi
  pusat = x_bin

  # menghitung theta
  theta = 2*pi/rot

  # definisi matriks rotasi
  A = matrix(c(cos(theta),-sin(theta),
              sin(theta),cos(theta)),
             ncol = 2,byrow = T)

  # membuat template
  temp = vector("list")
  temp[[1]] = x0

  # proses rotasi dan kontraksi
  for(i in 2:rot){
    xk = A %*% (x0-pusat) # diputar dengan x_bin sebagai pusat
    xk = pusat + (r * xk)
    temp[[i]] = xk
    x0 = xk
  }

  # membuat template data frame
  final = data.frame(x = rep(NA,rot),
                      y = rep(NA,rot))

  # gabung data dari list
  for(i in 1:rot){
    tempura = temp[[i]]
    final$x[i] = tempura[1]
    final$y[i] = tempura[2]
  }

  # membuat plot
  plot =
    ggplot() +
    geom_point(aes(x,y),data = final) +
    geom_point(aes(x[1],y[1]),
               data = final,
               color = "red") +
    geom_point(aes(x = pusat[1],
                  y = pusat[2]),
```

```

        color = "blue") +
  labs(title = "titik merah adalah titik initial\ntitik biru adalah pusat rotasi")

# enrich dengan garis panah
panah = data.frame(
  x_start = final$x[1:(rot-1)],
  x_end = final$x[2:rot],
  y_start = final$y[1:(rot-1)],
  y_end = final$y[2:rot]
)
# menambahkan garis panah ke plot
plot =
  plot +
  geom_segment(aes(x = x_start,
                    xend = x_end,
                    y = y_start,
                    yend = y_end),
               data = panah,
               arrow = arrow(length = unit(.3,"cm")))
)

# menyiapkan output
list("Grafik rotasi" = plot,
     "Titik-titik rotasi" = final)
}

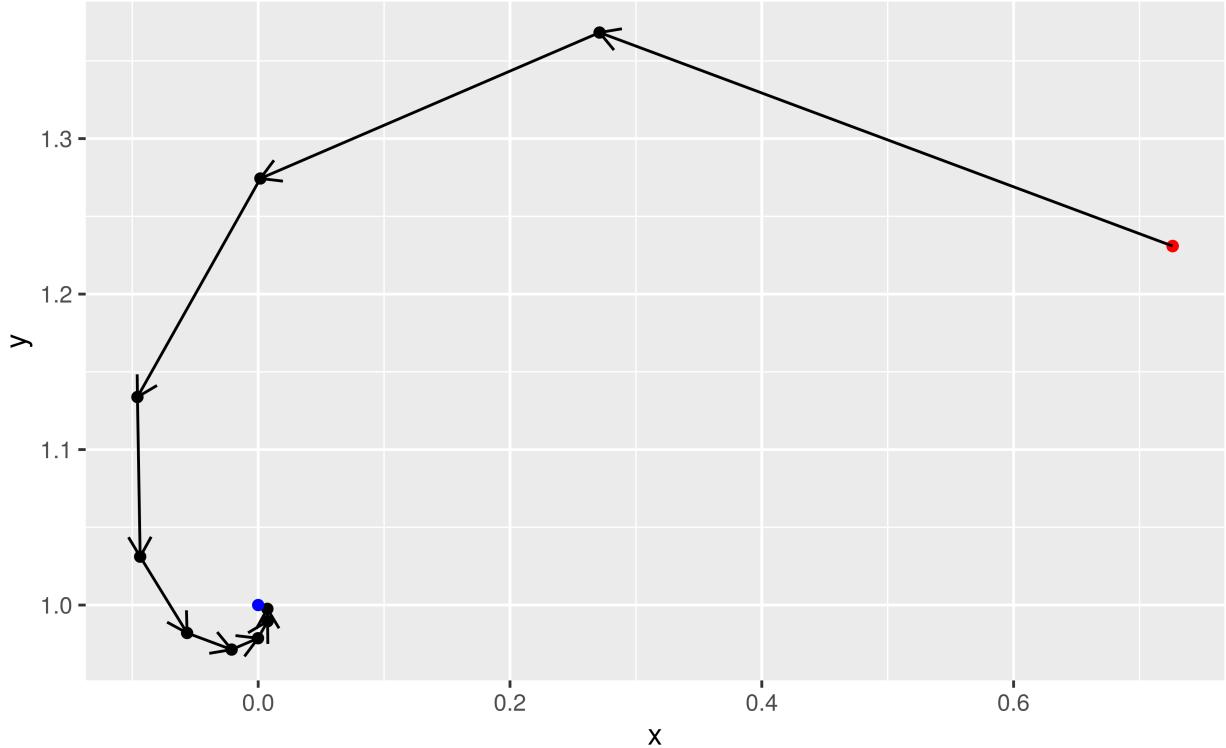
```

Saya akan coba dengan sembarang titik berikut:

```
# uji coba
rot = 10 # berapa banyak rotasi
x0 = rand_titik(0,4) # generate random titik
x_bintang = c(0,1) # contoh pusat rotasi
r = .6
rotasi_konstraksi_pusat_kan(x0,rot,r,x_bintang)
```

```
## $`Grafik rotasi`
```

titik merah adalah titik initial  
titik biru adalah pusat rotasi



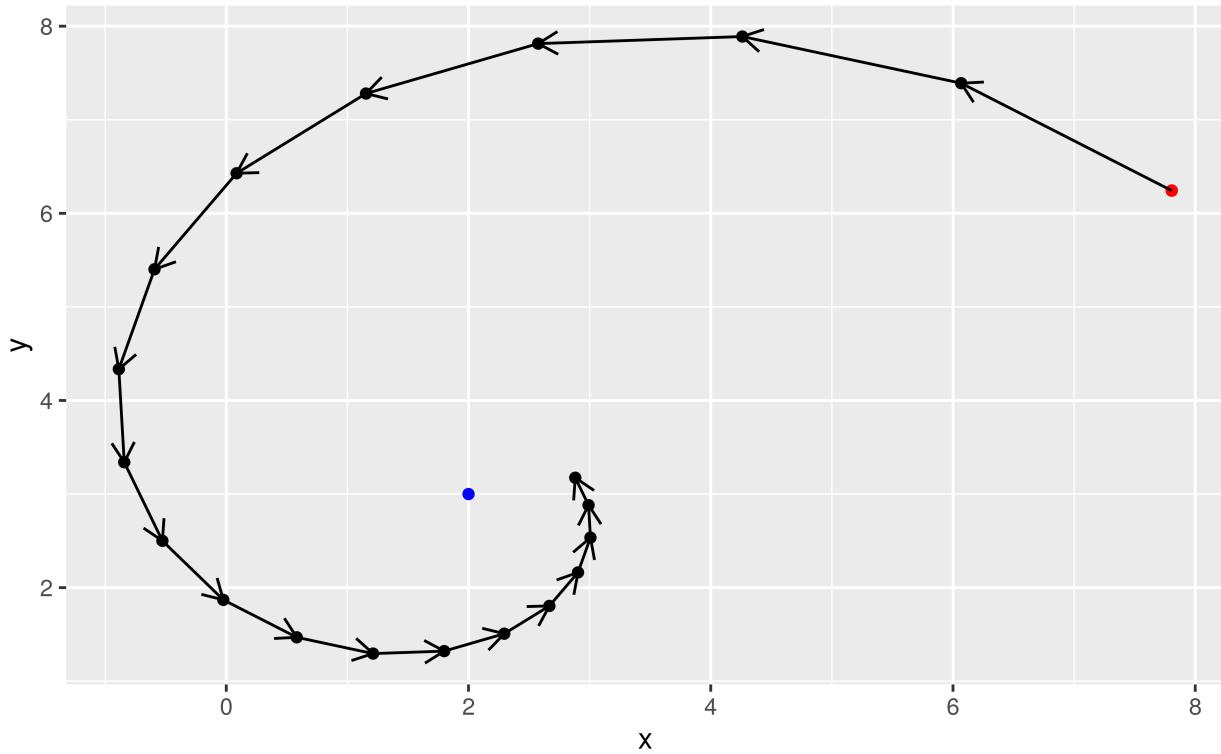
```
##
## $`Titik-titik rotasi`  
##          x      y
## 1   0.726300165 1.230892
## 2   0.271124444 1.368223
## 3   0.001745080 1.274357
## 4   -0.095910641 1.133791
## 5   -0.093740235 1.031119
## 6   -0.056477101 0.982046
## 7   -0.021082637 0.971367
## 8   -0.000135697 0.978666
## 9   0.007458011 0.989596
## 10  0.007289241 0.997580
```

Saya akan coba kembali dengan sembarang titik lainnya:

```
# uji coba
rot = 20 # berapa banyak rotasi
x0 = rand_titik(0,10) # generate random titik
x_bintang = c(2,3) # contoh pusat rotasi
r = .9
rotasi_konstraksi_pusat_kan(x0,rot,r,x_bintang)
```

```
## $`Grafik rotasi`
```

titik merah adalah titik initial  
titik biru adalah pusat rotasi



```
##
## $`Titik-titik rotasi`#
##      x      y
## 1  7.8059204 6.24283
## 2  6.0677032 7.39041
## 3  4.2607126 7.88927
## 4  2.5752782 7.81371
## 5  1.1536426 7.28030
## 6  0.0851439 6.42834
## 7  -0.5924959 5.40194
## 8  -0.8870648 4.33493
## 9  -0.8424496 3.33970
## 10 -0.5274718 2.50023
## 11 -0.0243993 1.86930
## 12  0.5816796 1.46916
## 13  1.2117383 1.29522
```

```
## 14 1.7994129 1.32156
## 15 2.2951066 1.50755
## 16 2.6676690 1.80461
## 17 2.9039474 2.16250
## 18 3.0066573 2.53454
## 19 2.9911009 2.88156
## 20 2.8812749 3.17426
```

## Program *Spiral Optimization Algorithm*

Berbekal program yang telah dituliskan di bagian sebelumnya, kita akan sempurnakan program untuk melakukan *spiral optimization* sebagai berikut:

```
soa_mrf = function(N,      # banyak titik
                    a,      # batas bawah
                    b,      # batas atas
                    rot,    # berapa banyak rotasi
                    k_max, # iterasi maks
                    r){    # berapa rate konstraksi

# N pasang titik random di selang [a,b] di R2
x1 = runif(N,a,b)
x2 = runif(N,a,b)

# hitung theta
theta = 2*pi / rot

# definisi matriks rotasi
A = matrix(c(cos(theta),-sin(theta),
             sin(theta),cos(theta)),
            ncol = 2,byrow = T)

# bikin data frame
temp = data.frame(x1,x2) %>% mutate(f = f(x1,x2))

# proses iterasi
for(i in 1:k_max){
  # mencari titik x* dengan min(f)
  f_min = temp %>% filter(f == min(f))
  pusat = c(f_min$x1,f_min$x2)

  for(j in 1:N){
    # kita akan ambil titiknya satu persatu
    x0 = c(temp$x1[j],temp$x2[j])

    # proses rotasi dan konstraksi terhadap pusat x*
    xk = A %*% (x0-pusat) # diputar dengan x_bin sebagai pusat
    xk = pusat + (r * xk)

    # proses mengembalikan nilai ke temp
    temp$x1[j] = xk[1]
    temp$x2[j] = xk[2]
  }

  # hitung kembali nilai f(x1,x2)
  temp = temp %>% mutate(f = f(x1,x2))
}

# proses output hasil
output = temp %>% filter(f == min(f))
return(output)
}
```

## Contoh Penggunaan Program

Kita akan coba performa program tersebut untuk menyelesaikan fungsi berikut:

$$f(x_1, x_2) = \frac{x_1^4 - 16x_1^2 + 5x_1}{2} + \frac{x_2^4 - 16x_2^2 + 5x_2}{2}$$

$$-4 \leq x_1, x_2 \leq 4$$

Dengan  $r = 0.8, N = 50, rot = 20, k_{max} = 60$ .

```
# definisi
N = 50
a = -4
b = 4
k_max = 60
r = .8
rot = 20
f = function(x1,x2){
  ((x1^4 - 16 * x1^2 + 5 * x1)/2) + ((x2^4 - 16 * x2^2 + 5*x x2)/2)
}

# solving
soa_mrf(N,a,b,rot,k_max,r)

##           x1           x2           f
## 1 -2.90353 -2.90354 -78.3323
```

### Catatan

Pada algoritma ini, penentuan  $\theta, r, x$  menjadi penentu hasil perhitungan.

## Mengubah Optimisasi Menjadi Pencarian Akar

*Spiral optimization algorithm* adalah suatu metode untuk mencari solusi minimum global. Jika kita hendak memakainya untuk mencari suatu akar persamaan (atau sistem persamaan), kita bisa melakukan modifikasi pada fungsi-fungsi yang terlibat (membuat fungsi *merit*).

Misalkan suatu sistem persamaan non linear:

$$g_1(x_1, x_2, \dots, x_n) = 0$$

$$g_2(x_1, x_2, \dots, x_n) = 0$$

$$g_n(x_1, x_2, \dots, x_n) = 0$$

dengan  $(x_1, x_2, \dots, x_n)^T \in D$

$$D = a_1, b_1 \times a_2, b_2 \times \dots \times a_n, b_n \subset \mathbb{R}^n$$

### Pencarian Akar

Sistem di atas memiliki solusi  $x = (x_1, x_2, \dots, x_n)^T$  jika  $F(x)$  yang kita definisikan sebagai:

$$F(x) = \frac{1}{1 + \sum_{i=1}^n |g_i(x)|}$$

memiliki nilai maksimum sama dengan 1.

**Kelak  $F(x)$  akan digunakan untuk menjawab soal-soal yang ada dalam tugas ini.**

## SOAL 1

Tentukanlah akar-akar sistem persamaan berikut dengan **SOA**. Buatlah terlebih dahulu *contour plot*-nya:

$$f_1(x_1, x_2) = \cos(2x_1) - \cos(2x_2) - 0.4 = 0$$

$$f_2(x_1, x_2) = 2(x_2 - x_1) + \sin(x_2) - \sin(x_1) - 1.2 = 0$$

dengan  $-10 \leq x_1, x_2 \leq 10$

## JAWAB

### *Contour Plot*

Pertama-tama, saya akan buat *contour plot* dari  $f_1(x_1, x_2)$  sebagai berikut:

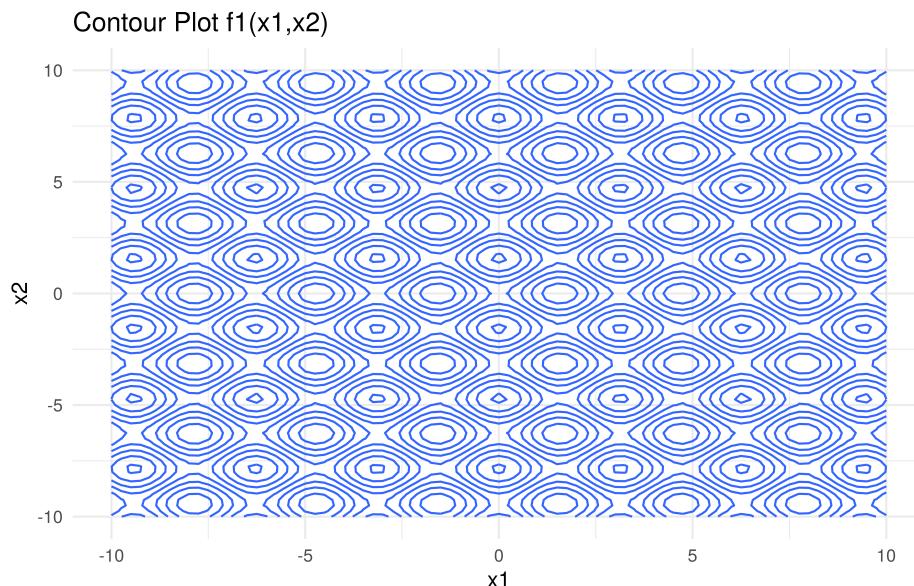


Figure 1: Contour Plot Soal 1: f1

Selanjutnya, saya akan buat *contour plot* dari  $f_2(x_1, x_2)$  sebagai berikut:

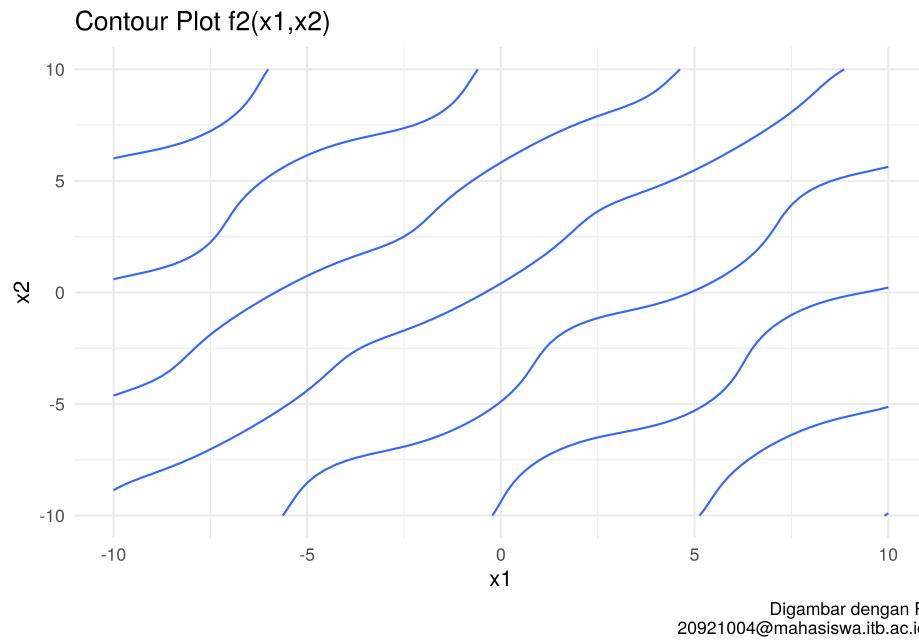


Figure 2: Contour Plot Soal 1:  $f_2$

## Grafik Sistem Persamaan

Kita akan mencari akar-akar sistem persamaan saat  $f_1 = 0$  dan  $f_2 = 0$  dengan bantuan grafik sebagai berikut:

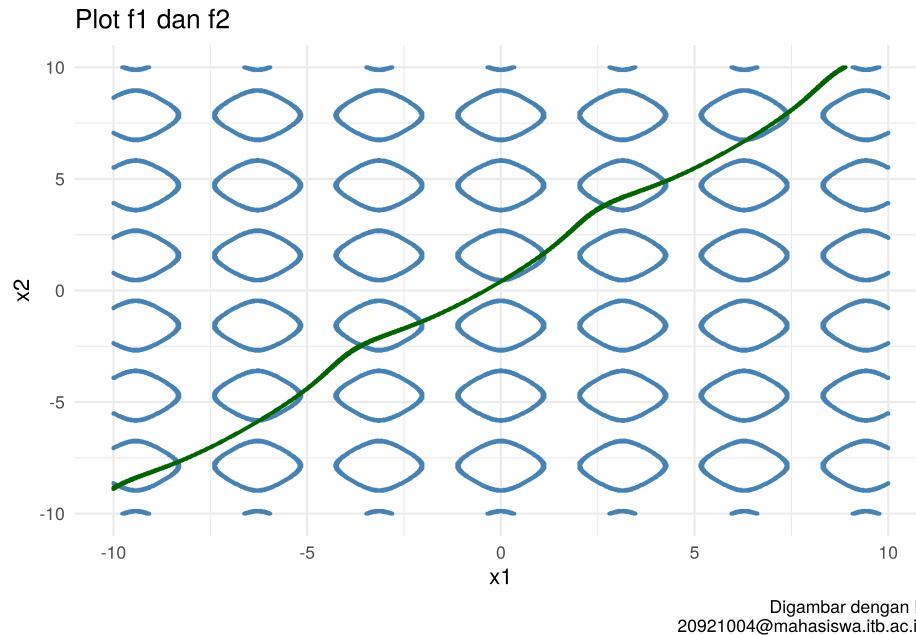


Figure 3: Plot Soal 1: f1 dan f2

Terlihat bahwa ada beberapa titik solusi (persinggungan antara  $f_1(x_1, x_2)$  dengan  $f_2(x_1, x_2)$ ).

## Mencari Akar Sistem Persamaan

Untuk mencari akarnya kita perlu membentuk  $F(x)$  sebagaimana yang telah dijelaskan pada bagian sebelumnya.

```
# fungsi f1 dan f2 dari soal
f1 = function(x1,x2){cos(2*x1) - cos(2*x2) - 0.4}
f2 = function(x1,x2){2*(x2 - x1) + sin(x2) - sin(x1) - 1.2}

# membuat F(x)
# saya notasikan sebagai f kecil
f = function(x1,x2){
  sum = abs(f1(x1,x2)) + abs(f2(x1,x2))
  bawah = 1 + sum
  hasil = 1/bawah
  return(hasil)
}
```

Sekarang saya akan selesaikan dengan program yang telah dibuat sebelumnya:

```
# solving
N = 50
a = -10
b = 10
rot = 20
k_max = 60
r = .8
soa_mrf(N,a,b,rot,k_max,r)

##           x1           x2           f
## 1  1.15649 -18.6162  0.0225232
```

## SOAL 2

Tentukanlah akar-akar sistem persamaan berikut dengan **SOA**. Buatlah terlebih dahulu *contour plot*-nya:

$$f_1(x_1, x_2) = \sin(x_1) \cos(x_2) + 2 \cos(x_1) \sin(x_2) = 0$$

$$f_2(x_1, x_2) = \cos(x_1) \sin(x_2) + 2 \sin(x_1) \cos(x_2) = 0$$

dengan  $0 \leq x_1, x_2 \leq 2\pi$

### Contour Plot

Pertama-tama, saya akan buat *contour plot* dari  $f_1(x_1, x_2)$  sebagai berikut:

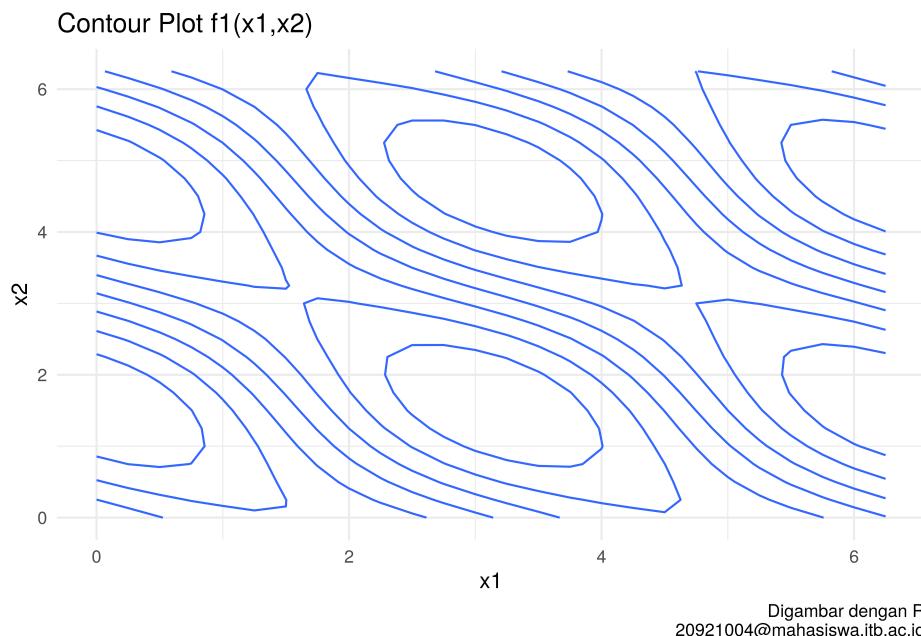


Figure 4: Contour Plot Soal 2:  $f_1$

Berikutnya adalah *contour plot* dari  $f_2(x_1, x_2)$  sebagai berikut:

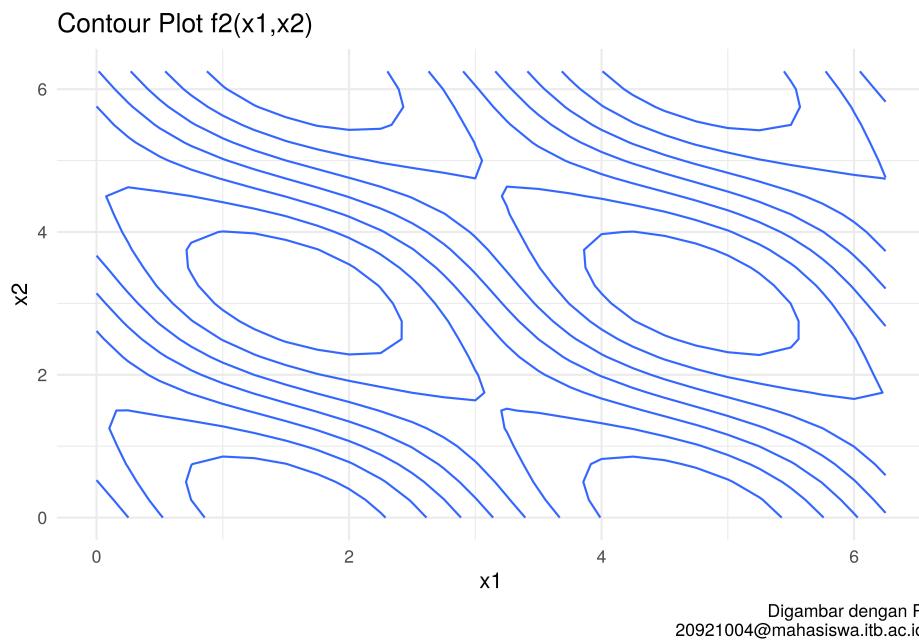


Figure 5: Contour Plot Soal 2:  $f_2$

### Grafik Sistem Persamaan

Kita akan mencari akar-akar sistem persamaan saat  $f_1 = 0$  dan  $f_2 = 0$  dengan bantuan grafik sebagai berikut:

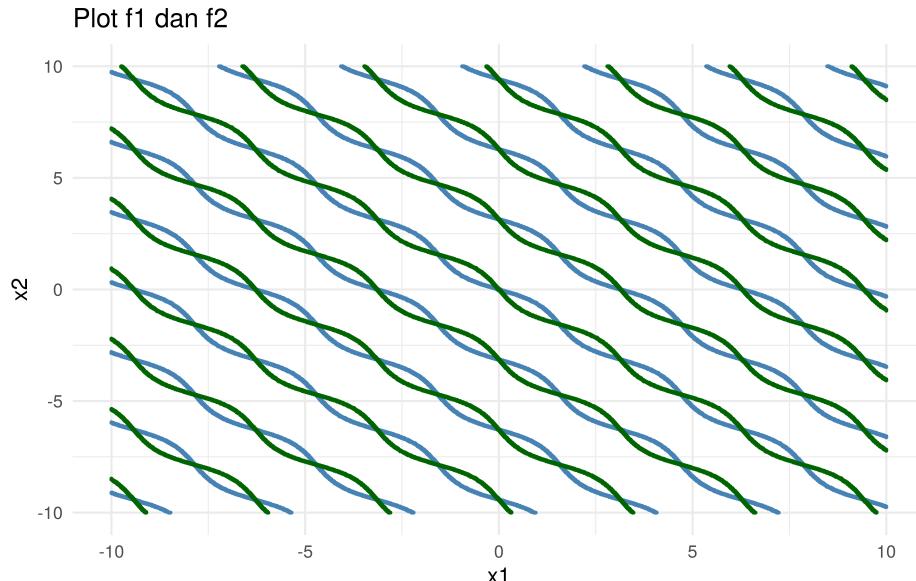


Figure 6: Plot Soal 1: f1 dan f2

### SOAL 3

Tentukanlah akar-akar sistem persamaan berikut dengan **SOA**. Buatlah terlebih dahulu *contour plot*-nya:

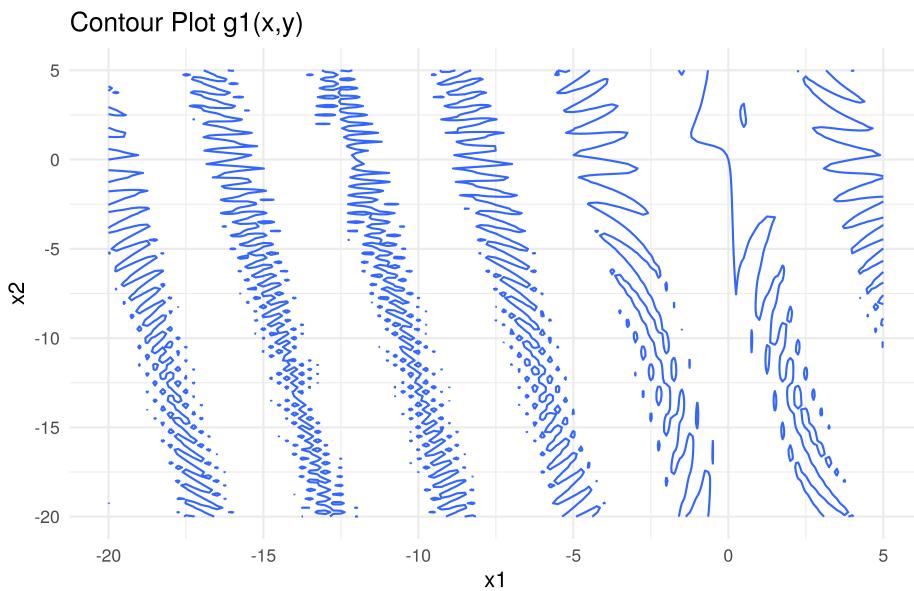
$$g_1(x, y) = 0.5 \sin(xy) - 0.25 \frac{y}{\pi} - 0.5x = 0$$

$$g_2(x, y) = \left(1 - \frac{0.25}{\pi}\right)(e^{2x} - e) + e \frac{y}{\pi} - 2ex = 0$$

dengan  $-1 \leq x \leq 3, -20 \leq y \leq 5$

#### *Contour Plot*

Pertama-tama, saya akan buat *contour plot* dari  $g_1(x, y)$  sebagai berikut:



Digambar dengan R  
20921004@mahasiswa.itb.ac.id

Figure 7: Contour Plot Soal 3:  $g_1$

Berikutnya adalah *contour plot* dari  $g_2(x, y)$  sebagai berikut:

### Grafik Sistem Persamaan

Kita akan mencari akar-akar sistem persamaan saat  $f_1 = 0$  dan  $f_2 = 0$  dengan bantuan grafik sebagai berikut:

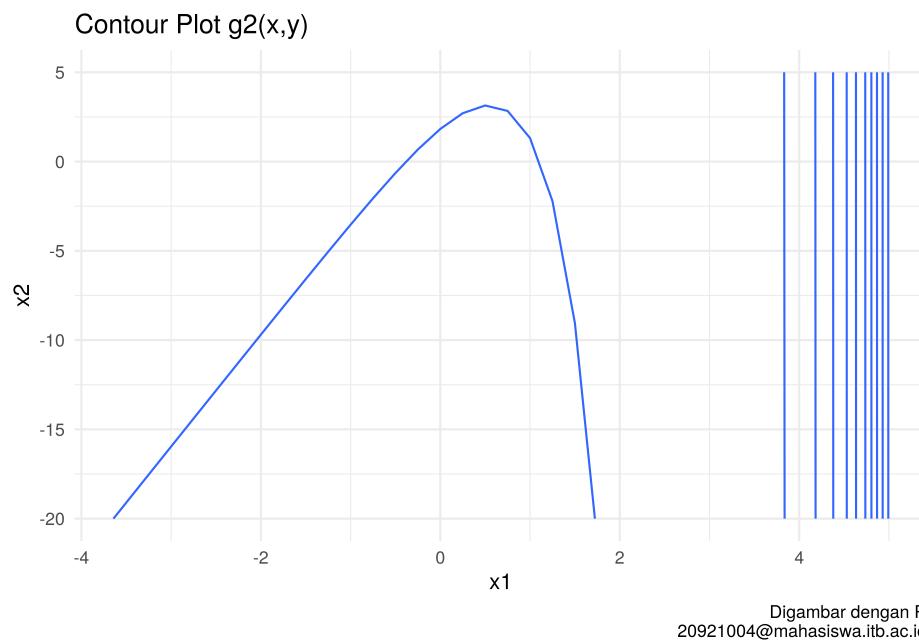


Figure 8: Contour Plot Soal 3: g2

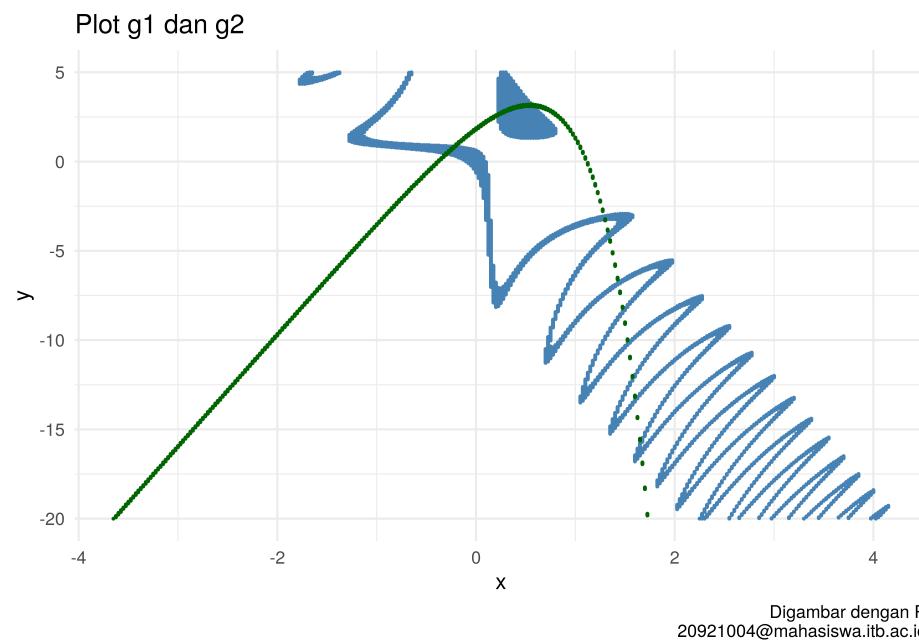


Figure 9: Plot Soal 1: f1 dan f2