

# 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)
}

```

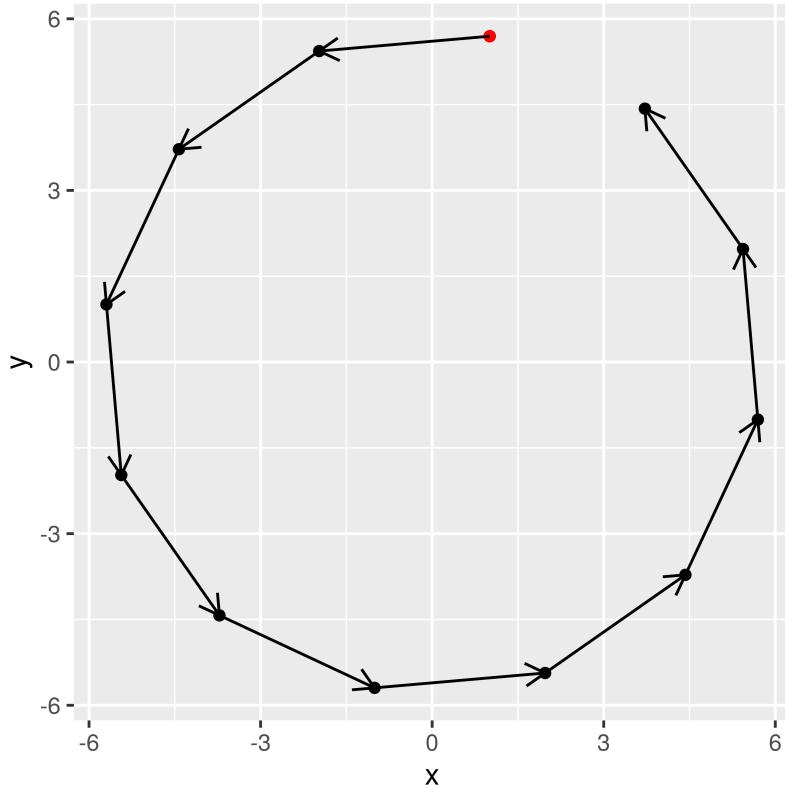
Berikut adalah uji coba dengan titik sembarang berikut ini:

```
# 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  1.00691  5.69547  
## 2 -1.97573  5.43588  
## 3 -4.42897  3.71974  
## 4 -5.69547  1.00691  
## 5 -5.43588 -1.97573  
## 6 -3.71974 -4.42897  
## 7 -1.00691 -5.69547  
## 8  1.97573 -5.43588  
## 9  4.42897 -3.71974  
## 10 5.69547 -1.00691  
## 11 5.43588  1.97573  
## 12 3.71974  4.42897
```

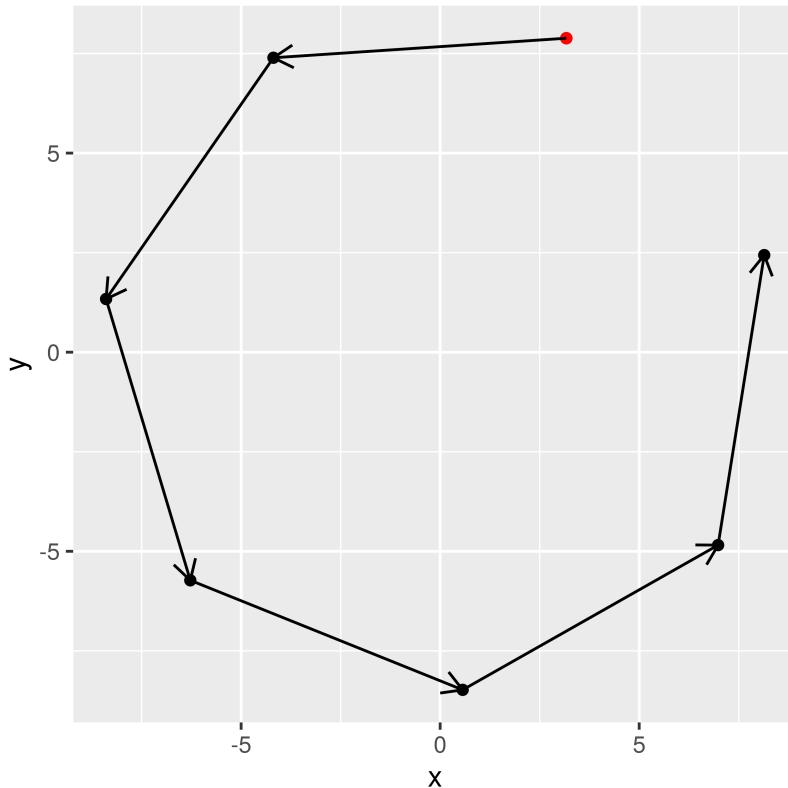
Uji coba kembali dengan titik sembarang lainnya berikut ini:

```
# 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  3.169548  7.88497  
## 2 -4.188541  7.39425  
## 3 -8.392573  1.33551  
## 4 -6.276826 -5.72890  
## 5  0.565499 -8.47933  
## 6  6.981991 -4.84465  
## 7  8.140902  2.43815
```

## 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)
}

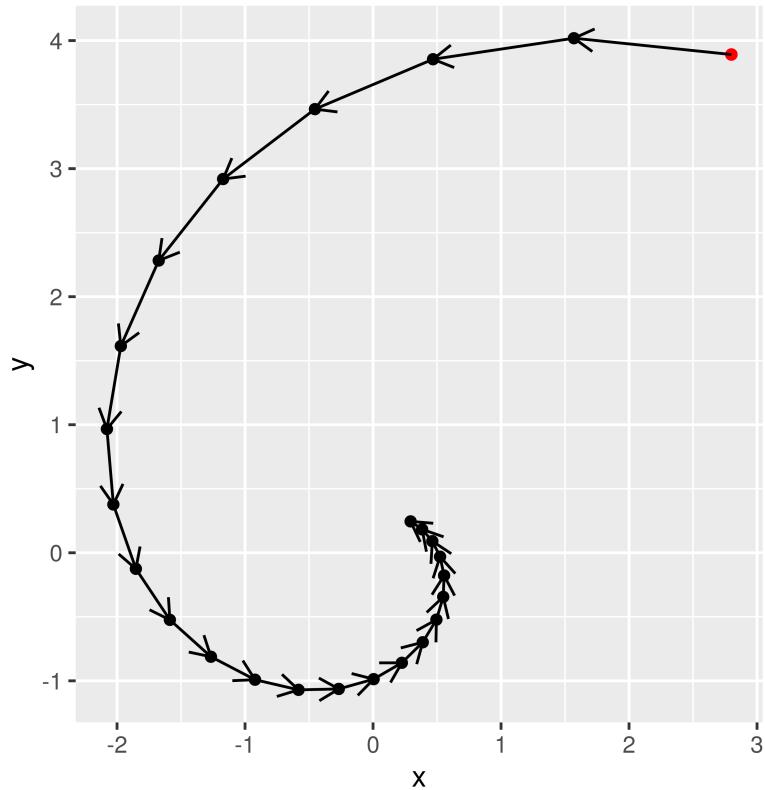
```

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



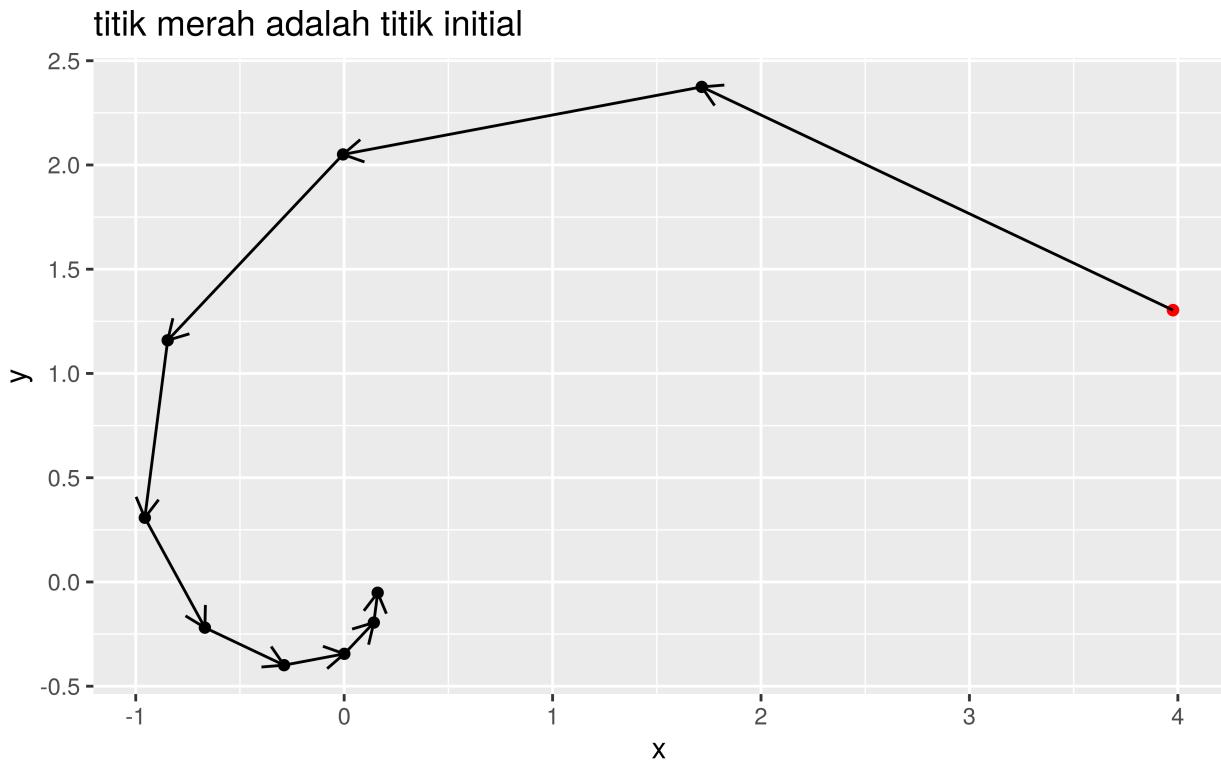
```
##
## $`Titik-titik rotasi`  
##          x      y  
## 1   2.79956525  3.8911263  
## 2   1.56953520  4.0185927  
## 3   0.46875780  3.8544019  
## 4  -0.45406786  3.4648957  
## 5  -1.17133830  2.9188057  
## 6  -1.67437442  2.2822255  
## 7  -1.97040354  1.6147127  
## 8  -2.07905617  0.9665677  
## 9  -2.02870296  0.3772448  
## 10 -1.85290605 -0.1252124  
## 11 -1.58719908 -0.5238699  
## 12 -1.26634785 -0.8119187  
## 13 -0.92218251 -0.9912048  
## 14 -0.58203705 -1.0704616
```

```
## 15 -0.26778448 -1.0634200
## 16  0.00458124 -0.9869454
## 17  0.22489259 -0.8593195
## 18  0.38837811 -0.6987545
## 19  0.49495470 -0.5221945
## 20  0.54834235 -0.3444287
## 21  0.55509399 -0.1775166
## 22  0.52362114 -0.0305040
## 23  0.46328099  0.0906063
## 24  0.38357397  0.1826757
## 25  0.29348432  0.2450948
```

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-titik rotasi`  
##           x         y  
## 1   3.976523858  1.3037209  
## 2   1.715537234  2.3744521  
## 3   -0.005438403 2.0505377  
## 4   -0.846772909 1.1590063  
## 5   -0.956410326  0.3079546  
## 6   -0.668334365 -0.2191164  
## 7   -0.288330343 -0.3990742  
## 8   0.000914032 -0.3446339  
## 9   0.142317123 -0.1947942  
## 10  0.160743883 -0.0517579
```

#### 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)
}

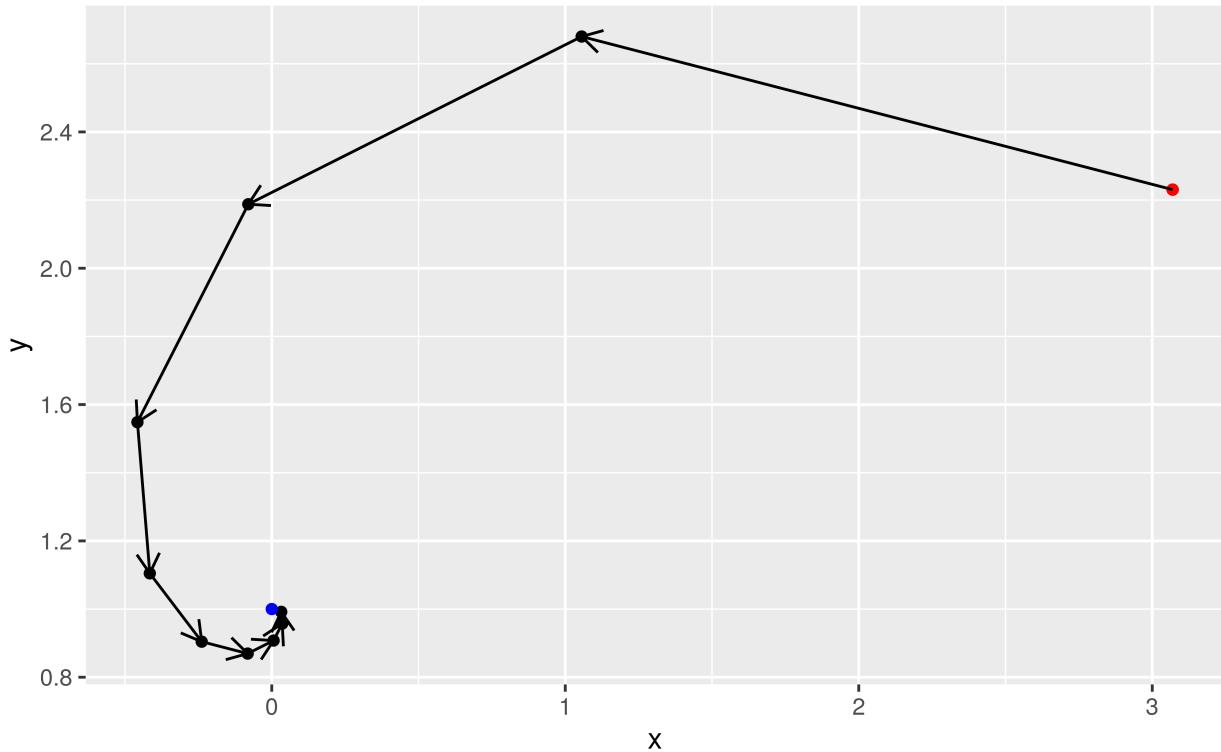
```

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



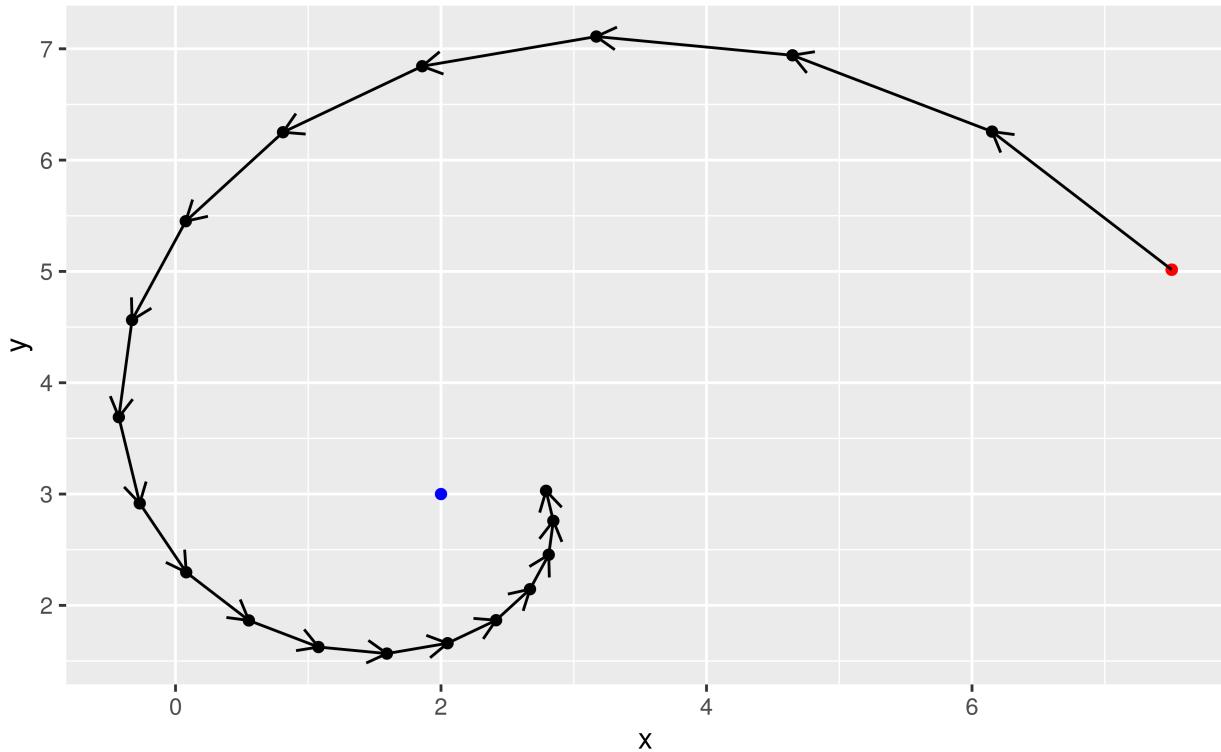
```
##
## $`Titik-titik rotasi`  
##           x      y
## 1   3.06937459 2.230669
## 2   1.05588425 2.679859
## 3   -0.07990089 2.187801
## 4   -0.45768774 1.548392
## 5   -0.41556827 1.104782
## 6   -0.23867457 0.904303
## 7   -0.08210556 0.869374
## 8    0.00621309 0.907637
## 9    0.03558980 0.957357
## 10   0.03231459 0.991852
```

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   7.5043991 5.01550
## 2   6.1509541 6.25603
## 3   4.6474623 6.94144
## 4   3.1699224 7.10998
## 5   1.8583478 6.84331
## 6   0.8098682 6.25029
## 7   0.0773495 5.45110
## 8  -0.3273819 4.56330
## 9  -0.4269022 3.69083
## 10 -0.2694387 2.91636
## 11  0.0807347 2.29724
## 12  0.5526518 1.86469
## 13  1.0768870 1.62570
```

```
## 14 1.5920733 1.56694
## 15 2.0493911 1.65992
## 16 2.4149733 1.86669
## 17 2.6703868 2.14536
## 18 2.8115079 2.45491
## 19 2.8462085 2.75912
## 20 2.7913043 3.02916
```

## 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 -2.83468 -2.89922 -78.2519
```

### 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:

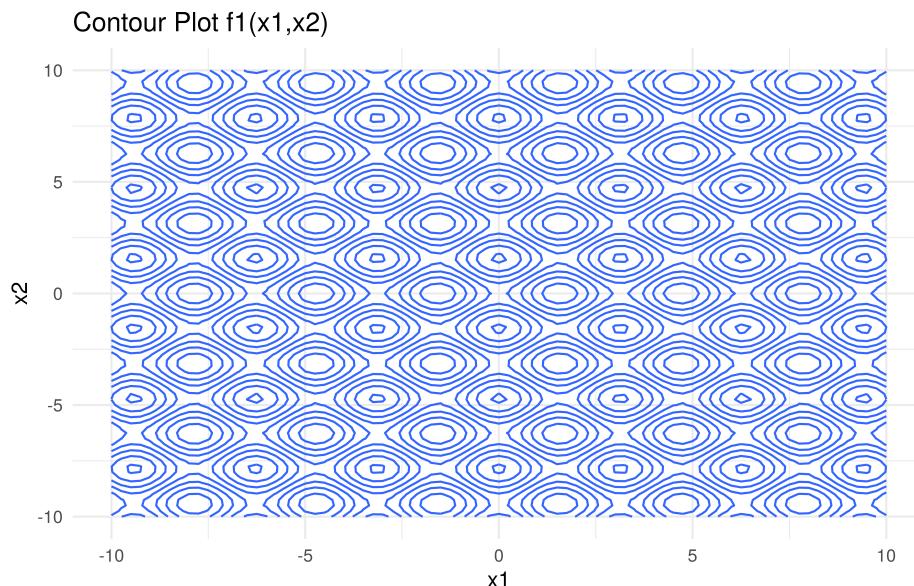


Figure 1: Contour Plot Soal 1: f1

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

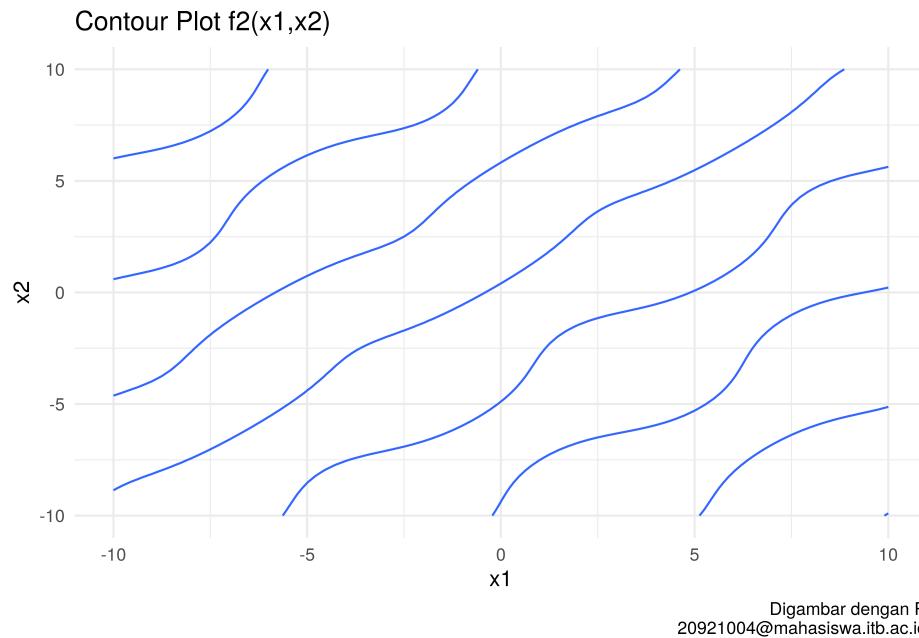


Figure 2: Contour Plot Soal 1:  $f_2$

## Grafik Sistem Persamaan

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

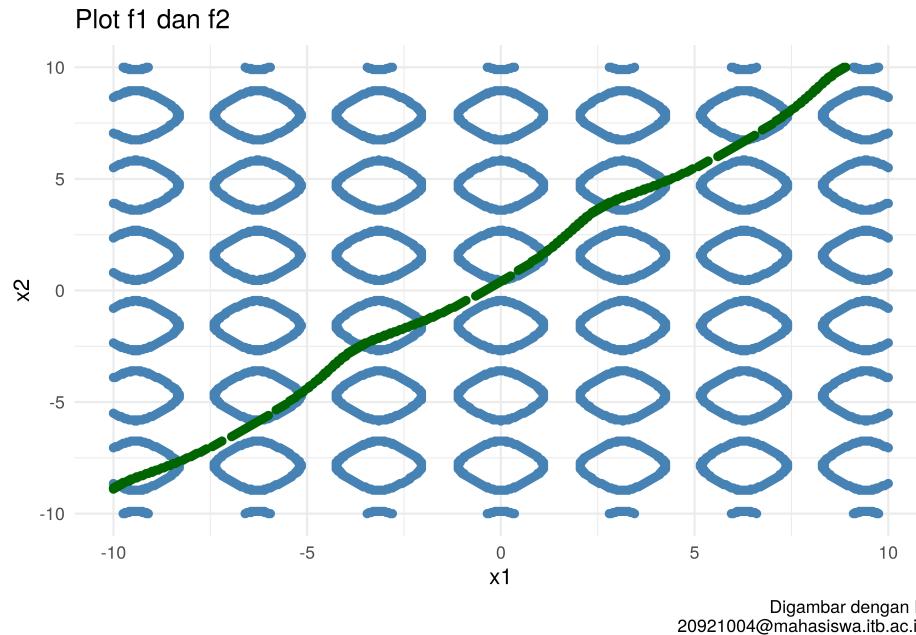


Figure 3: Plot Soal 1: f1 dan f2

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

Untuk mencari akarnya

## 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:

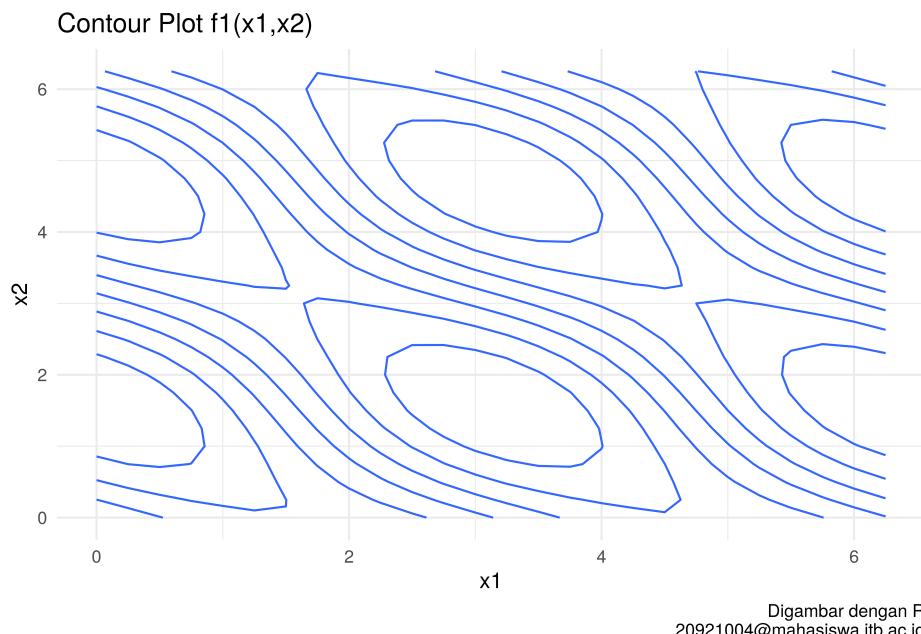


Figure 4: Contour Plot Soal 2:  $f_1$

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

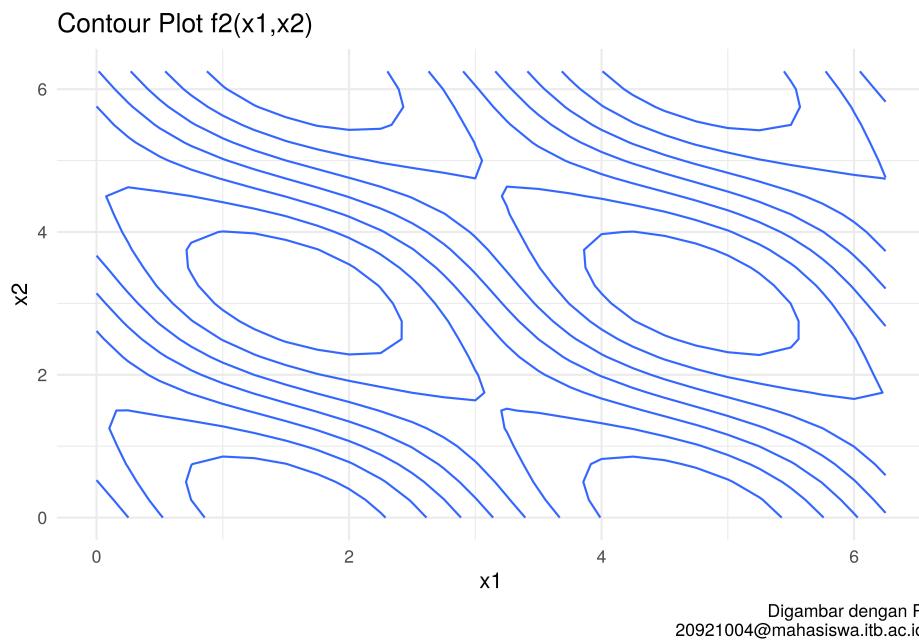


Figure 5: Contour Plot Soal 2:  $f_2$