

# SPIRAL OPTIMIZATION ALGORITHM

Tugas Kuliah  
SK5001 Analisis Numerik Lanjut

Mohammad Rizka Fadhli  
NIM: 20921004

18 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)
}

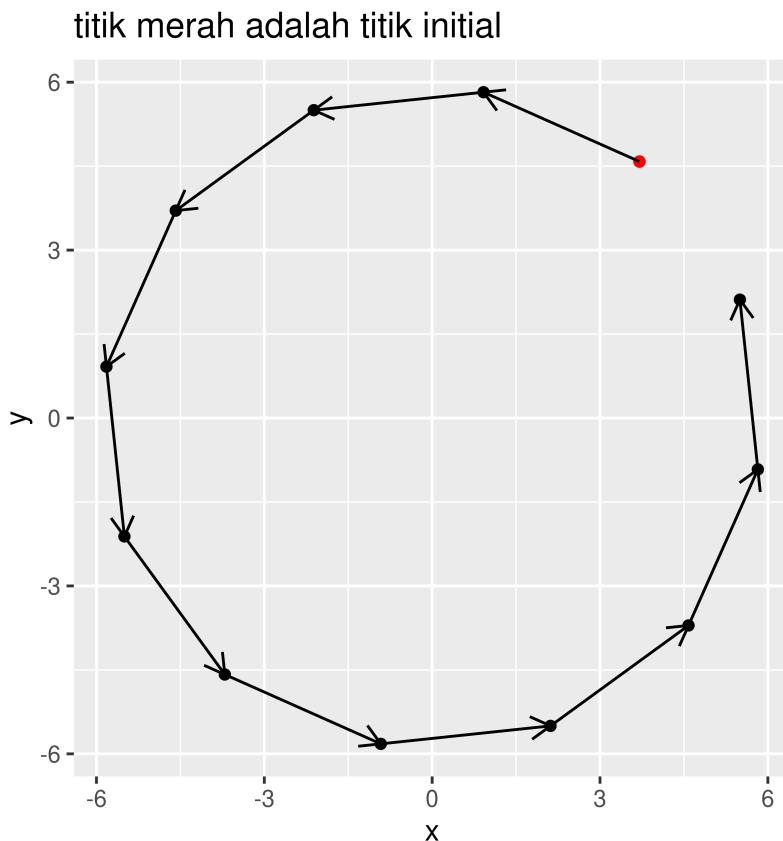
```

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-titik rotasi`  
##      x      y  
## 1  3.705807  4.582290  
## 2  0.918178  5.821282  
## 3 -2.115476  5.500467  
## 4 -4.582290  3.705807  
## 5 -5.821282  0.918178  
## 6 -5.500467 -2.115476  
## 7 -3.705807 -4.582290  
## 8 -0.918178 -5.821282  
## 9  2.115476 -5.500467  
## 10 4.582290 -3.705807  
## 11 5.821282 -0.918178  
## 12 5.500467  2.115476
```

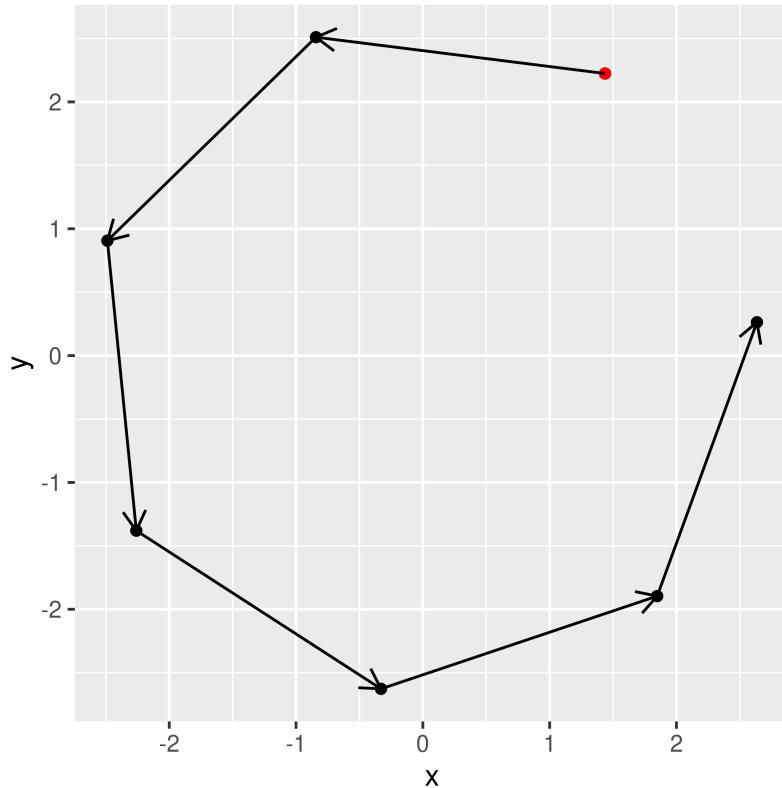
Uji coba kembali dengan titik sembarang lainnya berikut ini:

```
# 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  1.437130  2.223893
## 2 -0.842674  2.510168
## 3 -2.487927  0.906235
## 4 -2.259720 -1.380111
## 5 -0.329898 -2.627205
## 6  1.848344 -1.895961
## 7  2.634745  0.262981
```

## 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)
}

```

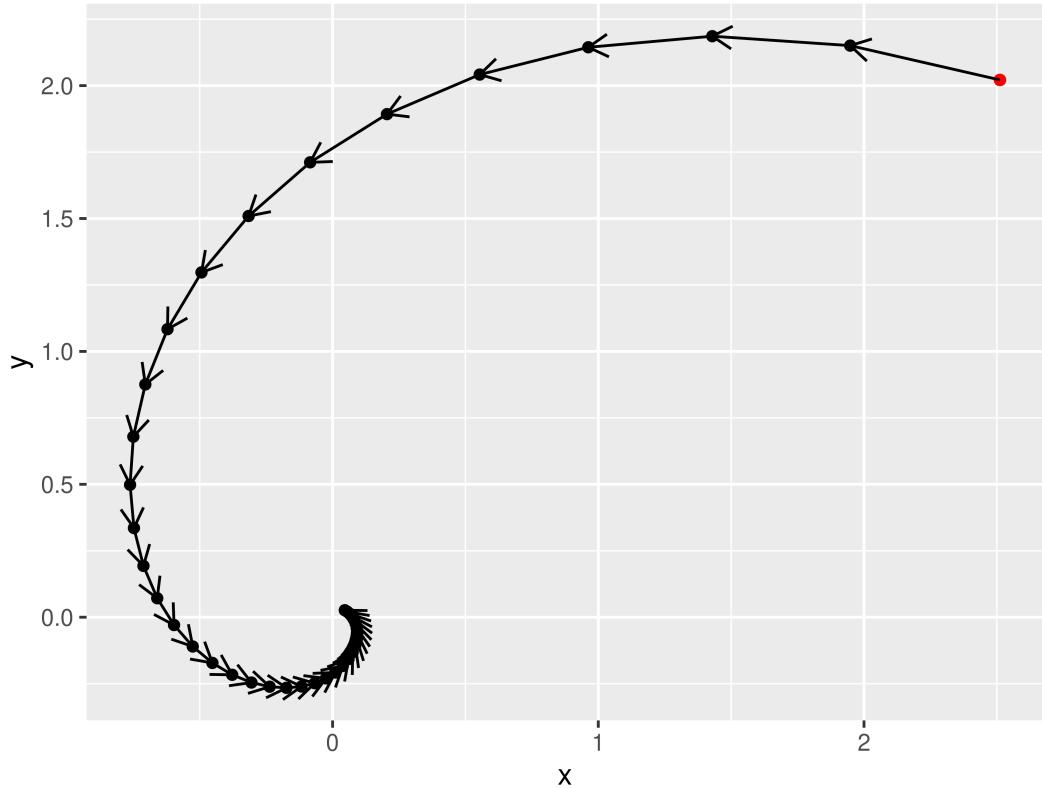
Berikutnya saya akan tunjukkan ilustrasi dari program ini.

Saya akan uji coba untuk sembarang titik berikut ini:

```
# uji coba
rot = 40 # 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  2.5117444  2.02136556  
## 2  1.9481484  2.15046232  
## 3  1.4289814  2.18586971  
## 4  0.9624986  2.14424997  
## 5  0.5536926  2.04157678  
## 6  0.2047525  1.89275238  
## 7  -0.0844740  1.71133182  
## 8  -0.3160308  1.50934306  
## 9  -0.4934279  1.29719019  
## 10 -0.6212504  1.08362745  
## 11 -0.7048066  0.87579111  
## 12 -0.7498198  0.67927736
```

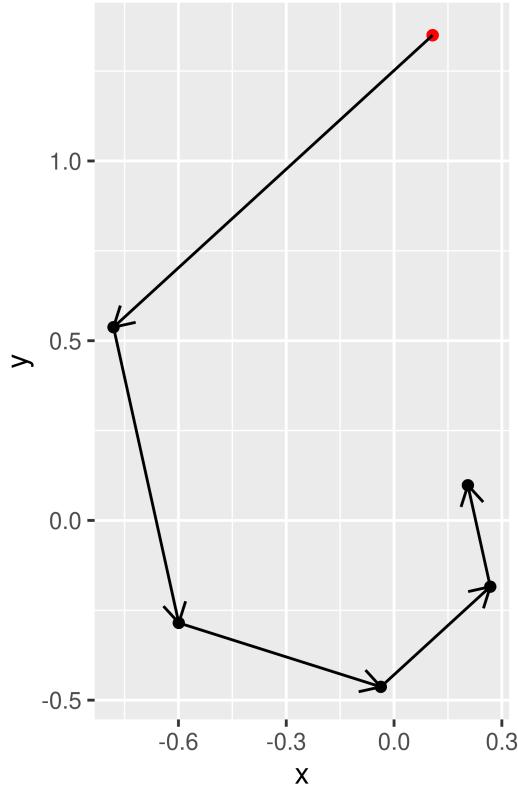
```
## 13 -0.7621656  0.49825500
## 14 -0.7476537  0.33560251
## 15 -0.7118538  0.19306069
## 16 -0.6599619  0.07139279
## 17 -0.5967045 -0.02945427
## 18 -0.5262754 -0.11019311
## 19 -0.4523023 -0.17204766
## 20 -0.3778375 -0.21661661
## 21 -0.3053695 -0.24575086
## 22 -0.2368494 -0.26144601
## 23 -0.1737308 -0.26575073
## 24 -0.1170174 -0.26069074
## 25 -0.0673161 -0.24820808
## 26 -0.0248931 -0.23011450
## 27  0.0102701 -0.20805800
## 28  0.0384220 -0.18350088
## 29  0.0599893 -0.15770804
## 30  0.0755295 -0.13174380
## 31  0.0856880 -0.10647575
## 32  0.0911606 -0.08258427
## 33  0.0926615 -0.06057618
## 34  0.0908972 -0.04080143
## 35  0.0865448 -0.02347167
## 36  0.0802360 -0.00867970
## 37  0.0725453  0.00358095
## 38  0.0639828  0.01339691
## 39  0.0549894  0.02091698
## 40  0.0459362  0.02633552
```

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

```
# uji coba
rot = 6 # 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  0.1073273 1.3498148  
## 2 -0.7807172  0.5374989  
## 3 -0.5990924 -0.2851600  
## 4 -0.0368133 -0.4629865  
## 5  0.2677860 -0.1843621  
## 6  0.2054887  0.0978099
```

#### 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 konstraksi
  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)
}

```

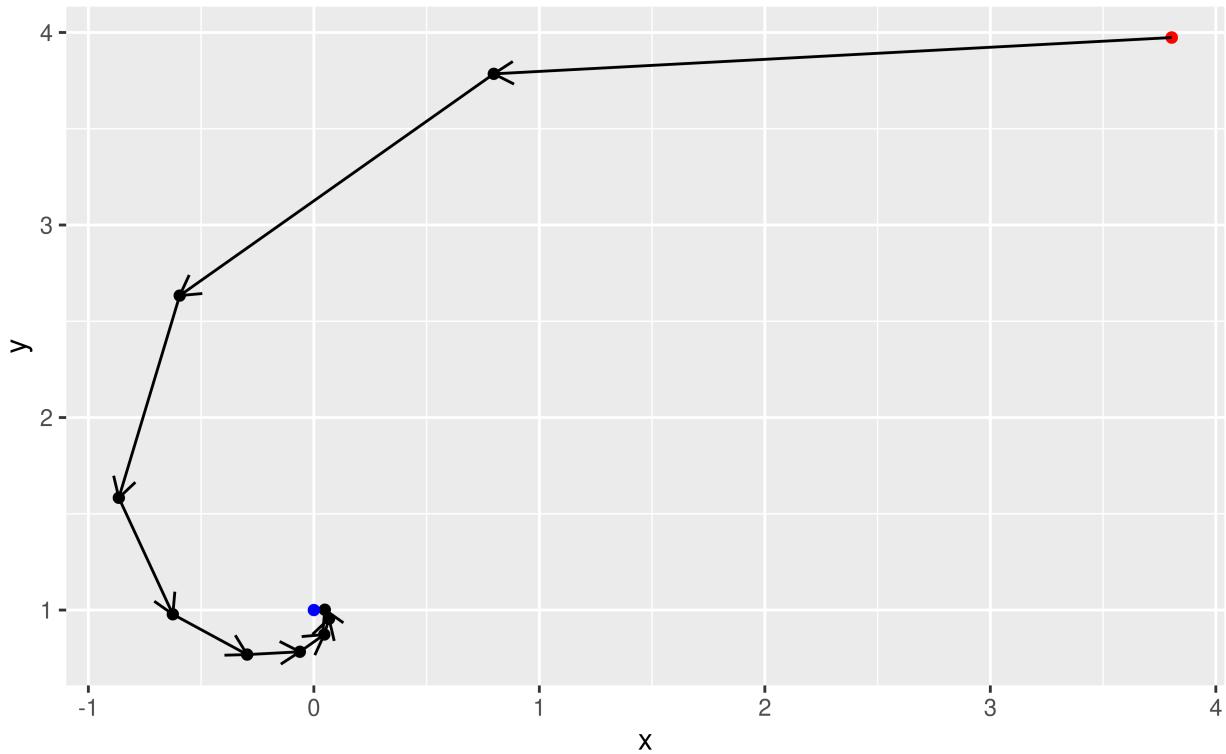
Berikutnya saya akan tunjukkan ilustrasi dari program ini.

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



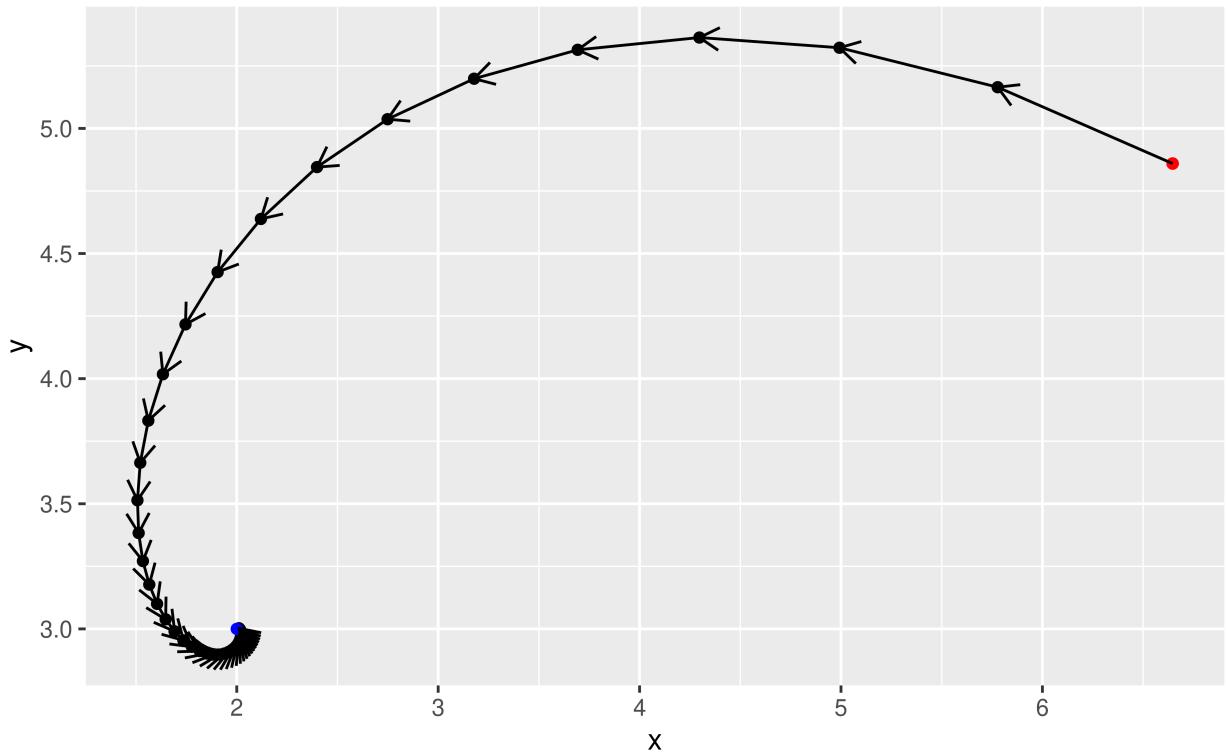
```
##
## $`Titik-titik rotasi`#
##      x      y
## 1  3.8044444 3.973729
## 2  0.7979677 3.785196
## 3 -0.5949167 2.633383
## 4 -0.8648256 1.583051
## 5 -0.6254203 0.978020
## 6 -0.2958336 0.768763
## 7 -0.0620500 0.783423
## 8  0.0462607 0.872988
## 9  0.0672488 0.954662
## 10 0.0486327 1.001709
```

Saya akan coba kembali dengan sembarang titik lainnya:

```
# uji coba
rot = 45 # berapa banyak rotasi
x0 = rand_titik(0,10) # generate random titik
x_bintang = c(2,3) # contoh pusat rotasi
r = .87
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  6.64643 4.85957
## 2  5.77790 5.16468
## 3  4.99268 5.32237
## 4  4.29710 5.36316
## 5  3.69290 5.31407
## 6  3.17830 5.19863
## 7  2.74893 5.03686
## 8  2.39861 4.84550
## 9  2.11996 4.63823
## 10 1.90499 4.42591
## 11 1.74550 4.21696
```

```
## 12 1.63339 4.01764
## 13 1.56093 3.83234
## 14 1.52095 3.66393
## 15 1.50689 3.51399
## 16 1.51294 3.38311
## 17 1.53399 3.27109
## 18 1.56569 3.17713
## 19 1.60438 3.10002
## 20 1.64705 3.03827
## 21 1.69129 2.99023
## 22 1.73522 2.95421
## 23 1.77743 2.92849
## 24 1.81691 2.91144
## 25 1.85298 2.90153
## 26 1.88526 2.89737
## 27 1.91357 2.89769
## 28 1.93793 2.90139
## 29 1.95846 2.90753
## 30 1.97541 2.91530
## 31 1.98907 2.92405
## 32 1.99978 2.93325
## 33 2.00789 2.94246
## 34 2.01377 2.95139
## 35 2.01775 2.95978
## 36 2.02016 2.96750
## 37 2.02130 2.97444
## 38 2.02145 2.98056
## 39 2.02083 2.98585
## 40 2.01966 2.99033
## 41 2.01811 2.99405
## 42 2.01632 2.99707
## 43 2.01442 2.99945
## 44 2.01249 3.00127
## 45 2.01060 3.00261
```

## 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
                    x1_d,    # batas bawah x1
                    x1_u,    # batas atas x1
                    x2_d,    # batas bawah x2
                    x2_u,    # batas atas x2
                    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,x1_d,x1_u)
  x2 = runif(N,x2_d,x2_u)

  # 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 # x1 dan x2 punya batas bawah yang sama
b = 4 # x1 dan x2 punya batas atas yang sama
k_max = 70
r = .75
rot = 30
f = function(x1,x2){
  ((x1^4 - 16 * x1^2 + 5 * x1)/2) + ((x2^4 - 16 * x2^2 + 5* x2)/2)
}

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

##           x1          x2          f
## 1 -2.92184 -3.01932 -78.0856

```

### 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. **Akibatnya algoritma yang sebelumnya adalah mencari  $\min F(x)$  diubah menjadi  $\max F(x)$ .** Kenapa demikian?

Karena jika  $F(x) = 1$  artinya  $\sum_{i=1}^n |g_i(x)| = 0$  yang merupakan akar dari  $g_i, i = 1, 2, \dots, n$ .

**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:

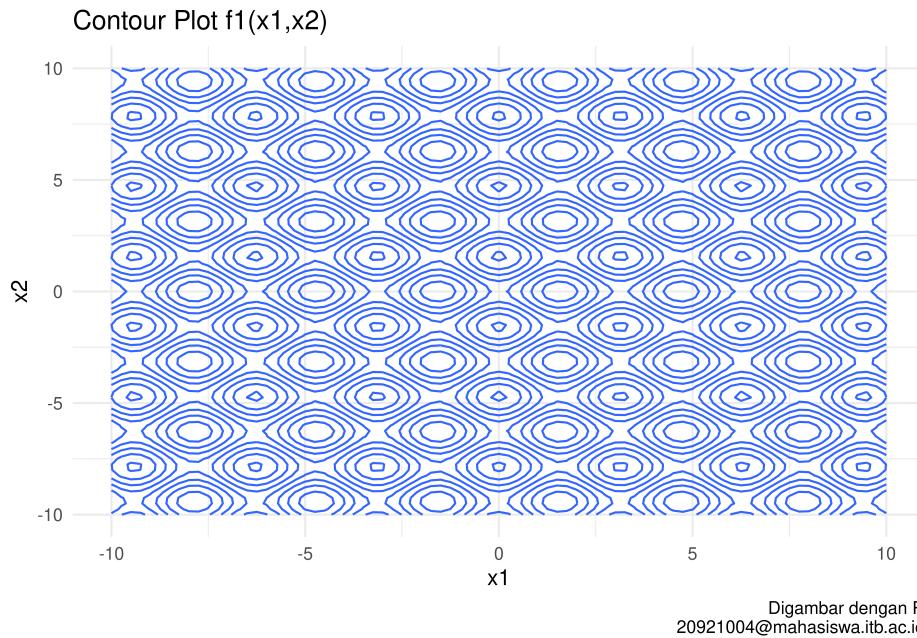


Figure 1: Contour Plot Soal 1: f1

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

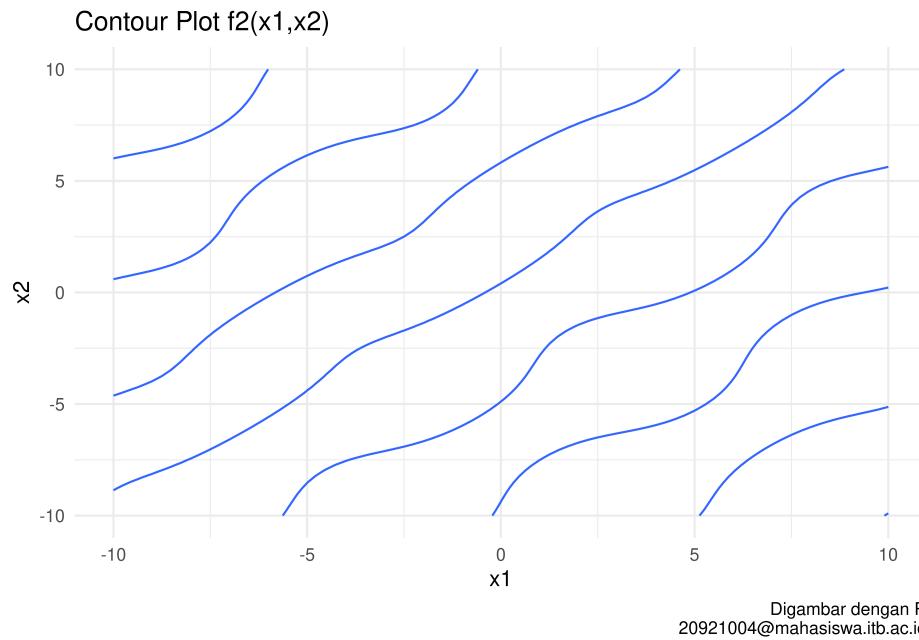


Figure 2: Contour Plot Soal 1:  $f_2$

## Grafik Sistem Persamaan

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

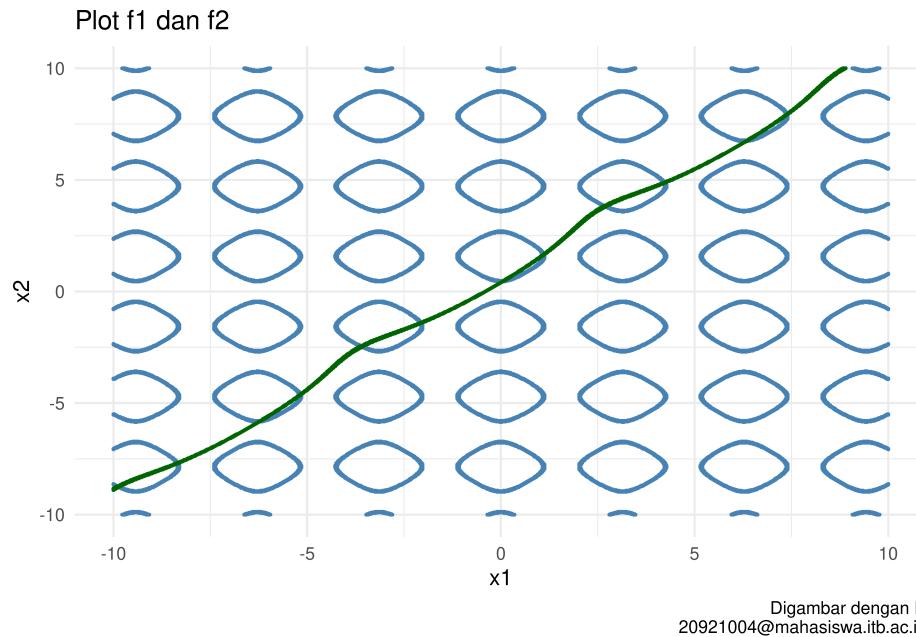


Figure 3: Plot Soal 1: f1 dan f2

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)
}
```

Oleh karena solusi dari grafik ada banyak, maka kita akan *run* program yang telah dibuat sebelumnya berulang kali:

```
# solving
N = 50
a = -10 # x1 dan x2 punya batas yang sama
b = 10 # x1 dan x2 punya batas yang sama
rot = 20
k_max = 60
r = .65
# run I
soa_mrf_2(N,a,b,a,b,rot,k_max,r)
```

```
##          x1      x2      f
## 1 0.0629407 0.46858 0.999991
```

```
# run II
soa_mrf_2(N,a,b,a,b,rot,k_max,r)
```

```
##          x1      x2 f
## 1 1.09995 1.64699 1
```

```
# run III
soa_mrf_2(N,a,b,a,b,rot,k_max,r)
```

```
##          x1      x2      f
## 1 -5.18231 -4.63439 0.99676
```

Berikutnya saya coba *run* sebanyak **100 kali**, berikut adalah rekap semua akar yang saya dapatkan:

```
##      x1      x2      f
## 1 -8.487 -7.723 0.71
## 2 -8.444 -7.834 0.73
## 3 -8.337 -7.687 0.98
## 4 -6.299 -5.898 0.89
## 5 -6.260 -5.840 0.93
## 6 -6.239 -5.835 0.97
## 7 -6.226 -5.821 0.99
## 8 -6.220 -5.815 1.00
## 9 -5.245 -4.721 0.89
## 10 -5.229 -4.697 0.91
## 11 -5.200 -4.719 0.86
## 12 -5.186 -4.645 0.98
## 13 -5.183 -4.648 0.97
## 14 -5.183 -4.636 1.00
## 15 -5.183 -4.635 1.00
## 16 -5.181 -4.640 0.99
## 17 -5.177 -4.671 0.92
## 18 -3.650 -2.490 0.86
## 19 -3.605 -2.472 0.97
## 20 -3.598 -2.463 1.00
## 21 -3.597 -2.463 1.00
## 22 -3.545 -2.474 0.89
## 23 -3.516 -2.574 0.76
## 24 -2.081 -1.420 0.97
## 25 -2.068 -1.408 0.98
## 26 -2.067 -1.406 0.98
## 27 -2.065 -1.412 1.00
## 28 -2.063 -1.408 0.99
## 29 -0.025  0.377 0.88
## 30  0.058  0.471 0.97
## 31  0.060  0.465 1.00
## 32  0.062  0.467 1.00
## 33  0.062  0.468 1.00
## 34  0.063  0.468 1.00
## 35  0.063  0.469 1.00
## 36  0.072  0.480 0.98
## 37  0.091  0.498 0.96
## 38  0.155  0.565 0.89
## 39  1.093  1.633 0.98
## 40  1.096  1.641 0.99
## 41  1.098  1.654 0.98
## 42  1.100  1.647 1.00
## 43  1.101  1.648 1.00
## 44  1.109  1.619 0.92
## 45  1.118  1.651 0.94
## 46  1.160  1.725 0.88
## 47  2.680  3.848 0.92
## 48  2.686  3.820 1.00
## 49  2.687  3.819 1.00
## 50  2.696  3.847 0.92
## 51  2.792  3.815 0.79
```

```
## 52 4.211 4.888 0.96
## 53 4.215 4.869 0.99
## 54 4.217 4.872 1.00
## 55 4.218 4.872 1.00
## 56 4.241 4.876 0.94
## 57 4.247 4.900 0.92
## 58 6.345 6.750 1.00
## 59 6.346 6.752 1.00
## 60 6.351 6.760 0.98
## 61 6.358 6.765 0.98
## 62 6.368 6.774 0.97
## 63 6.451 6.861 0.88
## 64 6.500 6.913 0.83
## 65 7.383 7.930 1.00
## 66 9.003 10.141 0.88
```

## 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:

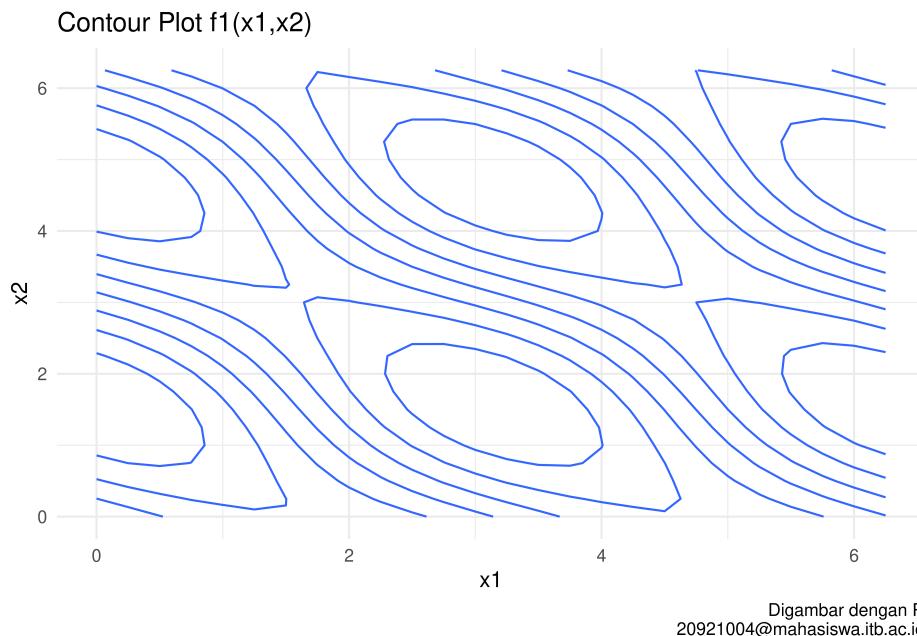


Figure 4: Contour Plot Soal 2: f1

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

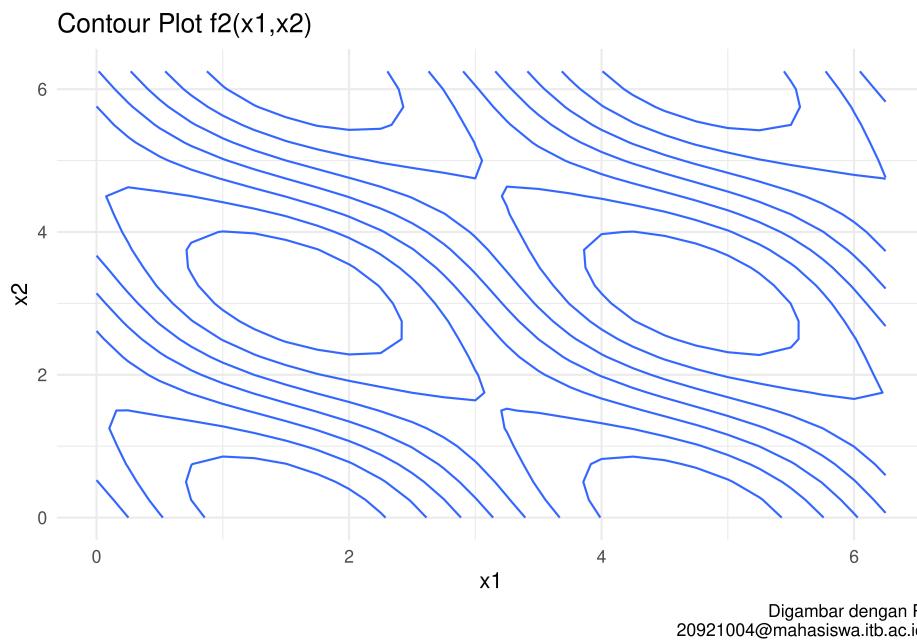


Figure 5: Contour Plot Soal 2:  $f_2$

## Grafik Sistem Persamaan

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

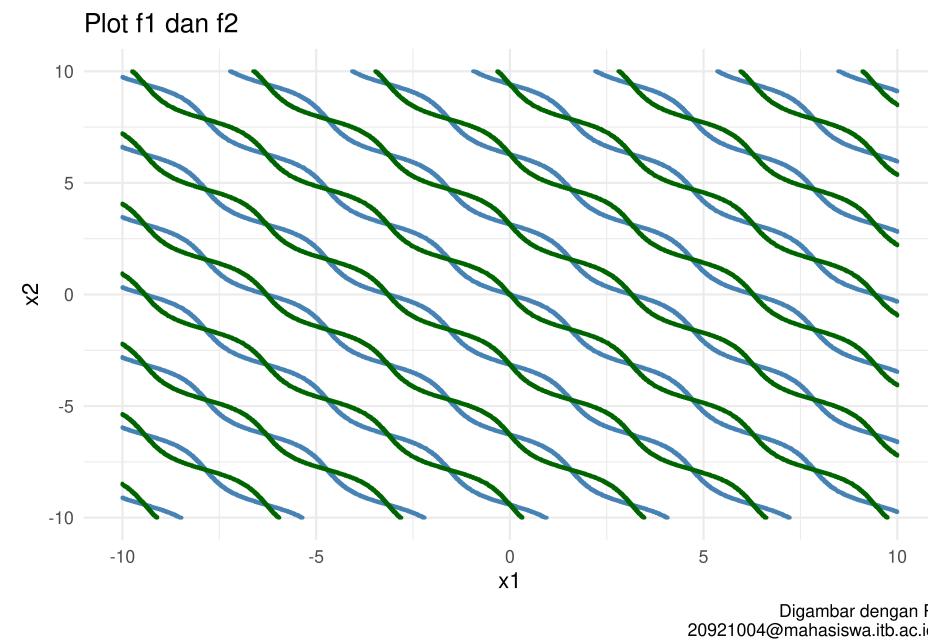


Figure 6: Plot Soal 2: f1 dan f2

## 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){sin(x1)*cos(x2) + 2*cos(x1)*sin(x2)}
f2 = function(x1,x2){cos(x1)*sin(x2) + 2*sin(x1)*cos(x2)}

# 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)
}
```

Oleh karena solusi dari grafik ada banyak, maka kita akan *run* program yang telah dibuat sebelumnya berulang kali:

```
# solving
N = 50
a = 0      # x1 dan x2 punya batas yang sama
b = 2*pi   # x1 dan x2 punya batas yang sama
rot = 30
k_max = 80
r = .75
# run I
soa_mrf_2(N,a,b,a,b,rot,k_max,r)
```

```
##           x1           x2           f
## 1 3.12953 3.14584 0.977066
```

```
# run II
soa_mrf_2(N,a,b,a,b,rot,k_max,r)
```

```
##           x1           x2           f
## 1 4.71124 4.71276 0.997668
```

```
# run III
soa_mrf_2(N,a,b,a,b,rot,k_max,r)
```

```
##           x1           x2           f
## 1 4.77274 1.54123 0.915463
```

Berikutnya saya coba *run* sebanyak **100 kali**, berikut adalah rekap semua akar yang saya dapatkan:

```
##      x1      x2      f
## 1 -0.129  3.208  0.84
## 2 -0.089  0.040  0.87
## 3 -0.045  6.378  0.87
## 4 -0.027  6.334  0.93
## 5  0.000  3.142  1.00
## 6  1.499  4.820  0.85
## 7  1.525  1.592  0.93
## 8  1.539  1.580  0.94
## 9  1.545  4.735  0.95
## 10 1.547  4.717  0.95
## 11 1.550  4.723  0.97
## 12 1.554  4.743  0.96
## 13 1.561  1.564  0.95
## 14 1.570  1.571  1.00
## 15 1.570  4.712  1.00
## 16 1.571  1.571  1.00
## 17 1.571  4.712  1.00
## 18 1.572  4.711  1.00
## 19 1.572  4.712  1.00
## 20 1.573  1.570  1.00
## 21 1.575  1.565  0.99
## 22 1.576  4.752  0.88
## 23 1.725  4.637  0.81
## 24 3.034  3.195  0.86
## 25 3.038  0.013  0.79
## 26 3.090  0.020  0.91
## 27 3.137  3.151  0.99
## 28 3.142  3.141  1.00
## 29 3.142  3.142  1.00
## 30 3.142  6.283  1.00
## 31 3.143  3.140  1.00
## 32 3.143  3.141  1.00
## 33 3.144  3.140  1.00
## 34 3.149  3.141  0.98
## 35 3.151  3.123  0.97
## 36 3.151  3.141  0.97
## 37 3.156  6.295  0.93
## 38 3.179  6.289  0.88
## 39 3.372  6.167  0.74
## 40 4.703  4.717  0.99
## 41 4.709  4.714  1.00
## 42 4.711  4.713  1.00
## 43 4.712  1.571  1.00
## 44 4.712  4.712  1.00
## 45 4.716  1.570  0.99
## 46 4.717  1.569  0.99
## 47 4.720  1.576  0.96
## 48 4.732  1.561  0.97
## 49 4.747  1.502  0.91
## 50 4.748  4.701  0.93
## 51 4.778  1.473  0.86
```

```
## 52 6.289 3.143 0.98  
## 53 6.304 3.136 0.96  
## 54 6.435 3.059 0.81
```

## 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:

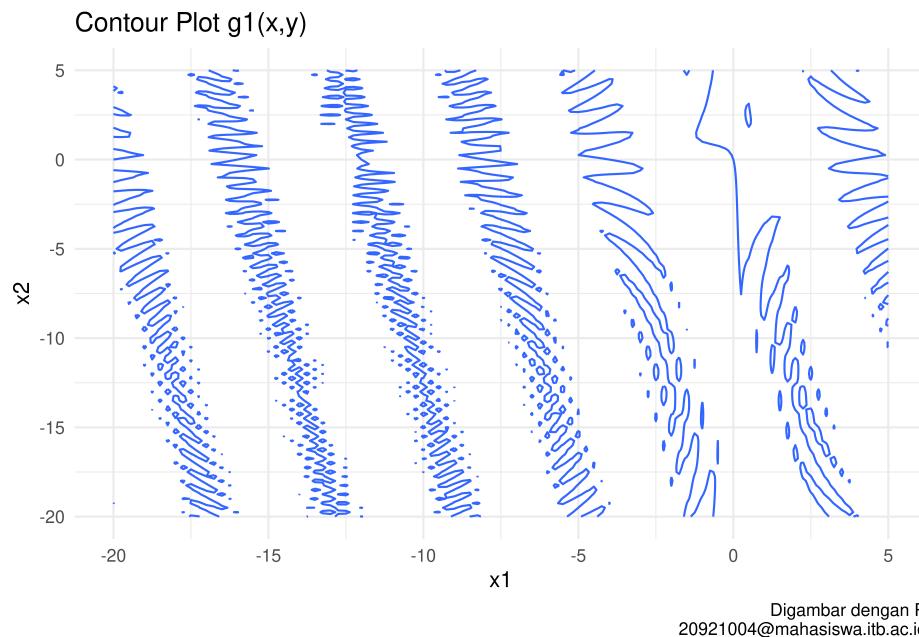


Figure 7: Contour Plot Soal 3: g1

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

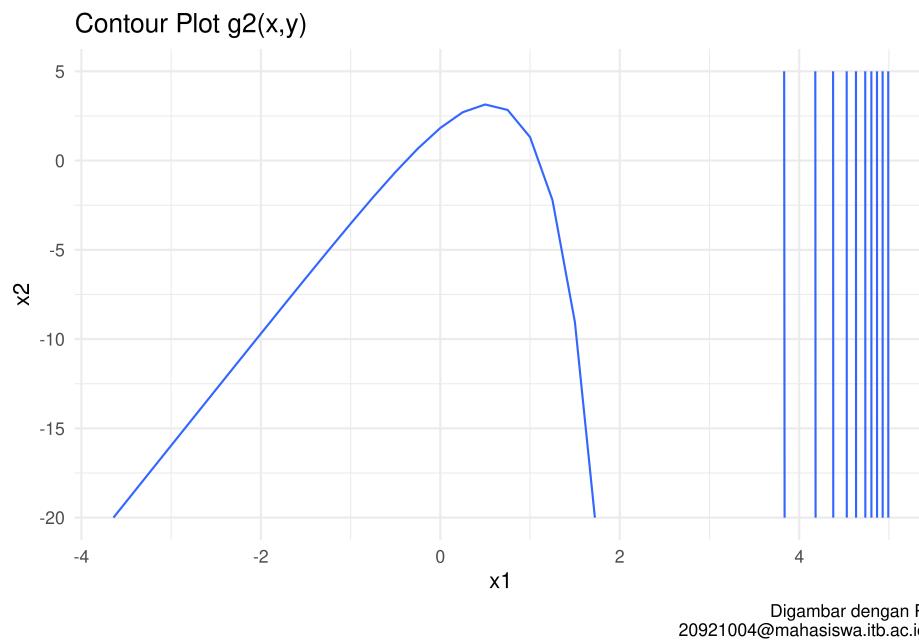


Figure 8: Contour Plot Soal 3: g2

## Grafik Sistem Persamaan

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

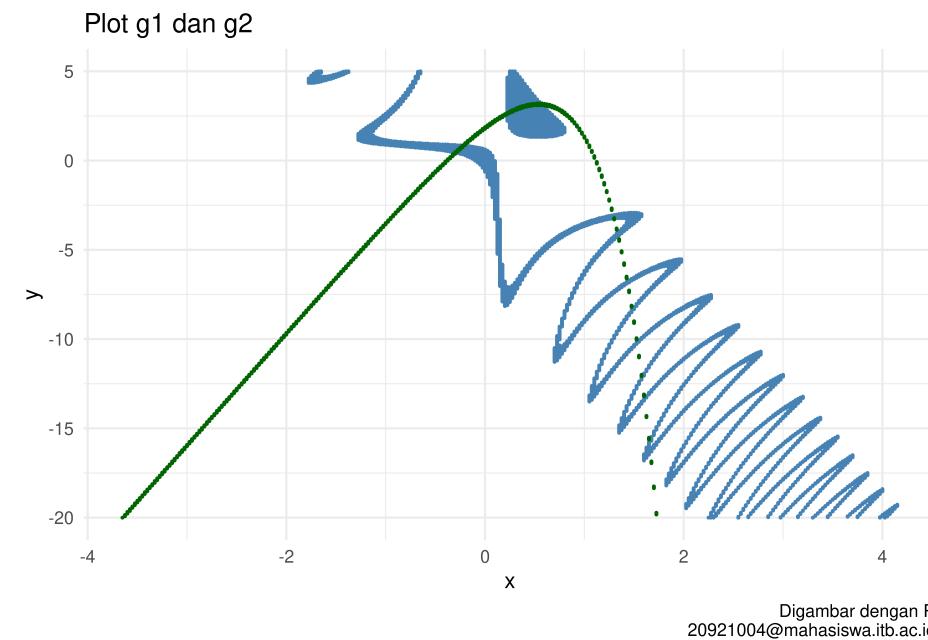


Figure 9: Plot Soal 3: g1 dan g2

## Mencari Akar Sistem Persamaan

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

```
# fungsi g1 dan g2 dari soal
g1 = function(x,y){0.5*sin(x*y) - 0.25*(y)/(pi) - 0.5*x}
g2 = function(x,y){(1 - .25/pi)*(exp(2*x)-exp(1)) + exp(1)*(y/pi) - 2*exp(1)*x}

# membuat F(x)
# saya notasikan sebagai f kecil
f = function(x1,x2){
  sum = abs(g1(x1,x2)) + abs(g2(x1,x2))
  bawah = 1 + sum
  hasil = 1/bawah
  return(hasil)
}
```

Oleh karena solusi dari grafik ada banyak, maka kita akan *run* program yang telah dibuat sebelumnya berulang kali:

```
# solving
N = 60
x1_lower = -1 # batas x1
x1_upper = 3 # batas x1
x2_lower = -20 # batas x2
x2_upper = 5 # batas x2
rot = 30
k_max = 80
r = .75
# run I
soa_mrf_2(N,x1_lower,x1_upper,x2_lower,x2_upper,rot,k_max,r)

##          x1          x2          f
## 1  0.49606  3.13958  0.997879

# run II
soa_mrf_2(N,x1_lower,x1_upper,x2_lower,x2_upper,rot,k_max,r)

##          x1          x2          f
## 1  0.300775  2.84291  0.996905

# run III
soa_mrf_2(N,x1_lower,x1_upper,x2_lower,x2_upper,rot,k_max,r)

##          x1          x2          f
## 1  1.30355 -3.3419  0.923282
```

Berikutnya saya coba *run* sebanyak **100 kali**, berikut adalah rekap semua akar yang saya dapatkan:

```
##      x1      x2      f
## 1 -0.261   0.622 1.00
## 2 -0.257   0.640 1.00
## 3 -0.257   0.663 0.97
## 4 -0.248   0.685 0.99
## 5 -0.234   0.754 0.97
## 6 -0.192   0.963 0.93
## 7 -0.120   1.301 0.89
## 8 -0.105   1.371 0.89
## 9 -0.095   1.414 0.88
## 10 0.186   2.517 0.94
## 11 0.299   2.837 1.00
## 12 0.300   2.839 1.00
## 13 0.304   2.849 1.00
## 14 0.317   2.878 0.99
## 15 0.322   2.849 0.96
## 16 0.338   2.850 0.93
## 17 0.400   3.242 0.83
## 18 0.406   3.085 0.94
## 19 0.420   3.204 0.88
## 20 0.427   3.089 0.97
## 21 0.450   3.096 0.97
## 22 0.466   3.330 0.84
## 23 0.473   3.168 0.95
## 24 0.485   3.133 0.99
## 25 0.488   3.135 0.99
## 26 0.489   3.149 0.98
## 27 0.492   3.145 0.99
## 28 0.492   3.160 0.98
## 29 0.494   3.136 0.99
## 30 0.495   3.141 1.00
## 31 0.495   3.142 0.99
## 32 0.496   3.141 1.00
## 33 0.496   3.151 0.99
## 34 0.496   3.157 0.98
## 35 0.497   3.139 1.00
## 36 0.497   3.143 1.00
## 37 0.497   3.145 0.99
## 38 0.497   3.149 0.99
## 39 0.499   3.137 1.00
## 40 0.499   3.141 1.00
## 41 0.499   3.158 0.99
## 42 0.500   3.137 1.00
## 43 0.500   3.140 1.00
## 44 0.500   3.141 1.00
## 45 0.500   3.142 1.00
## 46 0.501   3.142 1.00
## 47 0.501   3.160 0.98
## 48 0.502   3.140 1.00
## 49 0.503   3.137 0.99
## 50 0.503   3.245 0.91
## 51 0.504   3.143 1.00
```

```
## 52 0.517 3.173 0.97
## 53 0.520 3.049 0.92
## 54 0.524 3.152 0.98
## 55 0.544 3.199 0.93
## 56 0.623 3.125 0.90
## 57 0.838 2.472 0.85
## 58 0.927 1.922 0.89
## 59 0.927 1.926 0.89
## 60 0.942 1.812 0.89
## 61 0.961 1.656 0.90
## 62 0.964 1.634 0.90
## 63 0.965 1.622 0.90
## 64 0.965 1.624 0.90
## 65 0.966 1.617 0.90
## 66 0.968 1.597 0.90
## 67 0.971 1.575 0.90
## 68 0.975 1.537 0.90
## 69 0.979 1.506 0.90
## 70 0.991 1.399 0.90
## 71 1.009 1.229 0.89
## 72 1.013 1.186 0.88
## 73 1.294 -3.127 0.99
## 74 1.337 -4.130 1.00
## 75 1.427 -6.594 0.83
## 76 1.433 -6.806 0.99
## 77 1.481 -8.370 0.99
## 78 1.481 -8.361 0.98
## 79 1.578 -12.160 0.99
## 80 1.583 -12.393 0.87
## 81 1.604 -13.348 0.99
## 82 1.715 -19.153 0.85
## 83 1.716 -19.212 0.85
```

== END ==