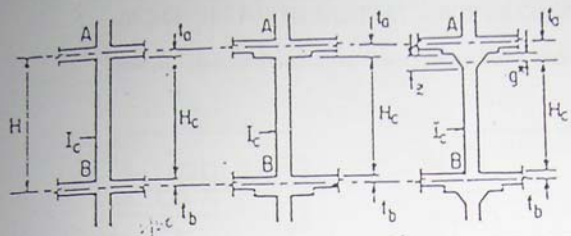
LAMPIRAN A

TABEL KONSTANTA UNTUK MOMEN DISTRIBUSI

Table A7—Stiffness and Carry-Over Factors for Columns



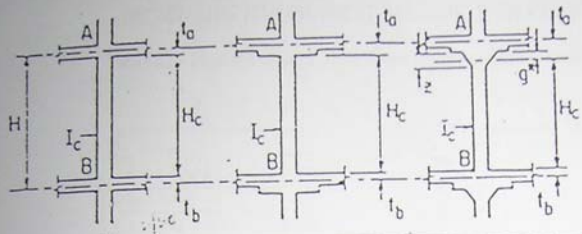
$K_c = k\frac{EI_c}{H}$

For values of k_{BA} and c_{BA} read (t_a/t_b) as (t_b/t_a)

*g can be approximated as 2/2.

t_a/t_c		H/H_c 1.05	1.10	1.15	1.20	1.25	1.30	1.35	1.40	1.45	1.50
0.00	k_{AB}	4.29	4.46	4.63	4.80	4.96	5.13	5.28	5.44	5.59	5.74
	C_{AB}	0.57	0.61	0.73	0.84	0.87	0.95	1.02	1.10	1.17	1.23
0.2	k_{AB}	4.31	4.62	4.95	5.30	5.65	6.02	6.40	6.79	7.20	7.62
	C_{AB}	0.56	0.62	0.68	0.74	0.80	0.85	0.91	0.96	1.01	1.07
0.4	k_{AB}	4.32	4.75	5.22	5.67	6.15	6.65	7.18	7.74	8.32	8.94
	C_{AB}	0.55	0.60	0.65	0.70	0.74	0.79	0.83	0.87	0.91	0.94
0.6	k_{AB}	4.44	4.91	5.42	5.96	6.54	7.15	7.81	8.50	9.23	10.01
	C_{AB}	0.55	0.59	0.63	0.67	0.70	0.74	0.77	0.80	0.83	0.85
0.8	k_{AB}	4.49	5.01	5.58	6.19	6.85	7.56	8.31	9.12	9.98	10.89
	C_{AB}	0.54	0.58	0.61	0.64	0.67	0.70	0.72	0.75	0.77	0.79
1.0	k_{AB}	4.52	5.09	5.71	6.38	7.11	7.89	8.73	9.63	10.60	11.63
	C_{AB}	0.54	0.57	0.60	0.62	0.65	0.67	0.69	0.71	0.73	0.74
1.2	k_{AB}	4.55	5.15	5.82	6.54	7.32	8.17	9.08	10.07	11.12	12.25
	C_{AB}	0.53	0.56	0.59	0.61	0.63	0.65	0.66	0.68	0.69	0.70
1.4	k_{AB}	4.56	5.21	5.91	6.68	7.51	8.41	9.38	10.43	11.57	12.78
	C_{AB}	0.53	0.55	0.58	0.60	0.61	0.63	0.64	0.65	0.66	0.67
1.6	k_{AB}	4.60	5.26	5.99	6.79	7.66	8.61	9.64	10.75	11.99	13.24
	C_{AB}	0.53	0.55	0.57	0.59	0.60	0.61	0.62	0.63	0.64	0.65
1.8	k_{AB}	4.62	5.30	6.06	6.89	7.80	8.79	9.87	11.03	12.29	13.65
	C_{AB}	0.52	0.55	0.56	0.58	0.59	0.60	0.61	0.61	0.62	0.63
2.0	k_{AB}	4.63	5.34	6.12	6.98	7.92	8.94	10.06	11.27	12.59	14.00
	C_{AB}	0.52	0.54	0.56	0.57	0.58	0.59	0.59	0.60	0.60	0.61
2.2	k_{AB}	4.65	5.37	6.17	7.05	8.02	9.08	10.24	11.49	12.85	14.31
	C_{AB}	0.52	0.54	0.55	0.56	0.57	0.58	0.58	0.59	0.59	0.59
2.4	k_{AB}	4.66	5.40	6.22	7.12	8.11	9.20	10.39	11.68	13.08	14.60
	C_{AB}	0.52	0.53	0.55	0.56	0.56	0.57	0.57	0.58	0.58	0.58
2.6	k_{AB}	4.67	5.42	6.26	7.18	8.20	9.31	10.53	11.86	13.29	14.85
	C_{AB}	0.52	0.53	0.54	0.55	0.56	0.56	0.56	0.57	0.57	0.57
2.8	k_{AB}	4.68	5.44	6.29	7.23	8.27	9.41	10.66	12.01	13.48	15.07
	C_{AB}	0.52	0.53	0.54	0.55	0.55	0.55	0.56	0.56	0.56	0.56
3.0	k_{AB}	4.69	5.46	6.33	7.28	8.34	9.50	10.77	12.15	13.65	15.28
	C_{AB}	0.52	0.53	0.54	0.54	0.55	0.55	0.55	0.55	0.55	0.55
3.2	k_{AB}	4.70	5.48	6.36	7.33	8.40	9.58	10.87	12.28	13.81	15.47
	C_{AB}	0.52	0.53	0.53	0.54	0.54	0.54	0.54	0.54	0.54	0.54
3.4	k_{AB}	4.71	5.50	6.38	7.37	8.45	9.65	10.97	12.40	13.95	15.64
	C_{AB}	0.51	0.52	0.53	0.53	0.54	0.54	0.54	0.53	0.53	0.53
3.6	k_{AB}	4.71	5.51	6.41	7.41	8.51	9.72	11.05	12.51	14.09	15.80
	C_{AB}	0.51	0.52	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.52
3.8	k_{AB}	4.72	5.53	6.43	7.44	8.56	9.78	11.13	12.60	14.21	15.95
	C_{AB}	0.51	0.52	0.53	0.53	0.53	0.53	0.53	0.52	0.52	0.52
4.0	k_{AB}	4.72	5.54	6.45	7.47	8.60	9.84	11.21	12.70	14.32	16.08
	C_{AB}	0.51	0.52	0.52	0.53	0.53	0.52	0.52	0.52	0.52	0.51
4.2	k_{AB}	4.73	5.55	6.47	7.50	8.64	9.90	11.27	12.78	14.42	16.20
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.52	0.52	0.51	0.51	0.51
4.4	k_{AB}	4.73	5.56	6.49	7.53	8.68	9.95	11.34	12.86	14.52	16.32
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.52	0.51	0.51	0.51	0.50
4.6	k_{AB}	4.74	5.57	6.51	7.55	8.71	9.99	11.40	12.93	14.61	16.43
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.52	0.51	0.51	0.50	0.50
4.8	k_{AB}	4.74	5.58	6.53	7.58	8.75	10.03	11.45	13.00	14.69	16.53
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.51	0.51	0.50	0.50	0.49
5.0	k_{AB}	4.75	5.59	6.54	7.60	8.78	10.07	11.50	13.07	14.77	16.62
	C_{AB}	0.51	0.51	0.52	0.52	0.51	0.51	0.51	0.50	0.49	0.49
5.5	k_{AB}	4.76	5.63	6.60	7.69	8.90	10.24	11.72	13.32	15.10	17.02
	C_{AB}	0.51	0.51	0.51	0.51	0.50	0.50	0.49	0.49	0.48	0.47
6.0	k_{AB}	4.78	5.66	6.65	7.75	9.00	10.37	11.88	13.54	15.35	17.32
	C_{AB}	0.51	0.51	0.51	0.50	0.50	0.49	0.48	0.48	0.47	0.46
6.5	k_{AB}	4.78	5.68	6.69	7.82	9.07	10.47	12.01	13.70	15.54	17.56
	C_{AB}	0.51	0.51	0.50	0.50	0.49	0.49	0.48	0.47	0.46	0.45
7.0	k_{AB}	4.79	5.69	6.71	7.86	9.13	10.55	12.11	13.83	15.70	17.74
	C_{AB}	0.50	0.50	0.50	0.50	0.49	0.48	0.47	0.46	0.45	0.45
10.0	k_{AB}	4.80	5.71	6.74	7.89	9.18	10.61	12.19	13.93	15.82	17.86
	C_{AB}	0.50	0.50	0.50	0.49	0.48	0.48	0.47	0.46	0.45	0.44

Table A7—Stiffness and Carry-Over Factors for Columns



$K_c = k\frac{EI_c}{H}$

For values of k_{BA} and c_{BA} read (t_a/t_b) as (t_b/t_a)

*g can be approximated as 2/2.

I_a/I_c	H_a/H_c	1.05	1.10	1.15	1.20	1.25	1.30	1.35	1.40	1.45	1.50
0.00	k_{AB}	4.29	4.46	4.63	4.80	4.96	5.13	5.29	5.45	5.61	5.77
	C_{AB}	0.57	0.55	0.53	0.52	0.50	0.49	0.48	0.47	0.46	0.45
0.2	k_{AB}	4.31	4.62	4.95	5.30	5.65	6.02	6.40	6.79	7.20	7.62
	C_{AB}	0.56	0.62	0.66	0.74	0.80	0.85	0.91	0.96	1.01	1.07
0.4	k_{AB}	4.32	4.75	5.22	5.67	6.15	6.65	7.18	7.74	8.32	8.94
	C_{AB}	0.55	0.60	0.65	0.70	0.74	0.79	0.83	0.87	0.91	0.94
0.6	k_{AB}	4.44	4.91	5.42	5.96	6.54	7.15	7.81	8.51	9.23	10.01
	C_{AB}	0.55	0.59	0.63	0.67	0.70	0.74	0.77	0.80	0.83	0.85
0.8	k_{AB}	4.48	5.01	5.58	6.19	6.85	7.56	8.31	9.12	9.98	10.89
	C_{AB}	0.54	0.58	0.61	0.64	0.67	0.70	0.72	0.75	0.77	0.79
1.0	k_{AB}	4.52	5.09	5.71	6.38	7.11	7.89	8.73	9.63	10.60	11.62
	C_{AB}	0.54	0.57	0.60	0.62	0.65	0.67	0.69	0.71	0.73	0.74
1.2	k_{AB}	4.55	5.16	5.82	6.54	7.32	8.17	9.08	10.07	11.12	12.25
	C_{AB}	0.53	0.56	0.59	0.61	0.63	0.65	0.66	0.68	0.69	0.70
1.4	k_{AB}	4.58	5.21	5.91	6.68	7.51	8.41	9.38	10.43	11.57	12.79
	C_{AB}	0.53	0.55	0.58	0.60	0.61	0.63	0.64	0.65	0.66	0.67
1.6	k_{AB}	4.60	5.26	5.99	6.79	7.66	8.61	9.64	10.75	11.95	13.24
	C_{AB}	0.53	0.55	0.57	0.59	0.60	0.61	0.62	0.63	0.64	0.65
1.8	k_{AB}	4.62	5.30	6.06	6.89	7.80	8.79	9.87	11.03	12.29	13.65
	C_{AB}	0.52	0.55	0.58	0.58	0.59	0.60	0.61	0.61	0.62	0.63
2.0	k_{AB}	4.63	5.34	6.12	6.98	7.92	8.94	10.06	11.27	12.59	14.00
	C_{AB}	0.52	0.54	0.56	0.57	0.58	0.59	0.59	0.60	0.60	0.61
2.2	k_{AB}	4.65	5.37	6.17	7.05	8.02	9.08	10.24	11.49	12.85	14.31
	C_{AB}	0.52	0.54	0.55	0.56	0.57	0.58	0.58	0.59	0.59	0.59
2.4	k_{AB}	4.66	5.40	6.22	7.12	8.11	9.20	10.39	11.68	13.08	14.60
	C_{AB}	0.52	0.53	0.55	0.56	0.56	0.57	0.57	0.58	0.58	0.58
2.6	k_{AB}	4.67	5.42	6.26	7.18	8.20	9.31	10.53	11.86	13.29	14.85
	C_{AB}	0.52	0.53	0.54	0.55	0.56	0.56	0.56	0.57	0.57	0.57
2.8	k_{AB}	4.68	5.44	6.29	7.23	8.27	9.41	10.66	12.01	13.48	15.07
	C_{AB}	0.52	0.53	0.54	0.55	0.55	0.55	0.56	0.56	0.56	0.56
3.0	k_{AB}	4.69	5.46	6.33	7.28	8.34	9.50	10.77	12.15	13.65	15.29
	C_{AB}	0.52	0.53	0.54	0.54	0.55	0.55	0.55	0.55	0.55	0.55
3.2	k_{AB}	4.70	5.48	6.36	7.33	8.40	9.58	10.87	12.28	13.81	15.47
	C_{AB}	0.52	0.53	0.53	0.54	0.54	0.54	0.54	0.54	0.54	0.54
3.4	k_{AB}	4.71	5.50	6.38	7.37	8.45	9.65	10.97	12.40	13.95	15.64
	C_{AB}	0.51	0.52	0.53	0.53	0.54	0.54	0.54	0.53	0.53	0.53
3.6	k_{AB}	4.71	5.51	6.41	7.41	8.51	9.73	11.05	12.51	14.09	15.80
	C_{AB}	0.51	0.52	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.52
3.8	k_{AB}	4.72	5.53	6.43	7.44	8.56	9.78	11.13	12.60	14.21	15.95
	C_{AB}	0.51	0.52	0.53	0.53	0.53	0.53	0.53	0.52	0.52	0.52
4.0	k_{AB}	4.72	5.54	6.45	7.47	8.60	9.84	11.21	12.70	14.32	16.08
	C_{AB}	0.51	0.52	0.52	0.53	0.53	0.52	0.52	0.52	0.52	0.51
4.2	k_{AB}	4.73	5.55	6.47	7.50	8.64	9.90	11.27	12.78	14.42	16.20
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.52	0.52	0.51	0.51	0.51
4.4	k_{AB}	4.73	5.56	6.49	7.53	8.68	9.95	11.34	12.86	14.52	16.32
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.52	0.51	0.51	0.51	0.50
4.6	k_{AB}	4.74	5.57	6.51	7.55	8.71	9.99	11.40	12.93	14.61	16.43
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.52	0.51	0.51	0.50	0.50
4.8	k_{AB}	4.74	5.58	6.53	7.58	8.75	10.03	11.45	13.00	14.69	16.53
	C_{AB}	0.51	0.52	0.52	0.52	0.52	0.51	0.51	0.50	0.50	0.49
5.0	k_{AB}	4.75	5.59	6.54	7.60	8.78	10.07	11.50	13.07	14.77	16.62
	C_{AB}	0.51	0.51	0.52	0.52	0.51	0.51	0.51	0.50	0.49	0.49
5.5	k_{AB}	4.75	5.63	6.60	7.69	8.90	10.24	11.72	13.35	15.10	17.02
	C_{AB}	0.51	0.51	0.51	0.51	0.50	0.50	0.49	0.49	0.48	0.47
6.0	k_{AB}	4.78	5.66	6.65	7.76	9.00	10.37	11.88	13.54	15.35	17.32
	C_{AB}	0.51	0.51	0.51	0.50	0.50	0.49	0.48	0.48	0.47	0.46
6.5	k_{AB}	4.78	5.68	6.69	7.82	9.07	10.47	12.01	13.70	15.54	17.56
	C_{AB}	0.51	0.51	0.50	0.50	0.49	0.49	0.48	0.47	0.46	0.45
7.0	k_{AB}	4.79	5.69	6.71	7.86	9.13	10.55	12.11	13.83	15.70	17.74
	C_{AB}	0.50	0.50	0.50	0.50	0.49	0.48	0.47	0.46	0.45	0.45
7.5	k_{AB}	4.80	5.71	6.74	7.89	9.18	10.61	12.19	13.93	15.83	17.90
	C_{AB}	0.50	0.50	0.50	0.49	0.48	0.48	0.47	0.46	0.45	0.44

LAMPIRAN B

PROGRAM ALGORITMA CONTOH SEDERHANA

```
%=========================================================
=====
% Algoritma Genetika Standar (dengan grafis 2D) terdiri dari:
%
% 1. Satu populasi dengan UkPop kromosom
% 2. Binary encoding
% 3. Linear fitness ranking
% 4. Roulette-wheel selection
% 5. Pindah silang satu titik potong
% 6. Probabilitas pindah silang dan probabilitas mutasi bernilai tetap
% 7. Elitisme, satu atau dua buah kopi dari individu bernilai fitness
%    tertinggi
% 8. Generational replacement : mengganti semua individu dengan individu
%    baru
%=========================================================
=====
%
Clc                         % Me-refresh command window
clear all                   % Menghapus semua variabel yang sedang aktif
%
Nvar    =  2;               % Jumlah variabel pada fungsi yang dioptimasi
Nbit    = 10;               % Jumlah bit yang mengkodekan satu variabel
JumGen  = Nbit*Nvar;        % Jumlah gen dalam kromosom
Rb      = -5.12;            % Batas bawah interval
Ra      = 5.12;             % Batas atas interval
%
UkPop   = 200;              % Jumlah kromosom dalam populasi
Psilang = 0.8;              % Probabilitas pindah silang
Pmutasi = 0.05;             % Probabilitas mutasi
MaxG    = 100;              % Jumlah generasi
%
BilKecil = 10^-1;           % Digunakan untuk menghindari pembagian dengan
0
Fthreshold = 1/BilKecil;    % Threshold untuk nilai Fitness
Bgraf      = Fthreshold;    % Untuk menangani tampilan grafis
%
% Inisialisasi grafis 2D
%
hfig = figure;
hold on
title ('Optimasi fungsi menggunakan AG standar dengan grafis 2 dimensi')
set(hfig, 'position', [50,50,600,400]);
set(hfig, 'DoubleBuffer', 'on');
axis([1 MaxG 0 Bgraf]);
hbestplot = plot(1:MaxG, zeros(1,MaxG));
htext1 = text(0.6*MaxG,0.25*Bgraf,sprintf('Fitness terbaik : %7.4f', 0.0));
htext2 = text(0.6*MaxG,0.20*Bgraf,sprintf('Variabel X1     : %5.4f', 0.0));
htext3 = text(0.6*MaxG,0.15*Bgraf,sprintf('Variabel X2     : %5.4f', 0.0));
```

```
htext4 = text(0.6*MaxG,0.10*Bgraf,sprintf('Nilai minimum : %5.4f', 0.0));
xlabel('Generasi');
ylabel('Fitness terbaik');
hold off
drawnow;
%
% Inisialisasi Populasi
%
Populasi = InisialisasiPopulasi(UkPop,JumGen);
%
% Loop evolusi
%
for generasi = 1:MaxG,
   x = DekodekanKromosom(Populasi(1,:),Nvar,Nbit,Ra,Rb);
   Fitness(1) = EvaluasiIndividu(x,BilKecil);
   MaxF = Fitness(1);
   MinF = Fitness(1);
   IndeksIndividuTerbaik = 1;
   for ii = 2:UkPop,
      Kromosom = Populasi(ii,:);
      x = DekodekanKromosom(Kromosom,Nvar,Nbit,Ra,Rb);
      Fitness(ii) = EvaluasiIndividu(x,BilKecil);
      if (Fitness(ii) > MaxF),
         MaxF = Fitness(ii);
         IndeksIndividuTerbaik = ii;
         BestX = x;
      end
      if (Fitness(ii) < MinF),
         MinF = Fitness(ii);
      end
   end
   %
   % Penanganan grafis 2D
   %
   plotvector = get(hbestplot, 'YData');
   plotvector(generasi) = MaxF;
   set(hbestplot, 'YData', plotvector);
   set(htext1, 'String', sprintf('Fitness terbaik : %7.4f', MaxF));
   set(htext2, 'String', sprintf('Variabel X1     : %5.4f', BestX(1)));
   set(htext3, 'String', sprintf('Variabel X2     : %5.4f', BestX(2)));
   set(htext4, 'String', sprintf('Nilai minimum   : %5.4f', (1/MaxF)-BilKecil));
   %
   if MaxF >= Fthreshold,
      break;
   end
   %
   TemPopulasi = Populasi;
   %
```

```
   % Elitisme:
   % - Buat satu kopi kromosom terbaik jika ukuran populasi ganjil
   % - Buat dua kopi kromosom terbaik jika ukuran populasi genap
   %
   if mod(UkPop,2) == 0,                % ukuran populasi genap
      IterasiMulai = 3;
      TemPopulasi(1,:) = Populasi(IndeksIndividuTerbaik,:);
      TemPopulasi(2,:) = Populasi(IndeksIndividuTerbaik,:);
   else
      InterasiMulai = 2;
      TemPopulasi(1,:) = Populasi(IndeksIndividuTerbaik,:);
   end
   %
   LinearFitness = LinearFitnessRanking(UkPop,Fitness,MaxF,MinF);
   %
   % Roulette-wheel dan pindah silang
   %
   for jj = IterasiMulai:2:UkPop,
      IP1 = RouletteWheel(UkPop,LinearFitness);
      IP2 = RouletteWheel(UkPop,LinearFitness);
      if(rand < Psilang),
         Anak = PindahSilang(Populasi(IP1,:),Populasi(IP2,:),JumGen);
         TemPopulasi(jj,:)   = Anak(1,:);
         TemPopupasi(jj+1,:) = Anak(2,:);
      else
         TemPopulasi(jj,:)   = Populasi(IP1,:);
         TemPopulasi(jj+1,:) = Populasi(IP2,:);
      end
   end
   %
   % Mutasi dilakukan pada semua kromosom
   for kk = IterasiMulai:UkPop,
      TemPopulasi(kk,:) = Mutasi(TemPopulasi(kk,:),JumGen,Pmutasi);
   end
   %
   % Generational Replacement: mengganti semua kromosom sekaligus
   %
   Populasi = TemPopulasi;
end

%=========================================================
=====
% Membangkitkan sejumlah UkPop kromosom, masing-masing kromosom
% berisi bilangan biner (0 dan 1) sejumlah JumGen
%
% Masukkan
%   UkPop : ukuran populasi atau jumlah kromosom dalam populasi
%   JumGen: jumlah gen dalam kromosom
```

```
%
% Keluaran
%   Populasi : kumpulan kromosom, matriks berukuran UkPop x JumGen
%=========================================================
=====

function Populasi = Inisialisasi(UkPop,JumGen)

Populasi = fix(2*rand(UkPop,JumGen));

%=========================================================
=====
% Mendekodekan kromosom yang berisi bilangan biner menjadi individu x yang
% bernilai real dalam interval yang ditentukan (Ra,Rb)
%
% Masukan
%   Kromosom : kromosom, matriks berukuran 1 x JumGen
%   Nvar     : jumlah variabel
%   Nbit     : jumlah bit yang mengkodekan satu variabel
%   Ra       : batas atas interval
%   Rb       : batas bawah interval
%
% Keluaran
%   x : individu hasil dekode kromosom
%=========================================================
=====

function x = DekodekanKromosom(Kromosom,Nvar,Nbit,Ra,Rb)

for ii = 1:Nvar,
   x(ii) = 0;
   for jj = 1:Nbit,
      x(ii) = x(ii) + Kromosom((ii-1)*Nbit + jj)*2^(-jj);
   end
   x(ii) = Rb + (Ra-Rb)*x(ii);
end

%=========================================================
=====
% Mengevaluasi individu sehingga didapatkan nilai fitness-nya
%
% Masukan
%   x       : individu
%   BilKecil : bilangan kecil digunakan untuk menghindari pembagian
%              dengan 0
%
% Keluaran
%   fitness : nilai fitness
```

```
%=========================================================
=====

function fitness = EvaluasiIndividu(x, BilKecil)

fitness = 1/((1000*(x(1) - 2*x(2))^2 + (1 - x(1))^2) + BilKecil);

%=========================================================
=====
% Menskalakan nilai fitness ke dalam ranking sehingga diperoleh
% nilai-nilai fitness baru yang berada dalam rentang (MaxF,MinF)
%
% Masukan
%   UkPop   : ukuran populasi atau jumlah kromosom dalam populasi
%   Fitness : nilai fitness, matriks ukuran 1 x UkPop
%   MaxF    : nilai fitness maximum
%   MinF    : nilai fitness minimum
%
% Keluaran
%   LFR : Linear Fitness Ranking
%
%=========================================================
=====

function LFR = LinearFitnessRanking(UkPop,Fitness,MaxF,MinF)

[SF,IndF] = sort(Fitness);

for rr = 1:UkPop,
   LFR(IndF(UkPop-rr+1)) = MaxF-(MaxF-MinF)*((rr-1)/(UkPop-1));
End

%=========================================================
=====
% Memilih orang tua menggunakan LinearFitness, yaitu nilai fitness hasil
% pen-skala-an. Pemilihan dilakukan secara proporsional sesuai dengan
% nilai fitness-nya
%
% Masukan
%   UkPop : ukuran populasi atau jumlah kromosom dalam populasi
%   LinearFitness : nilai fitness yang sudah di-skala-kan
%
% Keluaran
%   Pindex : indeks dari kromosom yang terpilih (bernilai 1 sampai UkPop)
%=========================================================
=====

function Pindex = RoulettWheel(UkPop,LinearFitness);
```

```
JumFitness = sum(LinearFitness);
KumulatifFitness = 0;
RN = rand;
ii =1;

while ii <= UkPop,
   KumulatifFitness = KumulatifFitness + LinearFitness(ii);
   if (KumulatifFitness/JumFitness) > RN,
      Pindex = ii;
      break;
   end
   ii = ii + 1;
end

%=========================================================
=====
% Memindah-silangkan bagian kromosom Bapak dan Ibu yang dipotong
% secara random, sehingga dihasilkan dua buah kromosom Anak
%
% Masukan
%   Bapak  : kromosom, matriks berukuran 1 x JumGen
%   Ibu    : kromosom, matriks berukuran 1 x JumGen
%   JumGen : jumlah gen
%
% Keluaran
%   Anak : kromosom hasil pindah silang, matriks berukuran 1 x JumGen
%=========================================================
=====

function Anak = PindahSilang(Bapak,Ibu,JumGen);

TP = 1 + fix(rand*(JumGen-1));
Anak(1,:) = [Bapak(1:TP) Ibu(TP+1:JumGen)];
Anak(2,:) = [Ibu(1:TP) Bapak(TP+1:JumGen)];

%=========================================================
=====
% Mutasi gen dengan probabilitas sebesar Pmutasi
% Gen-gen yang terpilih diubah nilainya: 0 menjadi 1, dan 1 menjadi 0
%
% Masukan
%   Kromosom : kromosom, matriks berukuran 1 x JumGen
%   JumGen   : jumlah gen
%   Pmutasi  : probabilitas mutasi
%
% Keluaran
%   MutKrom : kromosom hasil mutasi, matriks berukuran 1 x JumGen
```

```
%=========================================================
=====

function MutKrom = Mutasi(Kromosom,JumGen, Pmutasi);

MutKrom = Kromosom;
for ii = 1:JumGen,
   if (rand < Pmutasi),
      if Kromosom(ii) == 0,
         MutKrom(ii) = 1;
      else
         MutKrom(ii) = 0;
      end
   end
end


%=========================================================
=====
% Mencari parameter optimal: ukuran populasi dan probabilitas mutasi
%
% Pencarian parameter optimal dilakukan dengan mengkombinasikan kedua
% parameter tersebut. Dalam observasi ini, digunakan empat nilai untuk
% masing-masing parameter, sehingga diperoleh 16 kombinasi
% Nilai untuk kedua parameter adalah sebagai berikut:
% - UkPop   : 50, 100, 200 dan 400
% - Pmutasi : 0.01, 0.05, 0.1 dan 0.2
%
%=========================================================
=====
%
clc                         % Me-refresh command window
clear all                   % Menghapus semua variabel yang sedang aktif
%
Nvar    =  2;               % Jumlah variabel pada fungsi yang dioptimasi
Nbit    = 10;               % Jumlah bit yang mengkodekan satu variabel
JumGen  = Nbit*Nvar;            % Jumlah gen dalam kromosom
Rb      = -5.12;            % Batas bawah interval
Ra      = 5.12;             % Batas atas interval
%
Psilang   = 0.8;            % Probabilitas pindah silang
MaxJumInd = 60000;              % Jumlah generasi
%
BilKecil   = 10^-1;             % Digunakan untuk menghindari pembagian dengan
0
Fthreshold = 1/BilKecil;        % Threshold untuk nilai Fitness
Bgraf      = Fthreshold;        % Untuk menangani tampilan grafis
%
```

```
ObUkPop   = [50 100 200 400];     % Ukuran populasi yang diobservasi
ObPmutasi = [0.01 0.05 0.1 0.2]   % Probabilitas mutasi yang diobservasi
%
ObData = [];                      % Data hasil observasi
%
for ukp = 1:length(ObUkPop),
   UkPop = ObUkPop(ukp);
   MaxG  = fix(MaxJumInd/UkPop);
   for pm = 1:length(ObPmutasi),
      Pmutasi = ObPmutasi(pm);
      for observasi = 1:10,
         UkPop, Pmutasi, observasi
         %
         % Inisialisasi Populasi
         %
         Populasi = InisialisasiPopulasi(UkPop,JumGen);
         %
         % Loop evolusi
         %
         for generasi = 1:MaxG,
            x = DekodekanKromosom(Populasi(1,:),Nvar,Nbit,Ra,Rb);
            Fitness(1) = EvaluasiIndividu(x,BilKecil);
            MaxF = Fitness(1);
            MinF = Fitness(1);
            IndeksIndividuTerbaik = 1;
            for ii = 2:UkPop,
               Kromosom = Populasi(ii,:);
               x = DekodekanKromosom(Kromosom,Nvar,Nbit,Ra,Rb);
               Fitness(ii) = EvaluasiIndividu(x,BilKecil);
               if (Fitness(ii) > MaxF),
                  MaxF = Fitness(ii);
                  IndeksIndividuTerbaik = ii;
                  BestX = x;
               end
               if (Fitness(ii) < MinF),
                  MinF = Fitness(ii);
               end
            end
            %
            TemPopulasi = Populasi;
            %
            % Elitisme:
            % - Buat satu kopi kromosom terbaik jika ukuran populasi ganjil
            % - Buat dua kopi kromosom terbaik jika ukuran populasi genap
            %
            if mod(UkPop,2) == 0,          % ukuran populasi genap
               IterasiMulai = 3;
               TemPopulasi(1,:) = Populasi(IndeksIndividuTerbaik,:);
```

```
      TemPopulasi(2,:) = Populasi(IndeksIndividuTerbaik,:);
    else
      InterasiMulai = 2;
      TemPopulasi(1,:) = Populasi(IndeksIndividuTerbaik,:);
    end
    %
    LinearFitness = LinearFitnessRanking(UkPop,Fitness,MaxF,MinF);
    %
    % Roulette-wheel dan pindah silang
    %
    for jj = IterasiMulai:2:UkPop,
      IP1 = RouletteWheel(UkPop,LinearFitness);
      IP2 = RouletteWheel(UkPop,LinearFitness);
      if(rand < Psilang),
        Anak = PindahSilang(Populasi(IP1,:),Populasi(IP2,:),JumGen);
        TemPopulasi(jj,:)   = Anak(1,:);
        TemPopupasi(jj+1,:) = Anak(2,:);
      else
        TemPopulasi(jj,:)   = Populasi(IP1,:);
        TemPopulasi(jj+1,:) = Populasi(IP2,:);
      end
    end
    %
    % Mutasi dilakukan pada semua kromosom
    %
    for kk = IterasiMulai:UkPop,
      TemPopulasi(kk,:) = Mutasi(TemPopulasi(kk,:),JumGen,Pmutasi);
    end
    %
    % Generational Replacement: mengganti semua kromosom sekaligus
    %
    Populasi = TemPopulasi;
    %
    if MaxF >= Fthreshold,
      JumIndData(observasi) = generasi*UkPop;
      MaxFData(observasi)   = MaxF;
      break;
    else
      if generasi == MaxG,
        JumIndData(observasi) = MaxG*UkPop;
        MaxFData(observasi)   = MaxF;
      end
    end
    %
  end    % loop evolusi
end      % loop observasi
%
ObData = [ObData ; [UkPop Pmutasi mean(MaxFData) ...
```

```
                mean(JumIndData)]];
   end
end
%
save ObData.mat ObData
%
clc                     % me-refresh layar
%
disp(['Mencari nilai optimal : Ukuran Populasi dan Prob. Mutasi']);
disp(['Jumlah maksimum individu yang dievaluasi adalah ', ...
      num2str(MaxJumInd)]);
disp([']);
disp(['-------------------------------------------------------']);
disp(['Ukuran    Probabilitas    Rata-rata   Rata-rata    ']);
disp(['Populasi  mutasi          Fitness     Jumlah individu']);
disp(['-------------------------------------------------------']);
%
for ii = 1:length(ObData(:,1)),
   disp(['   ', num2str(ObData(ii,1)),  '      ', num2str(ObData(ii,2)), ...
        '      ', num2str(ObData(ii,3)), '      ', num2str(ObData(ii,4))]);
end
disp(['-------------------------------------------------------']);
```