Lampiran A

Data Simulasi DDE23

A.1 Model SI

tspan	S(t)	I(t)
0	0.98	0.02
0.351744439188879	0.97965535106732	0.0203446489326795
2.11046663513327	0.977933923656643	0.0220660763433573
7.11046663513327	0.973056457838782	0.0269435421612182
10.5552333175666	0.969710272090502	0.0302897279094977
14	0.96637559334058	0.0336244066594197
17.6086730138809	0.962587750008204	0.0374122499917965
21.2173460277619	0.958205327711661	0.0417946722883391
24.6086730138809	0.953554230546431	0.046445769453569
28	0.948396729264127	0.0516032707358727
29.9496717437118	0.945203626800449	0.0547963731995511
31.8993434874236	0.941832157582483	0.0581678424175175
33.8658976114384	0.938236644844562	0.0617633551554381
35.8968057352586	0.934303485821393	0.0656965141786068
37.9988297080576	0.929981844068282	0.0700181559317177
39.9994148540288	0.92561794051592	0.0743820594840799
42	0.92099662503851	0.0790033749614897
44.3588270391275	0.915200881360275	0.0847991186397253
46.7604766965582	0.908894807785957	0.0911051922140429
48.3802383482791	0.904399885052279	0.0956001149477206
50	0.89970186540753	0.10029813459247

A.2 Model SIR

tspan	S(t)	I(t)	R(t)
0	0.98	0.02	0
0.0258532162803826	0.979974664175553	0.0200080968575488	1.72389668983113e-05
0.155119297682296	0.97984799487804	0.0200484668364842	0.000103538285475812
0.801449704691861	0.979214893942928	0.0202474844550344	0.000537621602037755
1.67343824001644	0.978361401953462	0.0205086419734896	0.00112995607304816
2.59059879602482	0.977464498830996	0.0207744686427264	0.00176103252627811
3.6648981828509	0.976414973172528	0.0210746514462314	0.00251037538124027
4.88859549464926	0.975220867555491	0.021402390357011	0.00337674208749808
6.25804168593616	0.973886269091576	0.0217519633528803	0.00436176755554386
7.77251068611185	0.972412464824259	0.0221183456726871	0.00546918950305386
9.43355438284328	0.970798585961354	0.0224969850443838	0.00670442899426242
11.244656914816	0.969041961377621	0.0228836636378955	0.00807437498448305
12.622328457408	0.967707859032135	0.0231604831234086	0.00913165784445664
14	0.966375593376017	0.0234230826643503	0.0102013239596329
16.1915599356222	0.964224715068224	0.023847551800321	0.0119277331314546
18.1346631041981	0.962265596015033	0.02424916383855	0.0134852401464172
20.1661929481351	0.960169431286037	0.0246884507031734	0.0151421180107894
22.2912906852277	0.957928828063248	0.0251634579886108	0.0169077139481407
24.5212215446646	0.955529859850067	0.0256731100440935	0.0187970301058399
26.2606107723323	0.953627661715476	0.0260751470686205	0.0202971912159034
28	0.951700674839492	0.0264786114344107	0.0218207137260968
30.5691742290831	0.948811342932857	0.0270748175370346	0.0241138395301086
33.419577167025	0.945541681996612	0.0277404354998105	0.0267178825035774
36.8528870085331	0.941507781363386	0.0285532319136773	0.0299389867229365
39.4264435042665	0.938413135637469	0.0291719495887443	0.0324149147737864
42	0.935257119877057	0.0297986146706749	0.0349442654522686
46	0.930230987271728	0.0307859231783074	0.0389830895499641
50	0.925058897615495	0.0317866325897423	0.0431544697947632

A.3 Model SIS

tspan	S(t)	I(t)
0	0.98	0.02
1.100136862995	0.979661718560016	0.0203382814399836
3.51394573124205	0.978962771160841	0.0210372288391589
6.02590189385605	0.978294364252701	0.0217056357472994
8.63848723827681	0.977657707513853	0.0223422924861471
11.3192436191384	0.977061172681293	0.0229388273187065
14	0.976517093324099	0.0234829066759012
15.9950118692106	0.976114419020334	0.0238855809796658
17.9900237384212	0.975683491829155	0.0243165081708451
20.0442459662689	0.975216110040299	0.0247838899597009
22.1939353575349	0.974707088728482	0.0252929112715179
24.4505221404843	0.974156694182751	0.0258433058172491
26.2252610702421	0.973715818640801	0.0262841813591983
28	0.973270487357535	0.0267295126424647
30.6000127922737	0.972612312857465	0.027387687142535
33.3561366356451	0.971903024101459	0.0280969758985406
36.5846430770189	0.971051391563373	0.0289486084366265
39.2923215385094	0.970317429657762	0.0296825703422376
42	0.969564913891414	0.0304350861085854
46	0.968420548078575	0.0315794519214248
50	0.967238175231891	0.032761824768109

A.4 Model SIRS

tspan	S(t)	I(t)	R(t)
0	0.98	0.02	0
0.0258532162803826	0.9799746716009	0.0200080968576128	1.72315414868442e-05
0.155119297682296	0.979848261975519	0.0200484668502938	0.00010327117418727
0.801449704691861	0.979221995267898	0.020247486350236	0.000530518381865884
3.03163240246897	0.977133695151774	0.0208992537478606	0.00196705110036525
5.16832385154171	0.975235392076686	0.0214757464677222	0.00328886145559213
7.42819119290708	0.973332264537673	0.0220382672177037	0.00462946824462274
9.76935879504235	0.971469209449214	0.0225737752144807	0.00595701533630503
11.8846793975212	0.969876692980531	0.0230194443493341	0.00710386267013482
14	0.968366591482543	0.0234316038822393	0.00820180463521777
15.633291030005	0.967234911923235	0.0237479176732914	0.00901717040347372
17.2665820600099	0.966110698130014	0.0240827018191194	0.00980660005086627
18.9581401853115	0.964955240359014	0.0244454924154448	0.0105992672255412
20.7744842005231	0.963725725096957	0.0248496299340883	0.0114246449689549
22.7308570895304	0.962415734358439	0.0252978293512015	0.0122864362903593
24.8467436282236	0.961017280131317	0.0257931851071375	0.0131895347615449
26.4233718141118	0.959988646325081	0.0261669069871477	0.0138444466877715
28	0.958972215522548	0.0265427609274312	0.0144850235500205
30.6107320832867	0.957315564477353	0.0271680423674588	0.0155163931551878
33.3404735491564	0.95561140952185	0.0278296506378393	0.0165589398403108
36.1179947981813	0.953898625777251	0.0285151013050961	0.0175862729176528
38.982179460666	0.952148137822993	0.0292371448841228	0.0186147172928833
42	0.950315765686246	0.0300152094308282	0.0196690248829257
45.159925258371	0.948406434609724	0.0308479598154594	0.0207456055748162
47.5799626291855	0.946948458294499	0.0314975498426741	0.0215539918628267
50	0.945492226174767	0.0321573042114736	0.0223504696137591

Lampiran B

Algoritma Genetika

B.1 Model SI

```
% Kode ini akan menjalankan genetic algorithm untuk mencari parameter tidak
% diketahui dari model (dalam hal ini, semua nilai mu)

clear
clc

% Parameter:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        : step size
%   n,p      : ukuran matriks populasi; jumlah kromosom x ukuran kromosom
%   sval     : nilai seleksi
%   cval     : nilai crossover/persilangan
%   mval     : nilai mutasi
%   snum     : jumlah kromorom yang akan mengalami seleksi
%   cnum     : jumlah kromorom yang akan mengalami crossover
%   mnum     : jumlah kromorom yang akan mengalami mutasi

t0 = 0;
t1 = 50;
tspan=[t0,t1];
x0 = [0.98 0.02];

p = 2;
```

```
n = 40;

sval = 0.7;
cval = 0.7;
mval = 0.7;
snum = n*sval;
cnum = n*cval;


% INITIATION : menghasilkan matriks populasi awal secara acak
%C=zeros(n,3);
%C(:,1:2)=rand(n,2);
%C(:,3)=rand(n,1);

C = rand(n,p);
J = zeros(n,1);

% EVALUATION : menghasilkan nilai fitness untuk masing-masing
    kromosom
for i = 1 : n
     fit = fitnessFunction2param(x0,tspan,C(i,:));
     %disp(fit);
     J(i) = 1/((1+fit));
end

%disp(C);
%disp(J);
step = 1;
%tic

[maxJ,maxJ_id] = max(J);

while max(J) < 1 - 0.0000001
```

```
57
58      % Matriks populasi baru
59      C1 = zeros(n,p);
60      %disp(step);
61      % SELECTION :
62      % 1. Sorting populasi berdasarkan nilai fitness menggunakan
63      % quicksort
64      [J,C] = quicksort(J,C);
65
66      % 2. Pilih sebanyak snum kromosom yang bertahan hidup
67      C1(snum+1:n,:) = C(snum+1:n,:);
68
69      % 3. Bangkitkan n-snum kromosom lainnya berdasarkan snum
70      % kromosom yang bertahan hidup
71      pivot = C1(n,:);
72      for i = 1 : snum
73          for j = 1 : p
74              C1(i,j) = pivot(1,j)+0.1*(rand() -0.5);
75          %C1(i,:) = rand(1,3);%C1(randi(snum),:);
76          end
77      end
78      for i = 1 : n
79          fit = fitnessFunction2param(x0,tspan,C1(i,:));
80          J(i) = 1/((1+fit));
81      end
82      [maxJ,maxJ_id] = max(J);
83
84      fprintf('Selection \n')
85      disp(maxJ);
86      disp(maxJ_id);
87
88      % CROSSOVER :
```

```
89      % 1. Tentukan sebanyak cnum yang akan mengalami proses
            crossover secara
90      % acak
91      %   - B1 merupakan penanda (flag) apakah kromosom tersebut
            akan
92      %   mengalami crossover atau tidak
93      %   - Jika B1(i) = 1, maka kromosom ke-i akan terpilih untuk
            crossover
94      [J,C1] = quicksort(J,C1);
95      B1 = zeros(n-3,1);
96      B1(randperm(numel(B1),cnum)) = 1;
97
98      % 2. Tentukan pasangan parent yang akan mengalami crossover
            secara acak
99      k1 = randi([10,20]);
100     p1 = C1(n,:);
101     p2 = C1(n-1,:);
102     p3 = C1(n-2,:);
103
104     % 3. Proses crossover untuk setiap pasangan parent
105     for i = 1 : k1
106         if B1(i)==1
107           C1(i,:) = offspring(p1,p2);
108         end
109     end
110     for i = k1+1 : n-3
111         if B1(i)==1
112           C1(i,:) = offspring(p1,p3);
113         end
114     end
115     for i = 1 : n
116         fit = fitnessFunction2param(x0,tspan,C1(i,:));
117         J(i) = 1/((1+fit));
```

```
118     end
119     [maxJ,maxJ_id] = max(J);
120
121     fprintf('Crossover \n')
122     disp(maxJ);
123     disp(maxJ_id);
124
125     % MUTATION :
126     % 1. Tentukan sebanyak mnum yang akan mengalami proses mutasi
            secara
127     % acak
128     %    - B2 merupakan penanda (flag) apakah kromosom tersebut
            akan
129     %    mengalami mutasi atau tidak
130     %    - Jika B2(i) = 1, maka kromosom ke-i akan terpilih untuk
            mutasi
131     %B2 = zeros(n,1);
132     %B2(randperm(numel(B2),mnum)) = 1;
133
134     B2 = rand(size(C1))<mval;
135
136     [maxJ,maxJ_id] = max(J);
137     for j = 1:p
138         if B2(maxJ_id,j) == 1
139             B2(maxJ_id,j) = 0;
140         end
141     end
142
143     %disp(B2);
144     % 2. Proses mutasi untuk setiap kromosom terpilih
145     %    - Untuk setiap gen, pilih suatu angka acak b, jika b<mval
            maka
146     %    ubah nilai gen
```

```
147     for i = 1 : n
148         %if B2(i) == 1
149             for j = 1 : p
150                 %flag = 1;
151                 %while flag == 1
152                 %    C1(i,j) = abs(C1(i,j) + 0.1*(rand() - 0.5));
153                 %    flag = any([C1(i,j)<0,C1(i,j)>1]);
154                 %end
155                 if B2(i,j) == 1
156                     C1(i,j) = abs(C1(i,j)+0.4*(rand()-0.5));
157                     if C1(i,j) >= 1
158                         C1(i,j) = C1(i,j) - 1;
159                     end
160                 end
161             end
162         %end
163     end
164     fprintf('Mutation \n')
165
166     C = C1;
167     for i = 1 : n
168         fit = fitnessFunction2param(x0,tspan,C(i,:));
169         J(i) = 1/((1+fit));
170     end
171     [maxJ,maxJ_id] = max(J);
172     disp(maxJ);
173     disp(maxJ_id);
174     %fprintf('%d %f \n',step,maxJ )
175     step = step + 1;
176
177     % REPETITION : lakukan semua proses di atas untuk setiap
        generasi baru,
```

```
178     % hingga terdapat satu kromosom yang memenuhi J(i) < epsilon =
            0.01
179 end
180 %toc
181
182 % Output dari sistem
183 result = C(maxJ_id,:)
```

iLovePDF

B.2 Fungsi *Fitness* Model SI

```
function F = fitnessFunction2param(history,tspan,param)
%function J = fitnessFunction(t0,t1,x0,h,param)
% Fungsi ini menghasilkan nilai fitness dari kromosom yang
    dimasukkan
% menggunakan fungsi error.
% Input:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        : step size
%   param    : kromosom yang akan dicari nilai fitnessnya (dalam
    hai ini,
%              semua nilai mu)
% Output:
%   J        : nilai fitness dari kromosom

% Dummy data
Y1=readtable('SIDelayTspan50.txt');
Y2=table2array(Y1);
beta=param(1);
r=param(2)*30;
% Solusi numerik yang dihasilkan berdasarkan nilai kromosom
opts = ddeset('RelTol',1e-5,'AbsTol',1e-8);
%[T,Y] = dde23('SIRmodelDelay',r,x0,tspan,opts,beta,gamma);
sol1 = dde23('SImodelDelay',r,history,tspan,opts,beta);
Y=sol1.y';
T=sol1.x';


[m n] = size(Y2);

% Perhitungan nilai fitness
```

```
leap = 2;

ht = leap:leap:tspan(2)-leap;
n1=length(Y2(:,1));
dt=[1,1];
for i = 1 : length(Y2(:,1))
    for j = 1:length(ht)
        if Y2(i,1) < ht(j)
            dt(j) = i;
        end
    end
end
n2=length(T);
ds=[1,1];

for i = 1 : length(T)
    for j = 1:length(ht)
        if T(i,1) < ht(j)
            ds(j) = i;
        end
    end
end

 temp=0;
for i = 2 : n
    for j = 1:length(ht)
        f=Y2(dt(j),i) + (ht(j) - Y2(dt(j),1))/(Y2(dt(j)+1,1)-Y2(dt
            (j),1))*(Y2(dt(j)+1,i)-Y2(dt(j),i));
        g=Y(ds(j),i-1) + (ht(j) - T(ds(j)))/(T(ds(j)+1)-T(ds(j)))
            *(Y(ds(j)+1,i-1)-Y(ds(j),i-1));
        temp=temp+(f-g)^2;
    end
    temp=temp + (Y(end,i-1)-Y2(end,i))^2;
```

```
61 end
62
63 temp = temp*(1/(4*n));
64 F = temp;
```

iLovePDF

B.3 Model SIR

```
% Kode ini akan menjalankan genetic algorithm untuk mencari
    parameter tidak
% diketahui dari model (dalam hal ini, semua nilai mu)

clear
clc

% Parameter:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        : step size
%   n,p      : ukuran matriks populasi; jumlah kromosom x ukuran
    kromosom
%   sval     : nilai seleksi
%   cval     : nilai crossover/persilangan
%   mval     : nilai mutasi
%   snum     : jumlah kromorom yang akan mengalami seleksi
%   cnum     : jumlah kromorom yang akan mengalami crossover

t0 = 0;
t1 = 50;
tspan=[t0,t1];
x0 = [0.98 0.02 0];

p = 3;
n = 40;

sval = 0.7;
cval = 0.7;
mval = 0.7;
snum = n*sval;
```

```
cnum = n*cval;


% INITIATION : menghasilkan matriks populasi awal secara acak
%C=zeros(n,3);
%C(:,1:2)=rand(n,2);
%C(:,3)=rand(n,1);

C = rand(n,p);
J = zeros(n,1);

% EVALUATION : menghasilkan nilai fitness untuk masing-masing
    kromosom
for i = 1 : n
    fit = fitnessFunction(x0,tspan,C(i,:));
    %disp(fit);
    J(i) = 1/((1+fit));
end

%disp(C);
%disp(J);
step = 1;
%tic

[maxJ,maxJ_id] = max(J);

while max(J) < 1 - 0.0000001

    % Matriks populasi baru
    C1 = zeros(n,p);
    disp(step);
    % SELECTION :
    % 1. Sorting populasi berdasarkan nilai fitness menggunakan
```

```
62      % quicksort
63      [J,C] = quicksort(J,C);
64      %RWS
65  %       idx = RWS(J,n);
66  %       for i=1:n
67  %           C1(i,:) = C(idx(i),:);
68  %       end
69      %disp(C1);
70
71      % 2. Pilih sebanyak snum kromosom yang bertahan hidup
72      C1(snum+1:n,:) = C(snum+1:n,:);
73
74      % 3. Bangkitkan n-snum kromosom lainnya berdasarkan snum
75      % kromosom yang bertahan hidup
76      pivot = C1(n,:);
77      for i = 1 : snum
78          for j = 1 : p
79              C1(i,j) = pivot(1,j)+0.1*(rand() -0.5);
80          %C1(i,:) = rand(1,3);%C1(randi(snum),:);
81          end
82      end
83      for i = 1 : n
84          fit = fitnessFunction(x0,tspan,C1(i,:));
85          J(i) = 1/((1+fit));
86      end
87      [maxJ,maxJ_id] = max(J);
88
89      fprintf('Selection \n')
90      disp(maxJ);
91      disp(maxJ_id);
92
93      % CROSSOVER :
```

```
94    % 1. Tentukan sebanyak cnum yang akan mengalami proses
          crossover secara
95    % acak
96    %   - B1 merupakan penanda (flag) apakah kromosom tersebut
          akan
97    %   mengalami crossover atau tidak
98    %   - Jika B1(i) = 1, maka kromosom ke-i akan terpilih untuk
          crossover
99    [J,C1] = quicksort(J,C1);
100   B1 = zeros(n-3,1);
101   B1(randperm(numel(B1),cnum)) = 1;
102
103   % 2. Tentukan pasangan parent yang akan mengalami crossover
          secara acak
104   k1 = randi([10,20]);
105   p1 = C1(n,:);
106   p2 = C1(n-1,:);
107   p3 = C1(n-2,:);
108
109   % 3. Proses crossover untuk setiap pasangan parent
110   for i = 1 : k1
111       if B1(i)==1
112         C1(i,:) = offspring(p1,p2);
113       end
114   end
115   for i = k1+1 : n-3
116       if B1(i)==1
117         C1(i,:) = offspring(p1,p3);
118       end
119   end
120   for i = 1 : n
121       fit = fitnessFunction(x0,tspan,C1(i,:));
122       J(i) = 1/((1+fit));
```

```
123     end
124     [maxJ,maxJ_id] = max(J);
125
126     fprintf('Crossover \n')
127     disp(maxJ);
128     disp(maxJ_id);
129
130     % MUTATION :
131     % 1. Tentukan sebanyak mnum yang akan mengalami proses mutasi
            secara
132     % acak
133     %    - B2 merupakan penanda (flag) apakah kromosom tersebut
            akan
134     %    mengalami mutasi atau tidak
135     %    - Jika B2(i) = 1, maka kromosom ke-i akan terpilih untuk
            mutasi
136     %B2 = zeros(n,1);
137     %B2(randperm(numel(B2),mnum)) = 1;
138
139     B2 = rand(size(C1))<mval;
140
141     [maxJ,maxJ_id] = max(J);
142     for j = 1:p
143         if B2(maxJ_id,j) == 1
144             B2(maxJ_id,j) = 0;
145         end
146     end
147
148     %disp(B2);
149     % 2. Proses mutasi untuk setiap kromosom terpilih
150     %    - Untuk setiap gen, pilih suatu angka acak b, jika b<mval
            maka
151     %    ubah nilai gen
```

```
152     for i = 1 : n
153         %if B2(i) == 1
154             for j = 1 : p
155                 %flag = 1;
156                 %while flag == 1
157                 %    C1(i,j) = abs(C1(i,j) + 0.1*(rand() - 0.5));
158                 %    flag = any([C1(i,j)<0,C1(i,j)>1]);
159                 %end
160                 if B2(i,j) == 1
161                     C1(i,j) = abs(C1(i,j)+0.4*(rand()-0.5));
162                     if C1(i,j) >= 1
163                         C1(i,j) = C1(i,j) - 1;
164                     end
165                 end
166             end
167         %end
168     end
169     fprintf('Mutation \n')
170
171     C = C1;
172     for i = 1 : n
173         fit = fitnessFunction(x0,tspan,C(i,:));
174         J(i) = 1/((1+fit));
175     end
176     [maxJ,maxJ_id] = max(J);
177     disp(maxJ);
178     disp(maxJ_id);
179     %fprintf('%d %f \n',step,maxJ )
180     step = step + 1;
181
182     % REPETITION : lakukan semua proses di atas untuk setiap
        generasi baru,
```

```
183     % hingga terdapat satu kromosom yang memenuhi J(i) < epsilon =
            0.01
184 end
185 %toc
186
187 % Output dari sistem
188 result = C(maxJ_id,:)
```

B.4 Fungsi *Fitness* Model SIR

```
function F = fitnessFunction(history,tspan,param)
%function J = fitnessFunction(t0,t1,x0,h,param)
% Fungsi ini menghasilkan nilai fitness dari kromosom yang
    dimasukkan
% menggunakan fungsi error.
% Input:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        : step size
%   param    : kromosom yang akan dicari nilai fitnessnya (dalam
    hai ini,
%              semua nilai mu)
% Output:
%   J        : nilai fitness dari kromosom

% Dummy data

Y1=readtable('SIRDelayTspan50.txt');
Y2=table2array(Y1);
beta=param(1);
gamma=param(2);
r=param(3)*30;
% Solusi numerik yang dihasilkan berdasarkan nilai kromosom
opts = ddeset('RelTol',1e-5,'AbsTol',1e-8);
%[T,Y] = dde23('SIRmodelDelay',r,x0,tspan,opts,beta,gamma);
sol1 = dde23('SIRmodelDelay',r,history,tspan,opts,beta,gamma);
Y=sol1.y';
T=sol1.x';

[m n] = size(Y2);

```

```
% Perhitungan nilai fitness
leap = 2;

ht = leap:leap:tspan(2)-leap;
n1=length(Y2(:,1));
dt=[1,1,1];
for i = 1 : length(Y2(:,1))
for j = 1:length(ht)
        if Y2(i,1) < ht(j)
            dt(j) = i;
        end
    end
end

n2=length(T);
ds=[1,1,1];
for i = 1 : length(T)

for j = 1:length(ht)
        if T(i,1) < ht(j)
            ds(j) = i;
        end
    end
end

 temp=0;
for i = 2 : n
    for j = 1 :length(ht)
        f=Y2(dt(j),i) + (ht(j) - Y2(dt(j),1))/(Y2(dt(j)+1,1)-Y2(dt
            (j),1))*(Y2(dt(j)+1,i)-Y2(dt(j),i));
        g=Y(ds(j),i-1) + (ht(j) - T(ds(j)))/(T(ds(j)+1)-T(ds(j)))
            *(Y(ds(j)+1,i-1)-Y(ds(j),i-1));
        temp=temp+(f-g)^2;
```

```
61        end
62        temp=temp + (Y(end,i-1)-Y2(end,i))^2;
63    end
64
65
66    temp = temp*(1/(4*n));
67    F = temp;
```

UNIVERSITAS PELITA HARAPAN
iLovePDF

B.5 Model SIS

```
% Kode ini akan menjalankan genetic algorithm untuk mencari
    parameter tidak
% diketahui dari model (dalam hal ini, semua nilai mu)

clear
clc

% Parameter:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        : step size
%   n,p      : ukuran matriks populasi; jumlah kromosom x ukuran
    kromosom
%   sval     : nilai seleksi
%   cval     : nilai crossover/persilangan
%   mval     : nilai mutasi
%   snum     : jumlah kromorom yang akan mengalami seleksi
%   cnum     : jumlah kromorom yang akan mengalami crossover

t0 = 0;
t1 = 50;
tspan=[t0,t1];
x0 = [0.98 0.02];

p = 3;
n = 40;

sval = 0.7;
cval = 0.7;
mval = 0.7;
snum = n*sval;
```

```
cnum = n*cval;

% INITIATION : menghasilkan matriks populasi awal secara acak
%C=zeros(n,3);
%C(:,1:2)=rand(n,2);
%C(:,3)=rand(n,1);

C = rand(n,p);
J = zeros(n,1);

% EVALUATION : menghasilkan nilai fitness untuk masing-masing
    kromosom
for i = 1 : n
    fit = fitnessFunctionSIS(x0,tspan,C(i,:));
    %disp(fit);
    J(i) = 1/((1+fit));
end

%disp(C);
%disp(J);
step = 1;
%tic

[maxJ,maxJ_id] = max(J);

while max(J) < 1 - 0.0000001

    % Matriks populasi baru
    C1 = zeros(n,p);
    disp(step);
    % SELECTION :
    % 1. Sorting populasi berdasarkan nilai fitness menggunakan
    % quicksort
```

```
62      [J,C] = quicksort(J,C);
63      %RWS
64  %       idx = RWS(J,n);
65  %       for i=1:n
66  %           C1(i,:) = C(idx(i),:);
67  %       end
68      %disp(C1);
69
70      % 2. Pilih sebanyak snum kromosom yang bertahan hidup
71      C1(snum+1:n,:) = C(snum+1:n,:);
72
73      % 3. Bangkitkan n-snum kromosom lainnya berdasarkan snum
74      % kromosom yang bertahan hidup
75      pivot = C1(n,:);
76      for i = 1 : snum
77          for j = 1 : p
78              C1(i,j) = pivot(1,j)+0.1*(rand() -0.5);
79          %C1(i,:) = rand(1,3);%C1(randi(snum),:);
80          end
81      end
82      for i = 1 : n
83          fit = fitnessFunctionSIS(x0,tspan,C1(i,:));
84          J(i) = 1/((1+fit));
85      end
86      [maxJ,maxJ_id] = max(J);
87
88      fprintf('Selection \n')
89      disp(maxJ);
90      disp(maxJ_id);
91
92      % CROSSOVER :
93      % 1. Tentukan sebanyak cnum yang akan mengalami proses
            crossover secara
```

```
94      % acak
95      %   - B1 merupakan penanda (flag) apakah kromosom tersebut
            akan
96      %   mengalami crossover atau tidak
97      %   - Jika B1(i) = 1, maka kromosom ke-i akan terpilih untuk
            crossover
98      [J,C1] = quicksort(J,C1);
99      B1 = zeros(n-3,1);
100     B1(randperm(numel(B1),cnum)) = 1;
101
102     % 2. Tentukan pasangan parent yang akan mengalami crossover
            secara acak
103     k1 = randi([10,20]);
104     p1 = C1(n,:);
105     p2 = C1(n-1,:);
106     p3 = C1(n-2,:);
107
108     % 3. Proses crossover untuk setiap pasangan parent
109     for i = 1 : k1
110         if B1(i)==1
111           C1(i,:) = offspring(p1,p2);
112         end
113     end
114     for i = k1+1 : n-3
115         if B1(i)==1
116           C1(i,:) = offspring(p1,p3);
117         end
118     end
119     for i = 1 : n
120         fit = fitnessFunctionSIS(x0,tspan,C1(i,:));
121         J(i) = 1/((1+fit));
122     end
123     [maxJ,maxJ_id] = max(J);
```

```
124
125     fprintf('Crossover \n')
126     disp(maxJ);
127     disp(maxJ_id);
128
129     % MUTATION :
130     % 1. Tentukan sebanyak mnum yang akan mengalami proses mutasi
            secara
131     % acak
132     %    - B2 merupakan penanda (flag) apakah kromosom tersebut
            akan
133     %    mengalami mutasi atau tidak
134     %    - Jika B2(i) = 1, maka kromosom ke-i akan terpilih untuk
            mutasi
135     %B2 = zeros(n,1);
136     %B2(randperm(numel(B2),mnum)) = 1;
137
138     B2 = rand(size(C1))<mval;
139
140     [maxJ,maxJ_id] = max(J);
141     for j = 1:p
142         if B2(maxJ_id,j) == 1
143             B2(maxJ_id,j) = 0;
144         end
145     end
146
147     %disp(B2);
148     % 2. Proses mutasi untuk setiap kromosom terpilih
149     %    - Untuk setiap gen, pilih suatu angka acak b, jika b<mval
            maka
150     %    ubah nilai gen
151     for i = 1 : n
152         %if B2(i) == 1
```

```
153                 for j = 1 : p
154                     %flag = 1;
155                     %while flag == 1
156                     %    C1(i,j) = abs(C1(i,j) + 0.1*(rand() - 0.5));
157                     %    flag = any([C1(i,j)<0,C1(i,j)>1]);
158                     %end
159                     if B2(i,j) == 1
160                         C1(i,j) = abs(C1(i,j)+0.4*(rand()-0.5));
161                         if C1(i,j) >= 1
162                             C1(i,j) = C1(i,j) - 1;
163                         end
164                     end
165                 end
166             %end
167         end
168         fprintf('Mutation \n')
169
170         C = C1;
171         for i = 1 : n
172             fit = fitnessFunctionSIS(x0,tspan,C(i,:));
173             J(i) = 1/((1+fit));
174         end
175         [maxJ,maxJ_id] = max(J);
176         disp(maxJ);
177         disp(maxJ_id);
178         %fprintf('%d %f \n',step,maxJ )
179         step = step + 1;
180
181         % REPETITION : lakukan semua proses di atas untuk setiap
                generasi baru,
182         % hingga terdapat satu kromosom yang memenuhi J(i) < epsilon =
                0.01
183 end
```

```
184 %toc
185
186 % Output dari sistem
187 result = C(maxJ_id,:)
```

iLovePDF

B.6 Fungsi *Fitness* Model SIS

```
function F = fitnessFunctionSIS(history,tspan,param)
%function J = fitnessFunction(t0,t1,x0,h,param)
% Fungsi ini menghasilkan nilai fitness dari kromosom yang
    dimasukkan
% menggunakan fungsi error.
% Input:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        : step size
%   param    : kromosom yang akan dicari nilai fitnessnya (dalam
    hai ini,
%              semua nilai mu)
% Output:
%   J        : nilai fitness dari kromosom

% Dummy data
%Y1 = importdata('SIRDelay2.txt');
Y1=readtable('SISDelayTspan50.txt');
Y2=table2array(Y1);
beta=param(1);
gamma=param(2);
r=param(3)*30;
% Solusi numerik yang dihasilkan berdasarkan nilai kromosom
opts = ddeset('RelTol',1e-5,'AbsTol',1e-8);
%[T,Y] = dde23('SIRmodelDelay',r,x0,tspan,opts,beta,gamma);
sol1 = dde23('SISmodelDelay',r,history,tspan,opts,beta,gamma);
Y=sol1.y';
T=sol1.x';
% [T,Y] = RK4_sys(@(t,x,mu)sysOfOdes(t,x, mu),t0,t1,x0,h, param);

[m n] = size(Y2);
```

```
30
31 % Perhitungan nilai fitness
32 leap = 2;
33
34 ht = leap:leap:tspan(2)-leap;
35 n1=length(Y2(:,1));
36 dt=[1,1];
37 for i = 1 : length(Y2(:,1))
38     for j = 1:length(ht)
39         if Y2(i,1) < ht(j)
40             dt(j) = i;
41         end
42     end
43 end
44 n2=length(T);
45 ds=[1,1];
46 for i = 1 : length(T)
47     for j = 1:length(ht)
48         if T(i,1) < ht(j)
49             ds(j) = i;
50         end
51     end
52 end
53
54  temp=0;
55 for i = 2 : n
56     for j = 1:length(ht)
57         f=Y2(dt(j),i) + (ht(j) - Y2(dt(j),1))/(Y2(dt(j)+1,1)-Y2(dt
               (j),1))*(Y2(dt(j)+1,i)-Y2(dt(j),i));
58         g=Y(ds(j),i-1) + (ht(j) - T(ds(j)))/(T(ds(j)+1)-T(ds(j)))
               *(Y(ds(j)+1,i-1)-Y(ds(j),i-1));
59         temp=temp+(f-g)^2;
60     end
```

```
61      temp=temp + (Y(end,i-1)-Y2(end,i))^2;
62  end
63  %for i = 1 : m
64      %for j = 1 : n
65          %temp = temp + (Y(i,j) - Y1(i,j))^2;
66      %end
67  %end
68
69  temp = temp*(1/(4*n));
70  F = temp;
```

iLovePDF

B.7 Model SIRS

```
% Kode ini akan menjalankan genetic algorithm untuk mencari parameter tidak
% diketahui dari model (dalam hal ini, semua nilai mu)

clear
clc

% Parameter:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        :  step size
%   n,p      : ukuran matriks populasi; jumlah kromosom x ukuran kromosom
%   sval     : nilai seleksi
%   cval     : nilai crossover/persilangan
%   mval     : nilai mutasi
%   snum     : jumlah kromorom yang akan mengalami seleksi
%   cnum     : jumlah kromorom yang akan mengalami crossover
%   mnum     : jumlah kromorom yang akan mengalami mutasi

t0 = 0;
t1 = 50;
tspan=[t0,t1];
x0 = [0.98 0.02 0];

p = 4;
n = 40;

sval = 0.7;
cval = 0.7;
mval = 0.7;
```

```
snum = n*sval;
cnum = n*cval;
%mnum = n*mval;

% INITIATION : menghasilkan matriks populasi awal secara acak
%C=zeros(n,3);
%C(:,1:2)=rand(n,2);
%C(:,3)=rand(n,1);

C = rand(n,p);
J = zeros(n,1);

% EVALUATION : menghasilkan nilai fitness untuk masing-masing
    kromosom
for i = 1 : n
    fit = fitnessFunctionSIRS(x0,tspan,C(i,:));
    %disp(fit);
    J(i) = 1/((1+fit));
end

%disp(C);
%disp(J);
step = 1;
%tic

[maxJ,maxJ_id] = max(J);

while max(J) < 1 - 0.0000001

    % Matriks populasi baru
    C1 = zeros(n,p);
    disp(step);
    % SELECTION :
```

```
62      % 1. Sorting populasi berdasarkan nilai fitness menggunakan
63      % quicksort
64      [J,C] = quicksort(J,C);
65      %RWS
66  %       idx = RWS(J,n);
67  %       for i=1:n
68  %           C1(i,:) = C(idx(i),:);
69  %       end
70      %disp(C1);
71
72      % 2. Pilih sebanyak snum kromosom yang bertahan hidup
73      C1(snum+1:n,:) = C(snum+1:n,:);
74
75      % 3. Bangkitkan n-snum kromosom lainnya berdasarkan snum
76      % kromosom yang bertahan hidup
77      pivot = C1(n,:);
78      for i = 1 : snum
79          for j = 1 : p
80              C1(i,j) = pivot(1,j)+0.1*(rand() -0.5);
81          %C1(i,:) = rand(1,3);%C1(randi(snum),:);
82          end
83      end
84      for i = 1 : n
85          fit = fitnessFunctionSIRS(x0,tspan,C1(i,:));
86          J(i) = 1/((1+fit));
87      end
88      [maxJ,maxJ_id] = max(J);
89
90      fprintf('Selection \n')
91      disp(maxJ);
92      disp(maxJ_id);
93
94      % CROSSOVER :
```

```
95      % 1. Tentukan sebanyak cnum yang akan mengalami proses
            crossover secara
96      % acak
97      %   - B1 merupakan penanda (flag) apakah kromosom tersebut
            akan
98      %   mengalami crossover atau tidak
99      %   - Jika B1(i) = 1, maka kromosom ke-i akan terpilih untuk
            crossover
100     [J,C1] = quicksort(J,C1);
101     B1 = zeros(n-3,1);
102     B1(randperm(numel(B1),cnum)) = 1;
103
104     % 2. Tentukan pasangan parent yang akan mengalami crossover
            secara acak
105     k1 = randi([10,20]);
106     p1 = C1(n,:);
107     p2 = C1(n-1,:);
108     p3 = C1(n-2,:);
109
110     % 3. Proses crossover untuk setiap pasangan parent
111     for i = 1 : k1
112         if B1(i)==1
113           C1(i,:) = offspring(p1,p2);
114         end
115     end
116     for i = k1+1 : n-3
117         if B1(i)==1
118           C1(i,:) = offspring(p1,p3);
119         end
120     end
121     for i = 1 : n
122         fit = fitnessFunctionSIRS(x0,tspan,C1(i,:));
123         J(i) = 1/((1+fit));
```

```
124     end
125     [maxJ,maxJ_id] = max(J);
126
127     fprintf('Crossover \n')
128     disp(maxJ);
129     disp(maxJ_id);
130
131     % MUTATION :
132     % 1. Tentukan sebanyak mnum yang akan mengalami proses mutasi
            secara
133     % acak
134     %    - B2 merupakan penanda (flag) apakah kromosom tersebut
            akan
135     %    mengalami mutasi atau tidak
136     %    - Jika B2(i) = 1, maka kromosom ke-i akan terpilih untuk
            mutasi
137     %B2 = zeros(n,1);
138     %B2(randperm(numel(B2),mnum)) = 1;
139
140     B2 = rand(size(C1))<mval;
141
142     [maxJ,maxJ_id] = max(J);
143     for j = 1:p
144         if B2(maxJ_id,j) == 1
145             B2(maxJ_id,j) = 0;
146         end
147     end
148
149     %disp(B2);
150     % 2. Proses mutasi untuk setiap kromosom terpilih
151     %    - Untuk setiap gen, pilih suatu angka acak b, jika b<mval
            maka
152     %    ubah nilai gen
```

```
153     for i = 1 : n
154         %if B2(i) == 1
155             for j = 1 : p
156                 %flag = 1;
157                 %while flag == 1
158                 %    C1(i,j) = abs(C1(i,j) + 0.1*(rand() - 0.5));
159                 %    flag = any([C1(i,j)<0,C1(i,j)>1]);
160                 %end
161                 if B2(i,j) == 1
162                     C1(i,j) = abs(C1(i,j)+0.4*(rand()-0.5));
163                     if C1(i,j) >= 1
164                         C1(i,j) = C1(i,j) - 1;
165                     end
166                 end
167             end
168         %end
169     end
170     fprintf('Mutation \n')
171
172     C = C1;
173     for i = 1 : n
174         fit = fitnessFunctionSIRS(x0,tspan,C(i,:));
175         J(i) = 1/((1+fit));
176     end
177     [maxJ,maxJ_id] = max(J);
178     disp(maxJ);
179     disp(maxJ_id);
180     %fprintf('%d %f \n',step,maxJ )
181     step = step + 1;
182
183     % REPETITION : lakukan semua proses di atas untuk setiap
        generasi baru,
```

```
184     % hingga terdapat satu kromosom yang memenuhi J(i) < epsilon =
            0.01
185 end
186 %toc
187
188 % Output dari sistem
189 result = C(maxJ_id,:)
```

iLovePDF

B.8 Fungsi *Fitness* Model SIRS

```
function F = fitnessFunctionSIRS(history,tspan,param)
%function J = fitnessFunction(t0,t1,x0,h,param)
% Fungsi ini menghasilkan nilai fitness dari kromosom yang
    dimasukkan
% menggunakan fungsi error.
% Input:
%   t0,t1    : waktu sebagai syarat batas
%   x0       : jumlah populasi awal sebagai syarat awal
%   h        : step size
%   param    : kromosom yang akan dicari nilai fitnessnya (dalam
    hai ini,
%              semua nilai mu)
% Output:
%   J        : nilai fitness dari kromosom

% Dummy data

Y1=readtable('SIRSDelayTspan50.txt');
Y2=table2array(Y1);
beta=param(1);
gamma=param(2);
alpha=param(3);
r=param(4)*30;
% Solusi numerik yang dihasilkan berdasarkan nilai kromosom
opts = ddeset('RelTol',1e-5,'AbsTol',1e-8);
%[T,Y] = dde23('SIRmodelDelay',r,x0,tspan,opts,beta,gamma);
sol1 = dde23('SIRSmodelDelay',r,history,tspan,opts,beta,gamma,
    alpha);
Y=sol1.y';
T=sol1.x';

```

```
[m n] = size(Y2);

% Perhitungan nilai fitness
leap = 2;

ht = leap:leap:tspan(2)-leap;
n1=length(Y2(:,1));
dt=[1,1,1,1];
for i = 1 : length(Y2(:,1))
for j = 1:length(ht)
        if Y2(i,1) < ht(j)
            dt(j) = i;
        end
    end
end
n2=length(T);
ds=[1,1,1,1];
for i = 1 : length(T)
for j = 1:length(ht)
        if T(i,1) < ht(j)
            ds(j) = i;
        end
    end
end

 temp=0;
for i = 2 : n
    for j = 1 :length(ht)
        f=Y2(dt(j),i) + (ht(j) - Y2(dt(j),1))/(Y2(dt(j)+1,1)-Y2(dt
            (j),1))*(Y2(dt(j)+1,i)-Y2(dt(j),i));
        g=Y(ds(j),i-1) + (ht(j) - T(ds(j)))/(T(ds(j)+1)-T(ds(j)))
            *(Y(ds(j)+1,i-1)-Y(ds(j),i-1));
        temp=temp+(f-g)^2;
```

```
60        end
61        temp=temp + (Y(end,i-1)-Y2(end,i))^2;
62    end
63    %for i = 1 : m
64        %for j = 1 : n
65            %temp = temp + (Y(i,j) - Y1(i,j))^2;
66        %end
67    %end
68
69    temp = temp*(1/(4*n));
70    F = temp;
```

UNIVERSITAS HARAPAN
iLovePDF