Modellierung und Simulation **Übung 05**
WS 2016/17 **Abgabefrist: Siehe elearning bzw. exchange**

Name: _____ **Aufwand (h):** _____

Punkte: _____

Aufgabe 1 (8 Pkt): Epidemien – Logistisches Modell

Nehmen Sie für eine bestimmte Krankheit an, man können ihren epidemologischen Verlauf mit einem logistischen Modell beschrieben. Stellen Sie graphisch die Krankheitsverläufe für N = 10000 und k = 10, 20, 30, 50 in einem Kurvenfenster dar.

Aufgabe 2 (8 Pkt): Epidemien – SIR-Modelle

(a) Infolge Vorsichtsmaßnahmen kann die Basisreproduktionszahl bei einer Masernepidemie auf R_0 = 7,5 gedrückt werden. Die durchschnittliche infektiöse Periode eines Erkrankten dauert 10 Tage. Man berechne den Epidemieverlauf mit Hilfe eines SIR-Modells bei Vernachlässigung der Geburten- und Sterberate.

(b) Man berechne den Krankheitsverlauf (für S, I und R) von Beispiel (a) unter der Annahme einer Geburtenkonstanten μ = 0,0003. Welchem Gleichgewichtswert strebt die Zahl der Erkrankten für $t \rightarrow \infty$ zu? Wie hoch ist die kritische Durchimpfungsrate p_{crit} und wie ist der Langzeitverlauf der Krankheit (I) bei Impfung Neugeborener mit dieser Rate?

Aufgabe 3 (8 Pkt): Epidemien in heterogenen Populationen

Gegeben sind drei Gruppen mit einer „core group“ durch folgende Angaben:

Gruppe 1: n = 1000, 2 Kontakte pro Woche, Anteile: 0.9, 0.05, 0.05
Gruppe 2: n = 100, 10 Kontakte pro Woche, Anteile: 0.1, 0.7, 0.2
Gruppe 3: n = 500, 4 Kontakte pro Woche, Anteile: 0.3, 0.3, 0.4
Infektionswahrscheinlichkeit bei einem Kontakt: 15%
Infektiöse Periode dauert 2 Wochen für Gruppe 1 und 2, sowie 1.5 Wochen für Gruppe3

Wie verläuft die Infektion in den einzelnen Gruppen, wenn zu Beginn 5 Krankheitsfälle in Gruppe 3 auftreten?

Hinweise: Geben Sie Ihre Ausarbeitung gedruckt auf Papier ab.
Abgegebene Beispiele müssen in der Übungsstunde präsentiert werden können.

1 Epidemien - Logistisches Modell

Listing 1: Skript für die Simulation mit einem logistischen Modell

```
% simulation arguments
N       = 1000;
k       = [10,20,30,50];
kSize   = size(k);

% Logisitc Simulation parameters 'epidemiologyLogisitc'
ltStart = 0;
ltStep  = 0.01;
ltMax   = 2;

% Simulation parameters 'epidemiologyLogisitcProg'
start = 1;
step  = 1;
max   = kSize(1,2);

% result container
results(start,max) = struct('iProg', [], 'sProg', []);

% for each paramter combination
for i=start:step:max
    % calculate progression for parameter combination
    res = epidemiologyLogisitc(ltStart, ltStep, ltMax, N, k(i));

    % plot each result in own window
    figure;
    hold on;

    title(strcat('N=1000, k=',num2str(k(i))));
    plot(ltStart:ltStep:ltMax, [res.iProg res.sProg]);
    xlabel('t')
    legend('I_t', 'S_t');

    hold off;
end
```

Listing 2: Funktion zur Berechnung des logistischen Modell

```
function result = epidemiologyLogisitc(tStart, tStep, tMax, N, k)
% Epidiomology with logisitc model

    % Simulation arguments
    beta  = k/N;
    alpha = k;
    i     = 1;      % start one injected, otherwise desease cannot spread
    idx   = 1;      % matrix index

    % result matrizes
    result    = [];                     % result container
    sProgress = zeros(tMax/tStep+1,1); % susceptible
    iProgress = zeros(tMax/tStep+1,1); % infected

    for t=tStart:tStep:tMax
        % calculate deviation
        i_ = (i * alpha) - (i^2 * beta);

        % calculate current infected
        i  = i + i_ * tStep;
```

```
21
22          % keep current results
23          iProgress(idx) = i;
24          sProgress(idx) = N - i;
25
26          % inc idx counter
27          idx = idx + 1;
28      end
29
30      % set returned results
31      result.iProg = iProgress;
32      result.sProg = sProgress;
33
34  end
```

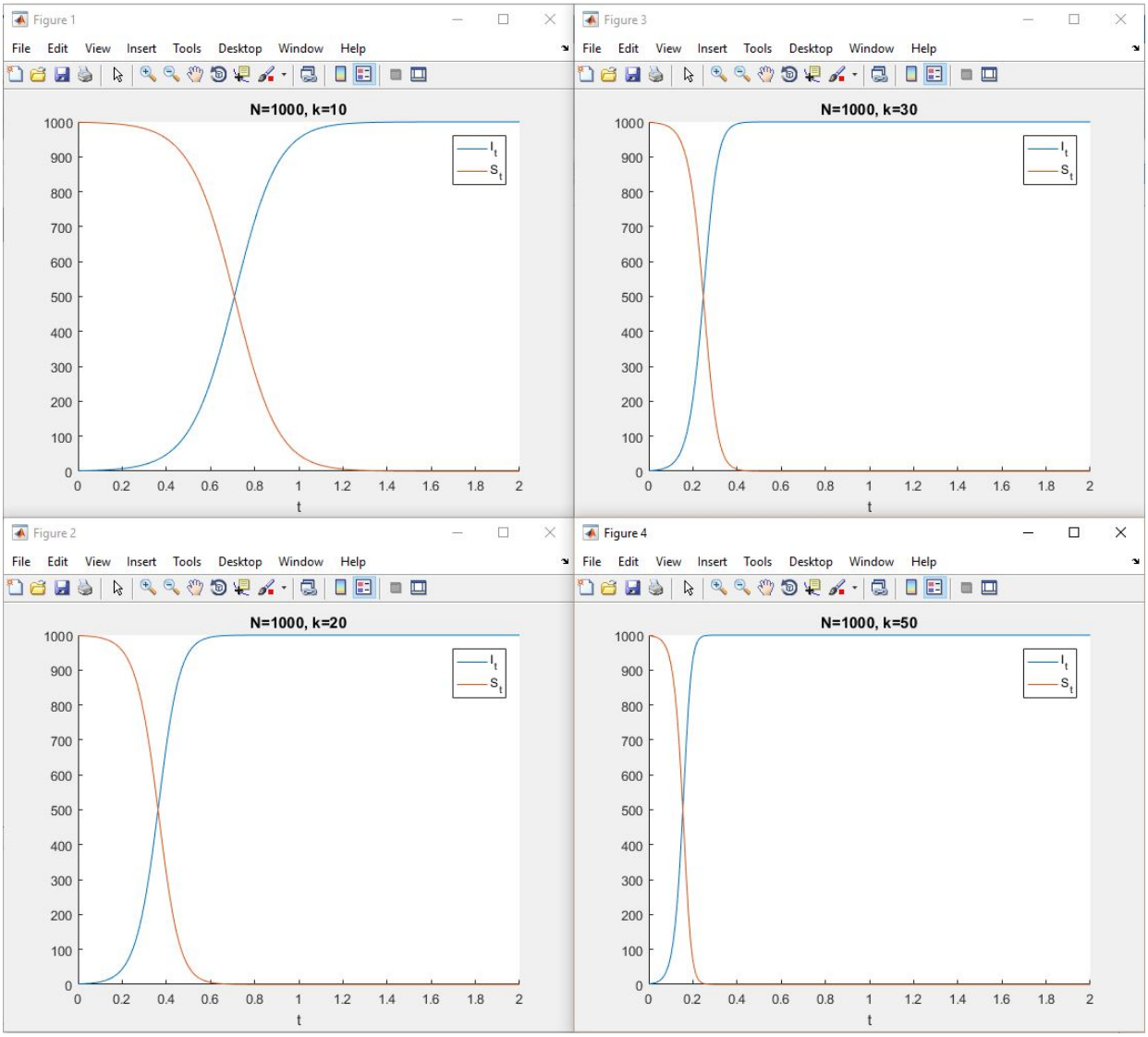

Abbildung 1: Verlauf der Epidemie mit einem logistischen Modell

Je kleiner k ist, desto länger braucht die Epidemie zum Ausbrechen und desto länger dauert die Epidemie an, wobei der Verlauf der Epidemie gegenüber einem größeren k flacher ist. Bei einem größeren k bricht die Epidemie schneller aus und hat einen steileren Verlauf.

2 Epidemien - SIR-Modell

Listing 3: Skript für die Simulation mit einem SIR-Modell

```
% simulation arguments
N       = 10000;
R0      = 7.5;              % Ro = alpha/beta
beta    = 1/10;             % one healing per 10-days
mu      = 0.0003;           % bearth rate
alpha_a = R0 * beta;        % alpha for a)
alpha_b = R0 * (beta + mu); % alpha for b)
iStart  = 1;                % start of infected
p_crit  = 1 - (1/R0);       % p_crit

% Logisitc Simulation parameters 'epidemiologySIR'
tStart  = 0;
tStep   = 0.001;
tMax    = 100;
tMaxMuP = 10000;

% with no mu and p_crit
resA = epidemiologySIR(tStart, tStep, tMaxMuP, alpha_a, beta, iStart, N);
% with mu and no p_crit
resB = epidemiologySIR(tStart, tStep, tMaxMuP, alpha_b, beta, iStart, N, mu);
% with mu and p_crit
resC = epidemiologySIR(tStart, tStep, tMaxMuP, alpha_b, beta, iStart, N, mu, p_crit);

% with no mu and p_cri
plotSir(resA, 'a) ', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_a), ...
                     num2str(beta), 'undefined', 'undefined');
% with mu and no p_crit
plotSir(resB, 'b.1) ', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_b), ...
                     num2str(beta), num2str(mu), 'undefined');
% with mu and p_crit
plotSir(resB, 'b.2) ', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_b), ...
                     num2str(beta), num2str(mu), num2str(p_crit));
% with mu and p_crit (zoom for equilibrium)
plotSir(resB, 'b.2.1) ', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_b), ...
                     num2str(beta), num2str(mu), num2str(p_crit));
```

Listing 4: Funktion zur Berechnung des SIR-Modell

```
function result = epidemiologySIR(tStart, tStep, tMax, alpha, beta, iStart, N, mu, p)
% Does not work properly, don't know why yet !!!!!

    % simulation arguments
    i = iStart;
    r = 0;
    s = N - i - r;

    sProgress = zeros(tMax/tStep+1,1);
    iProgress = zeros(tMax/tStep+1,1);
    rProgress = zeros(tMax/tStep+1,1);
    j = 1;

    for t=tStart:tStep:tMax
        %                                       birth rate
        s_ = -alpha * s * (i / N);
        i_ =  alpha * s * (i / N) - (beta * i);
        r_ =                        (beta * i);

```

```
20          % with birth rate
21          if nargin == 8
22              s_ = s_ + (mu * N);
23              r_ = r_ + (mu * N);
24          end
25          if nargin == 9
26              s_ = s_ + (mu * N * (1 - p));
27              r_ = r_ + (mu * N * p);
28          end
29
30          s = s + s_ * tStep;
31          i = i + i_ * tStep;
32          r = r + r_ * tStep;
33
34          sProgress(j) = s;
35          iProgress(j) = i;
36          rProgress(j) = r;
37
38          j = j + 1;
39
40      end
41
42      result.iProg = iProgress;
43      result.sProg = sProgress;
44      result.rProg = rProgress;
45  end
```

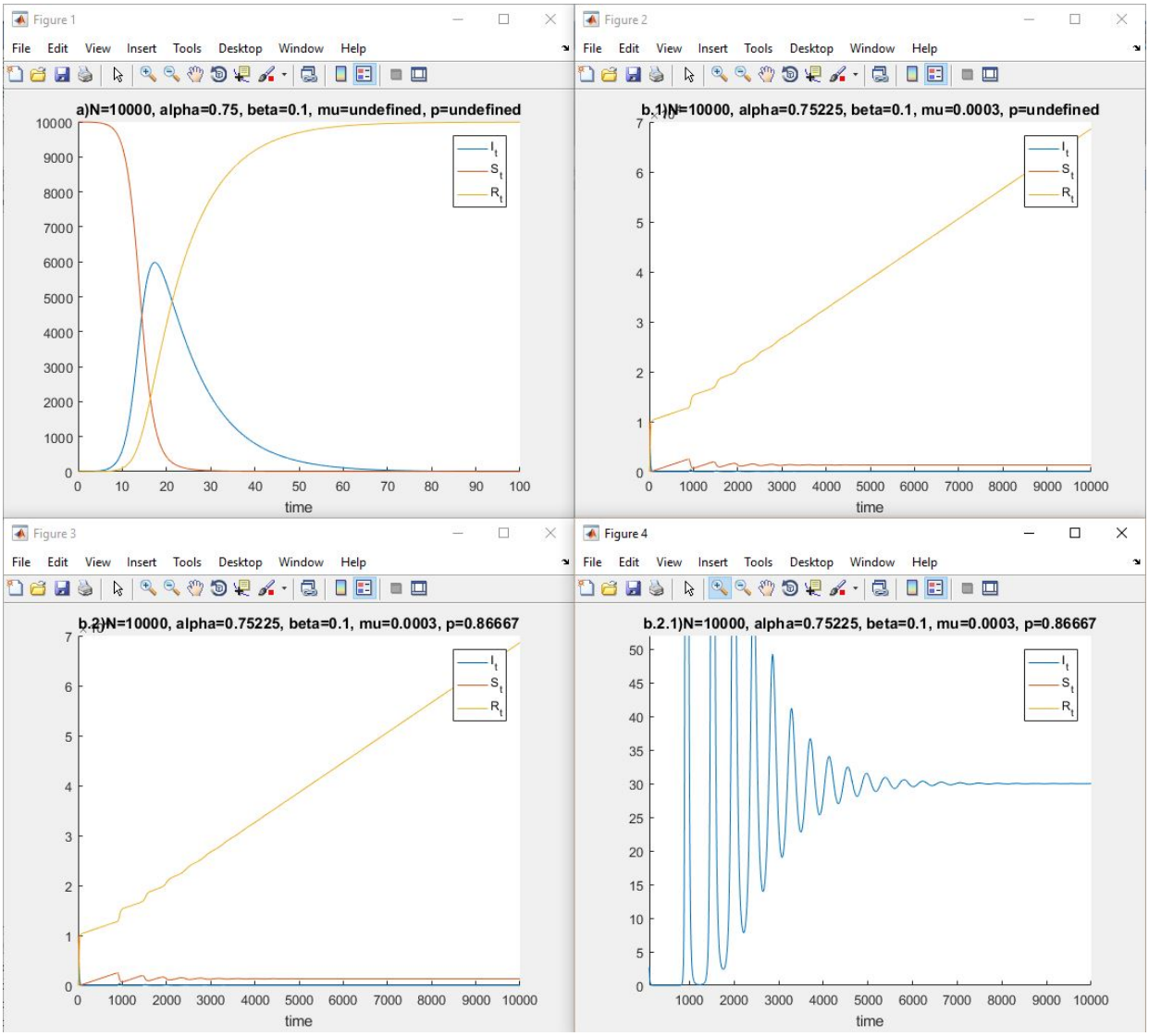

Abbildung 2: Verlauf der Epidemie mit SIR-Modell

Der erste Ausbruch der Epidemie dauert ca. 60 Tage. Wenn eine Geburtenrate und p_crit miteinbezogen werden, dann sieht man dass sich die Anzahl der infizierten Personen ab ca. 6000 Tagen ungefähr bei 30 Personen einpendelt (Equilibrium).

3 Epidemien in heterogenen Populationen

Listing 5: Skript für die Simulation des SIR-Modells mit heterogenen Populationen.

```
% simulation parameter
config        = [];
config.tStart = 0;
config.tStep  = 0.01;
config.tMax   = 365;   % in days

% simulation arguments (given parameters were normalized to days)
groups(1) = struct('N',     1000,   ...
                   'k',     (2/7),  ...
                   'iStart', 0,     ...
                   'alpha',  0.15,  ...
                   'beta',   (1/14), ...
                   'm',      [0.9 0.05 0.05]);

groups(2) = struct('N',     100,    ...
                   'k',     (10/7), ...
                   'iStart', 0,     ...
                   'alpha',  0.15,  ...
                   'beta',   (1/14), ...
                   'm',      [0.1 0.7  0.2]);

groups(3) = struct('N',     500,    ...
                   'k',     (4/7),  ...
                   'iStart', 5,     ...
                   'alpha',  0.15,  ...
                   'beta',   (1/(7 + (7/2))), ...
                   'm',      [0.3 0.3  0.4]);

result = epidemiologySIRGroups(config, groups);

for i=1:1:3
    curGroup = groups(i);
    plotSir(result(i), ...
            strcat('Group-', strcat(num2str(i), ', ')), ...
            (config.tStart:config.tStep:config.tMax), ...
            num2str(curGroup.N), ...
            num2str(curGroup.alpha), ...
            num2str(curGroup.beta),  ...
            'unused', ...
            'unused');
end
```

Listing 6: Funktion zur Berechnung des SIR-Modell mit heterogenen Populationen

```
function result = epidemiologySIRGroups(config, groups)
% calculates the epidemiology for n given groups via an SIR model

    % simulation arguments
    groupSize  = size(groups);
    dataCount = groupSize(1,2);
    idx       = 1;
    runCont(1:dataCount) = struct('i', 0, 'r', 0, 's', 0);

    % initialize run container with start values
    for j=1:1:dataCount
        curData = groups(j);
        runCont(j) = struct('s', (curData.N - curData.iStart), ...
```

```
                            'i', curData.iStart, ...
                            'r', 0);
    end

    % initialize result container
    result(1:dataCount) = struct('sProg', zeros(config.tMax/config.tStep+1,1), ...
                                 'iProg', zeros(config.tMax/config.tStep+1,1), ...
                                 'rProg', zeros(config.tMax/config.tStep+1,1));

    % main loop for time
    for t=config.tStart:config.tStep:config.tMax

        % loop for given data groups
        for j=1:1:dataCount
            curRun  = runCont(j);  % holds last current results or start values at begin
            curData = groups(j);     % the current group to calculate

            % old current values
            s     = curRun.s;
            i     = curRun.i;
            r     = curRun.r;
            sum   = 0;
            alpha = curData.alpha;
            beta  = curData.beta;

            % calcuate group-wide infections for current group
            for k=1:1:dataCount
                sData = groups(k);
                sum = sum + ((sData.k * sData.m(j)) * (runCont(k).i/sData.N));
            end

            % calculate deviation
            s_ = (-alpha * sum * s);
            i_ = (alpha  * sum * s) - (beta * i);
            r_ =                      (beta * i);

            % set new current values for next iteration
            runCont(j) = struct('s', (s + (s_ * config.tStep)), ...
                                'i', (i + (i_ * config.tStep)), ...
                                'r', (r + (r_ * config.tStep)));

            % remember results
            result(j).sProg(idx) = runCont(j).s;
            result(j).iProg(idx) = runCont(j).i;
            result(j).rProg(idx) = runCont(j).r;

        end

        % increase group prog index
        idx = idx + 1;
    end
end
```

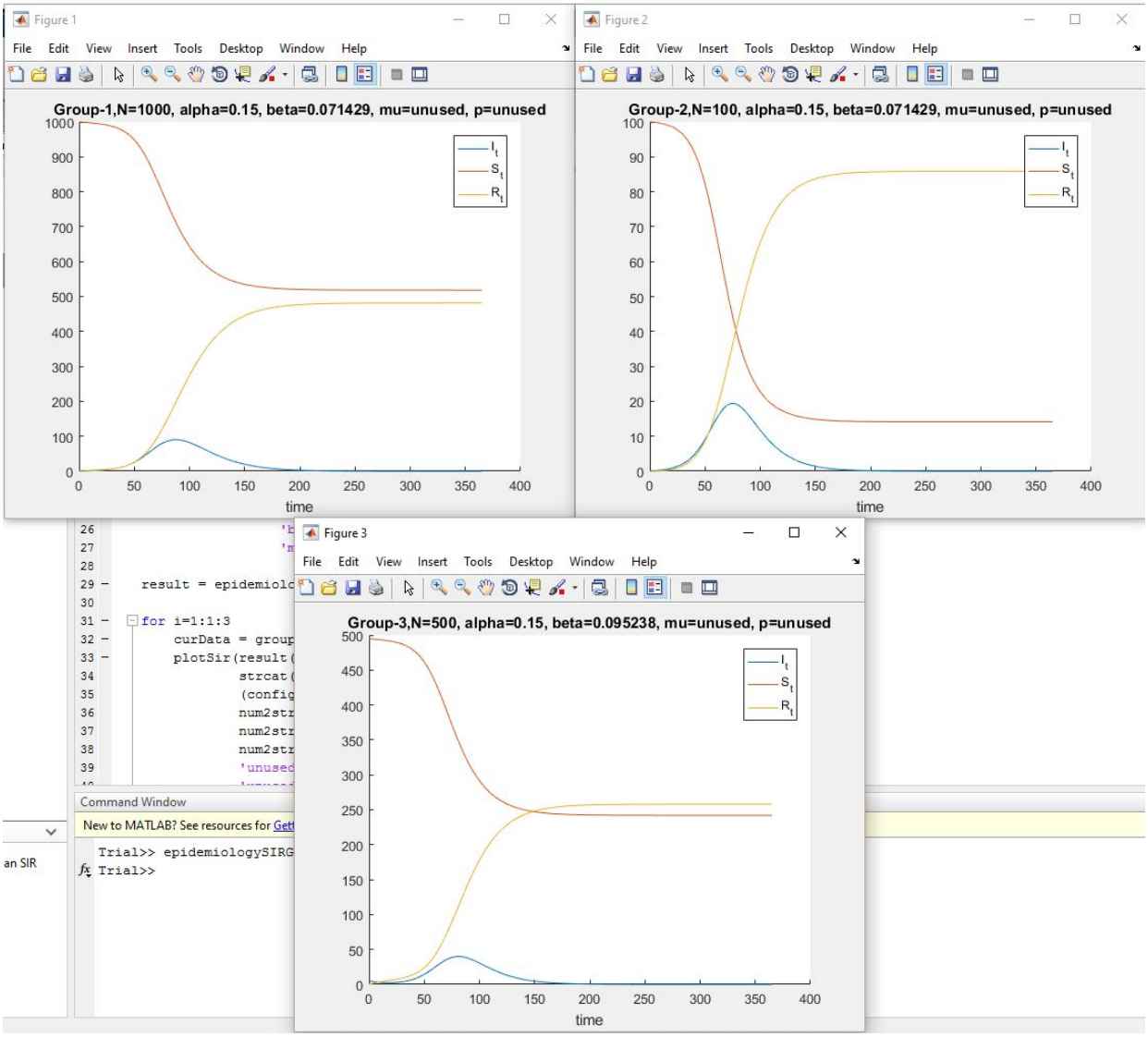

Abbildung 3: Verlauf der Epidemie in den verschiedenen Gruppen mit einem SIR-Modell