

Name: _____

Aufwand (h): _____

Punkte: _____

Aufgabe 1 (8 Pkt): Epidemien – Logistisches Modell

Nehmen Sie für eine bestimmte Krankheit an, man könne ihren epidemiologischen Verlauf mit einem logistischen Modell beschreiben. 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 $\mu = 0,0003$. 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 Gruppe 3

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

```

1 % simulation arguments
2 N = 1000;
3 k = [10,20,30,50];
4 kSize = size(k);
5
6 % Logisitic Simulation parameters 'epidemiologyLogisitic'
7 ltStart = 0;
8 ltStep = 0.01;
9 ltMax = 2;
10
11 % Simulation parameters 'epidemiologyLogisiticProg'
12 start = 1;
13 step = 1;
14 max = kSize(1,2);
15
16 % result container
17 results(start,max) = struct('iProg', [], 'sProg', []);
18
19 % for each paramter combination
20 for i=start:step:max
21     % calculate progression for parameter combination
22     res = epidemiologyLogisitic(ltStart, ltStep, ltMax, N, k(i));
23
24     % plot each result in own window
25     figure;
26     hold on;
27
28     title(strcat('N=1000, k=', num2str(k(i))));
29     plot(ltStart:ltStep:ltMax, [res.iProg res.sProg]);
30     xlabel('t')
31     legend('I_t', 'S_t');
32
33     hold off;
34 end

```

Listing 2: Funktion zur Berechnung des logistischen Modell

```

1 function result = epidemiologyLogisitic(tStart, tStep, tMax, N, k)
2 % Epidiomology with logisitic model
3
4     % Simulation arguments
5     beta = k/N;
6     alpha = k;
7     i = 1; % start one injected, otherwise desease cannot spread
8     idx = 1; % matrix index
9
10    % result matrizes
11    result = []; % result container
12    sProgress = zeros(tMax/tStep+1,1); % susceptible
13    iProgress = zeros(tMax/tStep+1,1); % infected
14
15    for t=tStart:tStep:tMax
16        % calculate deviation
17        i_ = (i * alpha) - (i^2 * beta);
18
19        % calculate current infected
20        i = i + i_ * tStep;

```

Übung 1

```

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

```

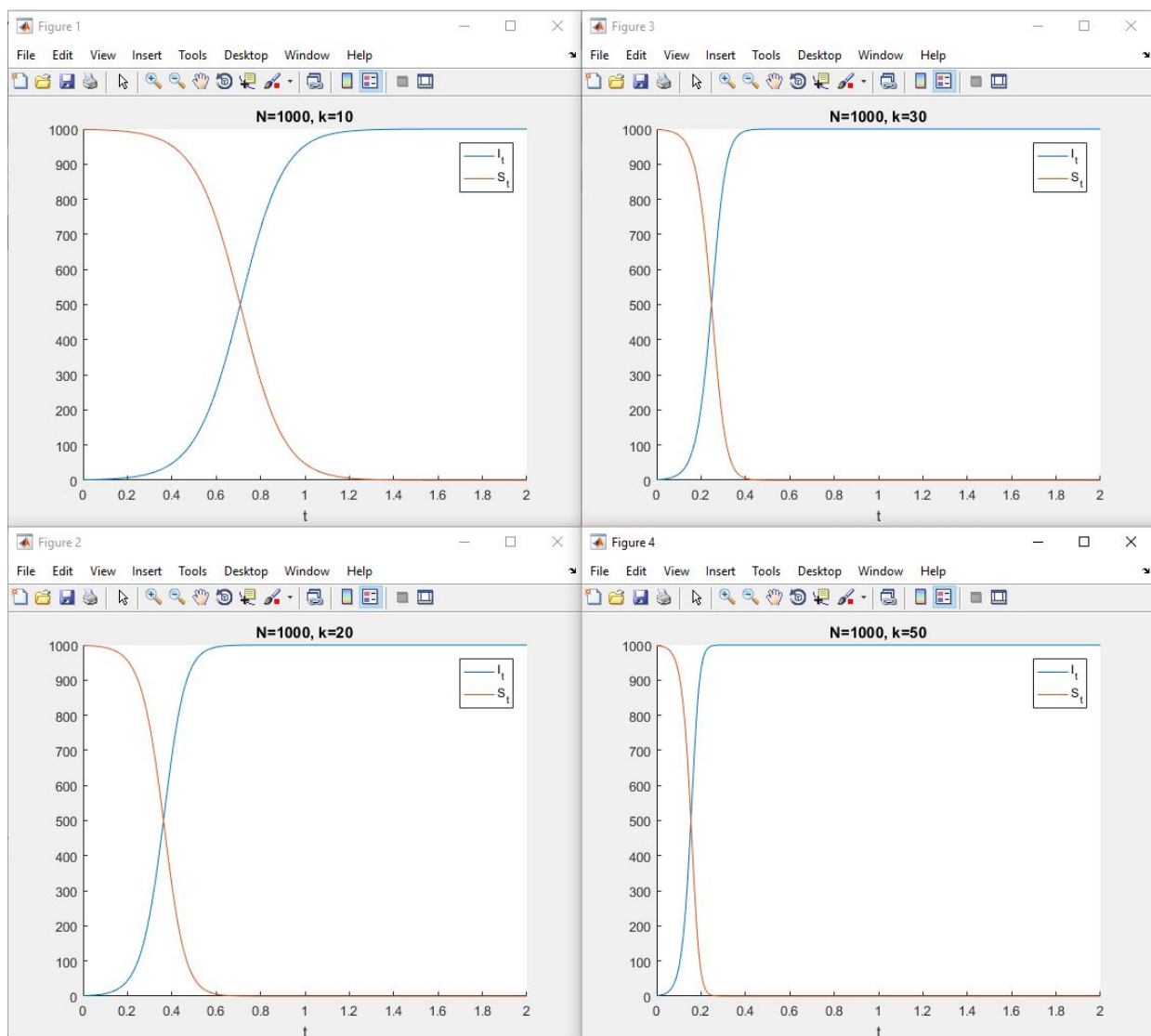


Abbildung 1: Verlauf der Epidemie mit einem logistischen Modell

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.

Übung 1

2 Epidemien - SIR-Modell

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

```

1 % simulation arguments
2 N = 10000;
3 RO = 7.5; % Ro = alpha/beta
4 beta = 1/10; % one healing per 10-days
5 mu = 0.0003; % bearth rate
6 alpha_a = RO * beta; % alpha for a)
7 alpha_b = RO * (beta + mu); % alpha for b)
8 iStart = 1; % start of infected
9 p_crit = 1 - (1/RO); % p_crit
10
11 % Logisitic Simulation parameters 'epidemiologySIR'
12 tStart = 0;
13 tStep = 0.001;
14 tMax = 100;
15 tMaxMuP = 10000;
16
17 % with no mu and p_crit
18 resA = epidemiologySIR(tStart, tStep, tMaxMuP, alpha_a, beta, iStart, N);
19 % with mu and no p_crit
20 resB = epidemiologySIR(tStart, tStep, tMaxMuP, alpha_b, beta, iStart, N, mu);
21 % with mu and p_crit
22 resC = epidemiologySIR(tStart, tStep, tMaxMuP, alpha_b, beta, iStart, N, mu, p_crit);
23
24 % with no mu and p_crit
25 plotSir(resA, 'a', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_a), ...
26 num2str(beta), 'undefined', 'undefined');
27 % with mu and no p_crit
28 plotSir(resB, 'b.1', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_b), ...
29 num2str(beta), num2str(mu), 'undefined');
30 % with mu and p_crit
31 plotSir(resB, 'b.2', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_b), ...
32 num2str(beta), num2str(mu), num2str(p_crit));
33 % with mu and p_crit (zoom for equilibrium)
34 plotSir(resB, 'b.2.1', (tStart:tStep:tMaxMuP), num2str(N), num2str(alpha_b), ...
35 num2str(beta), num2str(mu), num2str(p_crit));

```

Listing 4: Funktion zur Berechnung des SIR-Modell

```

1 function result = epidemiologySIR(tStart, tStep, tMax, alpha, beta, iStart, N, mu, p)
2 % Does not work properly, don't know why yet !!!!
3
4 % simulation arguments
5 i = iStart;
6 r = 0;
7 s = N - i - r;
8
9 sProgress = zeros(tMax/tStep+1,1);
10 iProgress = zeros(tMax/tStep+1,1);
11 rProgress = zeros(tMax/tStep+1,1);
12 j = 1;
13
14 for t=tStart:tStep:tMax
15 % birth rate
16 s_ = -alpha * s * (i / N);
17 i_ = alpha * s * (i / N) - (beta * i);
18 r_ = (beta * i);
19

```

Übung 1

```

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

```

Übung 1

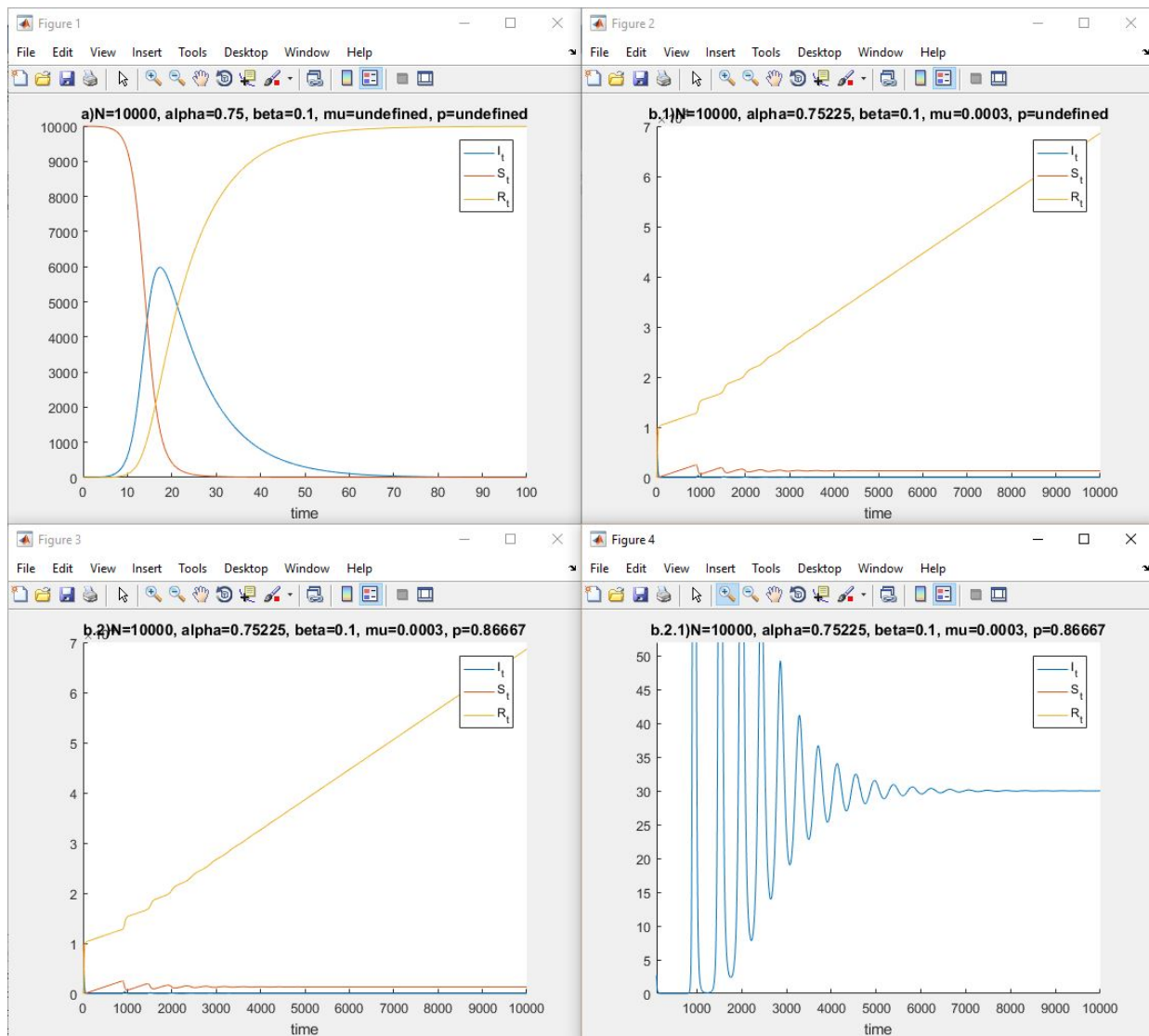


Abbildung 2: Verlauf der Epidemie mit SIR-Modell

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.

```

1 % simulation parameter
2 config = [];
3 config.tStart = 0;
4 config.tStep = 0.01;
5 config.tMax = 365; % in days
6
7 % simulation arguments (given parameters were normalized to days)
8 groups(1) = struct('N', 1000, ...
9                  'k', (2/7), ...
10                 'iStart', 0, ...
11                 'alpha', 0.15, ...
12                 'beta', (1/14), ...
13                 'm', [0.9 0.05 0.05]);
14
15 groups(2) = struct('N', 100, ...
16                 'k', (10/7), ...
17                 'iStart', 0, ...
18                 'alpha', 0.15, ...
19                 'beta', (1/14), ...
20                 'm', [0.1 0.7 0.2]);
21
22 groups(3) = struct('N', 500, ...
23                 'k', (4/7), ...
24                 'iStart', 5, ...
25                 'alpha', 0.15, ...
26                 'beta', (1/(7 + (7/2))), ...
27                 'm', [0.3 0.3 0.4]);
28
29 result = epidemiologySIRGroups(config, groups);
30
31 for i=1:1:3
32     curGroup = groups(i);
33     plotSir(result(i), ...
34            strcat('Group-', strcat(num2str(i), ',')), ...
35            (config.tStart:config.tStep:config.tMax), ...
36            num2str(curGroup.N), ...
37            num2str(curGroup.alpha), ...
38            num2str(curGroup.beta), ...
39            'unused', ...
40            'unused');
41 end

```

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

```

1 function result = epidemiologySIRGroups(config, groups)
2 % calculates the epidemiology for n given groups via an SIR model
3
4 % simulation arguments
5 groupSize = size(groups);
6 dataCount = groupSize(1,2);
7 idx = 1;
8 runCont(1:dataCount) = struct('i', 0, 'r', 0, 's', 0);
9
10 % initialize run container with start values
11 for j=1:1:dataCount
12     curData = groups(j);
13     runCont(j) = struct('s', (curData.N - curData.iStart), ...

```

Übung 1

```

14         'i', curData.iStart, ...
15         'r', 0);
16     end
17
18     % initialize result container
19     result(1:dataCount) = struct('sProg', zeros(config.tMax/config.tStep+1,1), ...
20                                 'iProg', zeros(config.tMax/config.tStep+1,1), ...
21                                 'rProg', zeros(config.tMax/config.tStep+1,1));
22
23     % main loop for time
24     for t=config.tStart:config.tStep:config.tMax
25
26         % loop for given data groups
27         for j=1:1:dataCount
28             curRun = runCont(j); % holds last current results or start values at begin
29             curData = groups(j); % the current group to calculate
30
31             % old current values
32             s = curRun.s;
33             i = curRun.i;
34             r = curRun.r;
35             sum = 0;
36             alpha = curData.alpha;
37             beta = curData.beta;
38
39             % calculate group-wide infections for current group
40             for k=1:1:dataCount
41                 sData = groups(k);
42                 sum = sum + ((sData.k * sData.m(j)) * (runCont(k).i/sData.N));
43             end
44
45             % calculate deviation
46             s_ = (-alpha * sum * s);
47             i_ = (alpha * sum * s) - (beta * i);
48             r_ = (beta * i);
49
50             % set new current values for next iteration
51             runCont(j) = struct('s', (s + (s_ * config.tStep)), ...
52                                 'i', (i + (i_ * config.tStep)), ...
53                                 'r', (r + (r_ * config.tStep)));
54
55             % remember results
56             result(j).sProg(idx) = runCont(j).s;
57             result(j).iProg(idx) = runCont(j).i;
58             result(j).rProg(idx) = runCont(j).r;
59
60         end
61
62         % increase group prog index
63         idx = idx + 1;
64     end
65 end

```


Übung 1

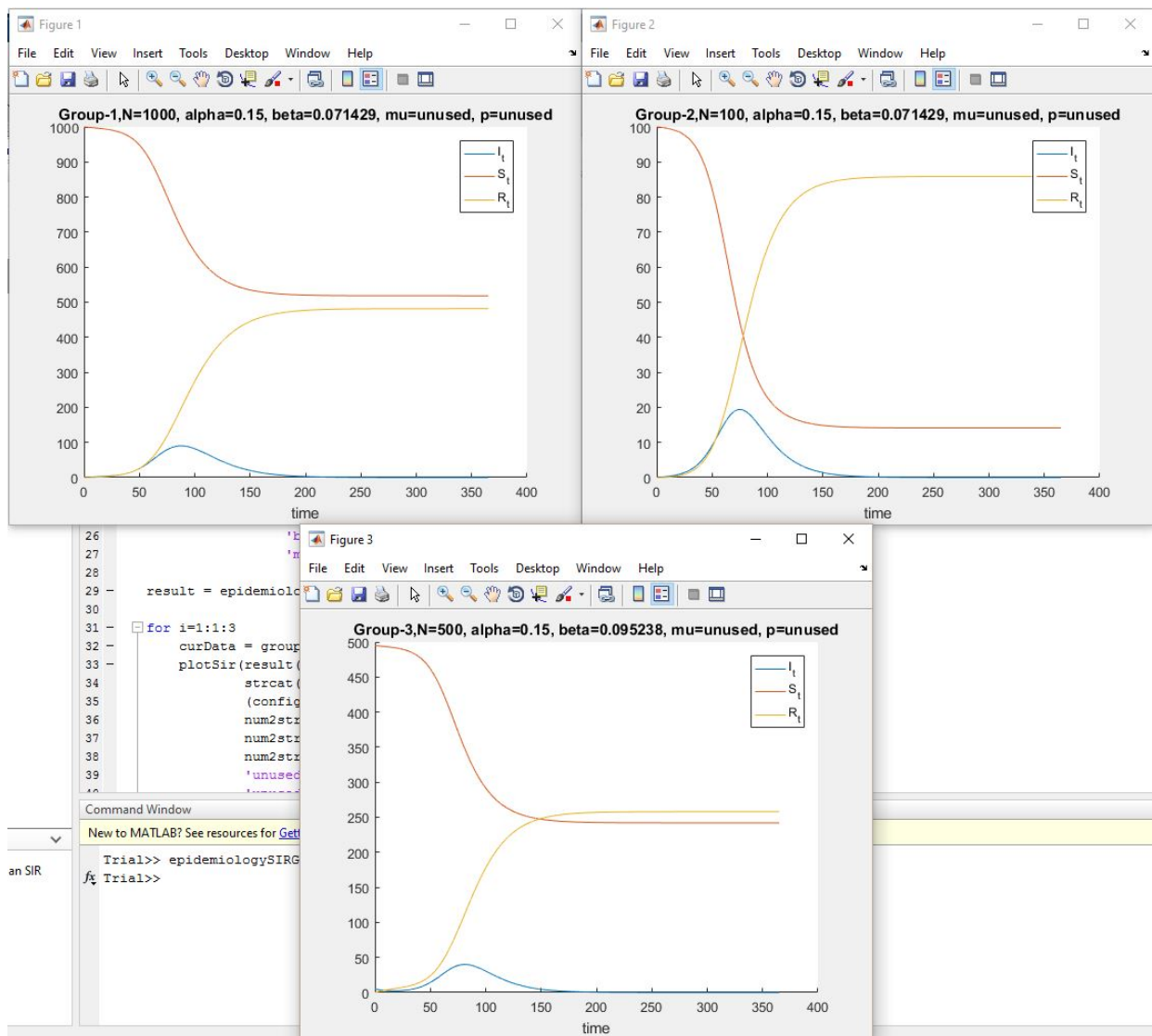


Abbildung 3: Verlauf der Epidemie in den verschiedenen Gruppen mit einem SIR-Modell