# Pandemieausbreitung

Wayne Ströbel, Silas Müller

5/30/2021

## Pandemieausbreitung

Wir betrachten eine leicht generalisierte Form des sogenannten SIR Modells. Das SIR Modell ist ein einfaches Modell um Pandemien zu beschreiben. S steht für susceptible, I für infected und R für recovered. S, I und R bezeichnen also jeweils die Anzahl an Personen. Außerdem beschreibt V die Anzahl an geimpften Personen. Die Zeitentwicklung dieser Größen wird durch folgende gekoppelte, gewöhnliche Differentialgleichungen beschrieben, die im folgenden unter Gesichtspunkten gelöst werden.

## Differentialgleichungen

$$S'(t) = -\frac{\beta}{N} S(t) I(t) - \Gamma(t) + \delta V(t) \tag{1}$$

$$I'(t) = \frac{\beta}{N} S(t) I(t) - \alpha I(t) \tag{2}$$

$$R'(t) = \alpha I(t) \tag{3}$$

$$V'(t) = \Gamma(t) - \delta V(t) \tag{4}$$

N ist die Größe der Population und die Zeit wird in Tagen gerechnet. Das Verhältnis $R_0 = \frac{\beta}{\alpha}$ ist die sogenannte Basisreproduktionszahl. Fur SARS-CoV-2 ist $R_0 \approx 2.9$ (0hne Maßnahmen), $\alpha$ ist ungefähr die inverse Zeit, die Infizierte ansteckend sind, also $\frac{1}{\alpha} = 7$ Tage für SARS-CoV-2.

## Lösungsverfahren: Runge-Kutta 2

Das Lösen von Differentialgleichungen kann letzlich auf ein Integrationsproblem zurückgeführt werden, wenn die entsprechenden Anfangswerte bekannt sind. Dabei gilt: $x_i = x(t_i)$ Zunächst betrachte man nur:

$$\dot{x}(t) = f(t, x(t))$$

Nun werden die Zeitschritte $t$ diskretisiert:

$$t_i = a + ih \;\; i = 0, 1, 2, ..., n \;\; und \;\; h = \frac{b-a}{n} \tag{5}$$

Nun hat muss man das Integrationsproblem lösen:

$$x_{i+1} = x_i + \int_{t_i}^{t_{i+1}} f(t', x(t'))\, dt' \tag{6}$$

Die einfachste Möglichkeit ist das Euler-Verfahren, wir nutzen hier jedoch das RK-2-Verfahren, was in der Vorlesung genauer vorgestellt wurde:

$$xi + 1 = x_i + \frac{h}{2}(f(t_i, x_i) + f(t_i + h, x_i + hf(t_i, x_i))) \tag{7}$$

Dieses Verfahren leifert einen Fehler der Ordnung $O(h^3)$ und wird im folgenden genutzt.

## Code

```
#include <Rcpp.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <algorithm>



//Definitionen für benutzte Größen

typedef struct{

    double alpha = 0, beta = 0, delta = 0, N = 0, gamma = 0;

} Parameter;

typedef struct{

    double S = 0, I = 0, R = 0, V = 0;

} Status;

//Definition der Funktion f für die DGLs

void f(Status oldStatus, Parameter parameter, Status& newStatus){

    double Gamma = std::min(parameter.gamma,oldStatus.S);
    newStatus.S = - ((parameter.beta/parameter.N) * oldStatus.S * oldStatus.I)
    - (Gamma) + (parameter.delta * oldStatus.V);
    newStatus.I = ((parameter.beta/parameter.N) * oldStatus.S * oldStatus.I)
    - (parameter.alpha * oldStatus.I);
    newStatus.R = (parameter.alpha * oldStatus.I);
    newStatus.V = (Gamma) - (parameter.delta * oldStatus.V);

}

//Einzelner Schritt des Runge Kutta Verfahrens

void rungeKuttaStep(Status oldStatus, Parameter parameter, Status& newStatus, double h){

    //Definition für f(x_i)
    Status f_i;
    f(oldStatus, parameter, f_i);
```

```
    //Definition für f(x_(i+1))
    Status f_i1;

    Status rechnung;

    rechnung.S = oldStatus.S + h*f_i.S;
    rechnung.I = oldStatus.I + h*f_i.I;
    rechnung.R = oldStatus.R + h*f_i.R;
    rechnung.V = oldStatus.V + h*f_i.V;

    f(rechnung, parameter, f_i1);

    //Rechenschritt
    newStatus.S = oldStatus.S + h/2 * (f_i.S + f_i1.S);
    newStatus.I = oldStatus.I + h/2 * (f_i.I + f_i1.I);
    newStatus.R = oldStatus.R + h/2 * (f_i.R + f_i1.R);
    newStatus.V = oldStatus.V + h/2 * (f_i.V + f_i1.V);
}


//Anwendung des Runge Kutta Verfahrens

//[[Rcpp::export]]
Rcpp::List Simulation(double alpha, double beta, double delta,
    double gamma, double N, double I0, double R0, double V0, double h, double max)
{

    Rcpp::NumericVector t(1);
    Rcpp::NumericVector Su(1);
    Rcpp::NumericVector In(1);
    Rcpp::NumericVector Re(1);
    Rcpp::NumericVector Va(1);


    //Definition der Parameter
    Parameter SIR_P;
    SIR_P.alpha = alpha;
    SIR_P.beta = beta;
    SIR_P.delta = delta;
    SIR_P.gamma = gamma;
    SIR_P.N = N;

    //Anfangswerte
    Status old;
    old.S = SIR_P.N - I0 - R0 - V0;
    old.I = I0;
    old.R = R0;
    old.V = V0;

    //Schreibt Anfangswerte in Ausgabeliste
    t[0] = 0;
    Su[0] = old.S;
    In[0] = old.I;
```

```
Re[0] = old.R;
Va[0] = old.V;

//Laufvariable, wichtig zum Interpretieren der Zeit
double i = 0;


if (max == 0) {
// max = 0 entspricht keiner Obergrenze
// Laufende Schleife, bricht ab sobald infizierte unter 1 läuft
// h beschreibt die Schritteweite in Tagen.
//(z.B. h=0.5 entspricht einer Schrittweite von 12 Stunden)
while (old.I > 1)
  {
      rungeKuttaStep(old, SIR_P, old, h);
      i += h;
      t.push_back(i);
      Su.push_back(old.S);
      In.push_back(old.I);
      Re.push_back(old.R);
      Va.push_back(old.V);
  }
}

else if (max == -1){
//Sonderfall für letzte Aufgabe
while (old.I > 10000)
  {
      rungeKuttaStep(old, SIR_P, old, h);
      i += h;
      t.push_back(i);
      Su.push_back(old.S);
      In.push_back(old.I);
      Re.push_back(old.R);
      Va.push_back(old.V);
  }


}

else {
// falls ein höchster Tag angegeben wurde, bricht der Code
// nach dem max-ten Tag ab
  while (old.I > 1 && i <= max){

    rungeKuttaStep(old, SIR_P, old, h);
    i += h;
    t.push_back(i);
    Su.push_back(old.S);
    In.push_back(old.I);
    Re.push_back(old.R);
    Va.push_back(old.V);
```

```
      }
    }
    //Schreibt Vectoren der Werte in Liste, welche an R weitergegeben wird
    Rcpp::List L = Rcpp::List::create(t, Su, In, Re, Va);

    return L;
}
```

# Neue Krankheit: DGL ohne Gegenmaßnahmen

Zunächst wird für eine neue Krankheit (also $R(0) = 0$) mit $\frac{1}{\alpha} = 7$ und $\Gamma = \gamma = 0$, sowie der Gesamtbevölkerung Deutschlands $N = 8.3 \cdot 10^7$, den Anteil an der Gesamtbevölkerung bestimmt, der infiziert wurde, bis die Krankheit abklingt ($I(t) < 1$), als Funktion von $R_0 \in [1, 20]$. Dabei sollen folgende Anfangsbedingungen gelten:

$$S(0) = N - 5000, \;\; I(0) = 5000, \;\; R(0) = V(0) = 0$$

Das Ergebnis soll mit der üblicherweise angegebenen Herdenimmunität

$$\frac{S}{N} = \frac{1}{R_0}$$

verglichen und die Diskrepanz erklärt werden.

## Anwendung von RK 2

Im folgenden Code wird das Vefahren angewendet. Dabei ist zu beachten, dass die jeweiligen Funktionen $f(t, x(t))$ nicht explizit von der Zeit abhängen, sodass sich jweils Rekursionen der Art $x_{i+1} = F(x_i, y_i)$ ergeben. Also explizit:

$$f(t_i, s_i) = -\alpha \frac{R_0}{N} I(t_i) s_i \tag{8}$$

$$f(t_i, I_i) = \alpha \frac{R_0}{N} S(t_i) I_i - \alpha I_i \tag{9}$$

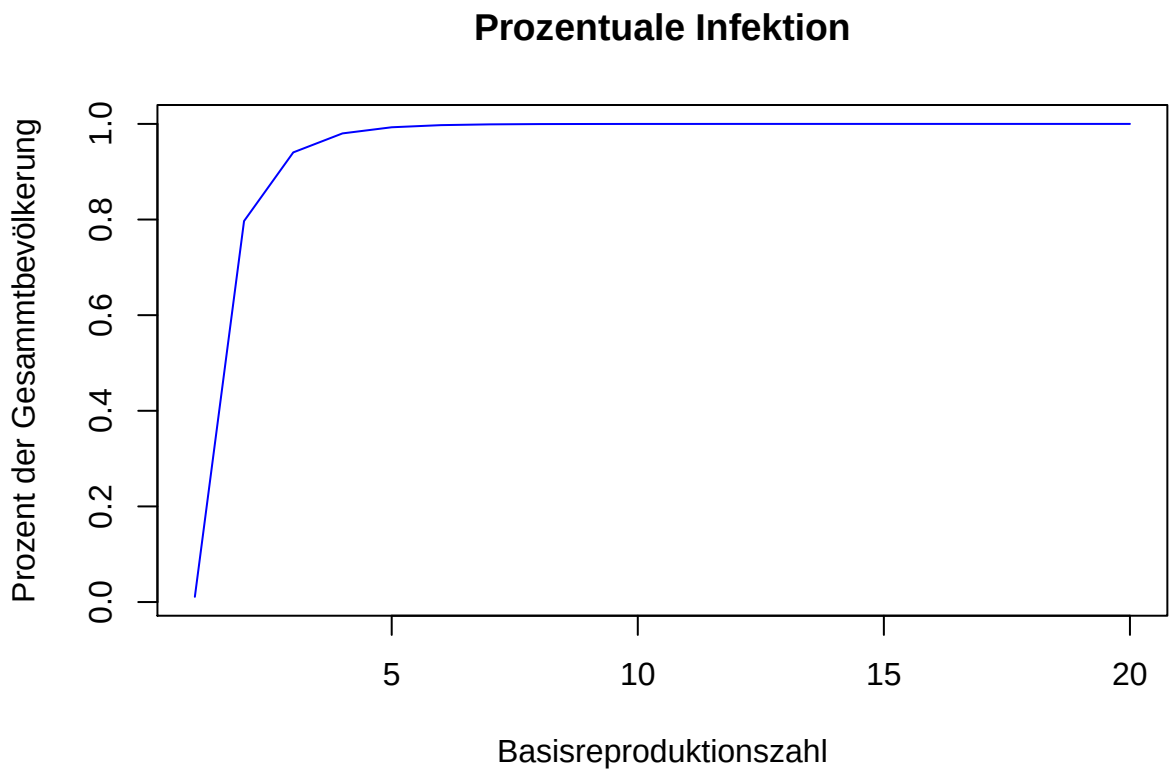
## Code

```
#Basisreproduktionszahl
R_0 <- 1:20

#Anzahl an Leuten N
N <- 8.3e7

#Initialisierung für %
v <- rep(0, 20)

#Ausführung des Verfahrens für verscheidene R_0
for (i in R_0) {
List <- Simulation(1./7.,i/7.,0,0,N,5000, 0, 0, 0.25, 0)
R <- unlist(List[4])
v[i] <- tail(R, n=1) / N
}
```

### Ergebnis

Wir plotten den relativen Anteil der Bevölkerung, die das Virus besiegt haben muss, bis die Krankheit abgklingt, gegen $R_0$. Für unsere Krankheit mit $R_0 \approx 2.9$ liefert die Simulation einen Anteil von $\approx 0.9404657$ Vergleicht man dies mit dem erwarteten Wert von $1-\frac{1}{R_0} \approx 0.655$ So stellt sich doch eine deutliche Diskrepanz dar. Dabei ist zu beachten, dass diese Simulation keine Tode berücksichtigt. Außerdem ist der R-Wert nur grob angegeben und eine Veränderung dessen während der Pandemie wird ignoriert. Die Fehler des Verfahrens sollten, wie bereits erwähnt, nicht annähernd in dieser Größenordnung liegen.

```
#Plotten von R_0 gegen %
plot(R_0, v,
     main="Prozentuale Infektion",
     ylab="Prozent der Gesammtbevölkerung",
     xlab="Basisreproduktionszahl",
     col="blue",
     type="l")
```




## Simulation des Pandemieverlaufs: DGL mit Gegenmaßnahmen

Nun erlauben wir nicht-pharmazeutische Maßnahmen, die $R_0$ reduzieren, und nehmen an, dass Impfungen vorgenommen werden. Naturlich ist $\Gamma(t)$ nach oben beschränkt durch $S(t)$.Wir nehmen an, dass $\gamma(t)$ Personen pro Zeit geimpft werden können. Dementsprechend gilt:

$$\Gamma(t) = min[\gamma(t), S(t)]$$

Geimpfte können mit Rate $\delta$ wieder infiziert werden. $\delta$ ist ungefähr von der Größenordnung von einigen inversen Monaten. Wir nehmen hier an:

$$\frac{1}{\delta} = 150 \ Tage$$

Nun wird mit den realen Zahlen $S, I, R, V, \gamma(t)$ vom 1. Januar bis erstem Juni der restliche Pandemieverlauf für 2021 simuliert.Für die Zukunft wird $\gamma(t)$ konstant auf den letzten bekannten Wert gesetzt.$R_0$ wird so angepasst, dass die Simulation möglichst realistisch den bisherigen tatsächlichen Verlauf wiedergibt.

## Daten

Zum Plotten benutzen wir die folgenden Daten:

| Datum | Infizierte | Geimpft | Geimpfte pro Zeit | $R_0$ |
|---|---|---|---|---|
| 4.1 | 339700 | 0 | 0 | 0.98 |
| 29.1 | 238200 | 416646 | 43878 | 0.89 |
| 26.2 | 119400 | 1971043 | 42536 | 1 |
| 31.3 | 210700 | 4059489 | 90070 | 0.96 |
| 30.4 | 303600 | 6381397 | 126674 | 0.88 |
| 31.5 | 106000 | 14615052 | 172147 | 0.77 |

Table 1: Daten des RKI für Covid Verlauf von 2021

Mit diesen Daten können wir nun den Verlauf Simulieren und Plotten.

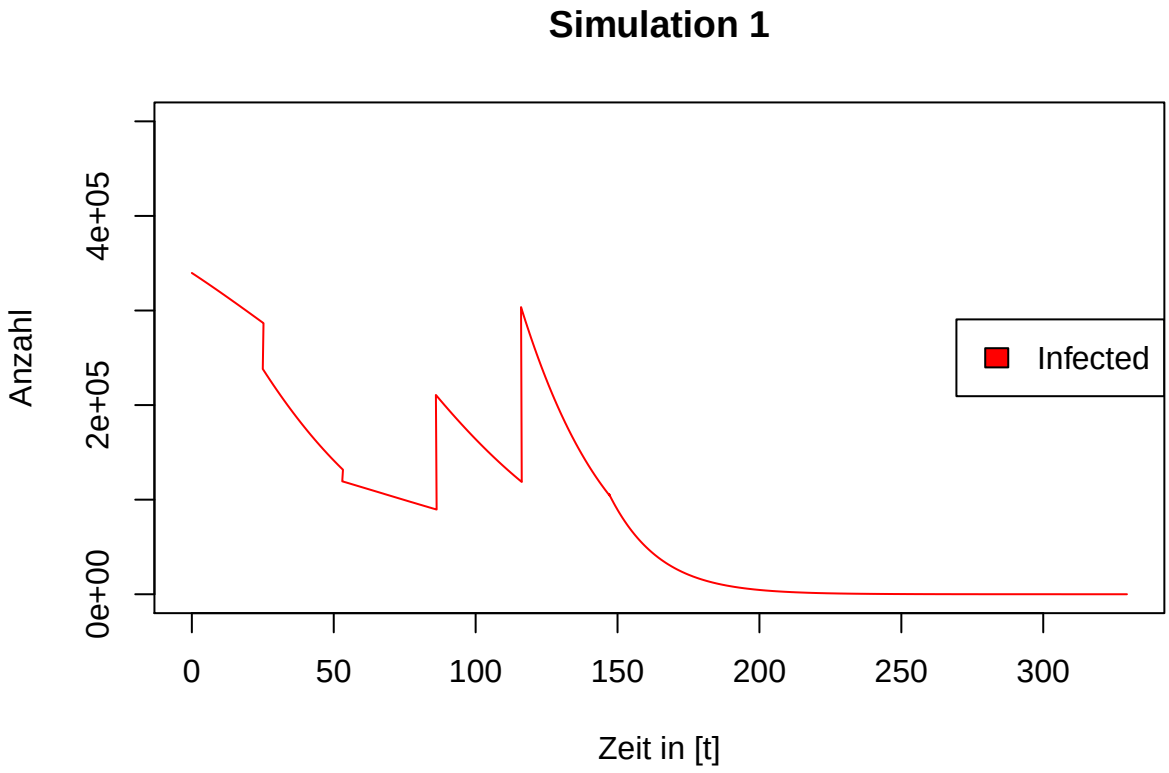
## Code

```
#Schrittweite
h <- 0.25

#Simulationen für bestimmte Zeiträume


#4.1 bis 29.1
hStep1 <- Simulation(1./7.,0.98/7.,1./150.,0,8.3e7,339700, 1401200, 0, h, 25)

I <- unlist(hStep1[3])
t <- unlist(hStep1[1])

#29.1 bis 26.2
hStep2 <- Simulation(1./7.,0.89/7.,1./150.,43878,8.3e7,238200, 1898900, 416646, h, 28)

I <- append(I ,unlist(hStep2[3]))
t <- append(t ,unlist(hStep2[1]) + 25)

#26.2 bis 31.3   MIT ANPASSUNG VON R_0
hStep3 <- Simulation(1./7.,1/7.,1./150.,42536,8.3e7,119400, 2235700, 1971043, h, 33)

I <- append(I ,unlist(hStep3[3]))
t <- append(t ,unlist(hStep3[1]) + 25 + 28)

#31.3 bis 30.4  MIT ANPASSUNG VON R_0
hStep4 <- Simulation(1./7.,0.96/7.,1./150.,90070,8.3e7,210700, 2521800, 4059489, h, 30)
```

```
I <- append(I ,unlist(hStep4[3]))
t <- append(t ,unlist(hStep4[1]) + 25 + 28 + 33)

#30.4 bis 31.5
hStep5 <- Simulation(1./7.,0.88/7.,1./150.,126674,8.3e7,303600, 2995200, 6381397, h, 31)

I <- append(I ,unlist(hStep5[3]))
t <- append(t ,unlist(hStep5[1]) + 25 + 28 + 33 + 30)

#31.5 bis zur Abklingung der Krankheit
hStep6 <- Simulation(1./7.,0.77/7.,1./150.,172147,8.3e7,106000, 3485700, 14615052, h, 200)

I <- append(I ,unlist(hStep6[3]))
t <- append(t ,unlist(hStep6[1]) + 25 + 28 + 33 + 30 + 31)
```

### Plots und Erläuterung

```
plot(t, I,
     main="Simulation 1",
     ylab="Anzahl",
     xlab="Zeit in [t]",
     ylim=c(0,5e5),
     type="l",
     col="red")
legend("right", c("Infected"), fill=c("red"))
```

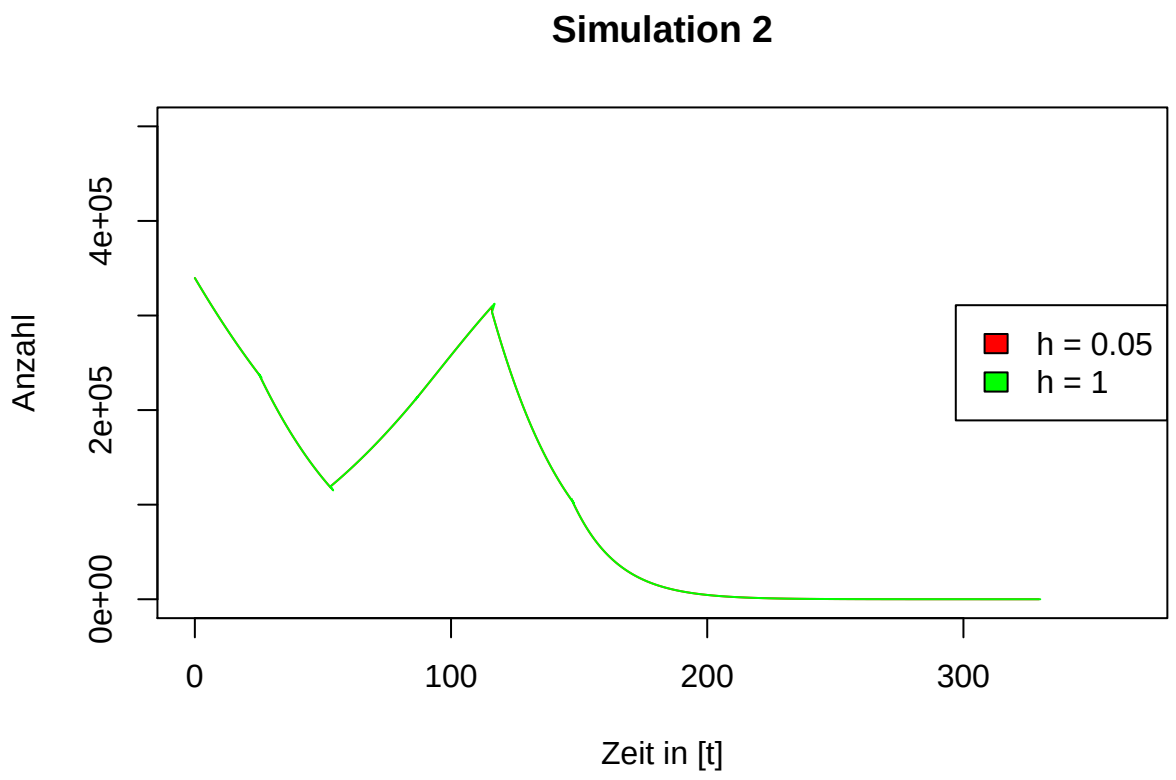



Nun können wir die $R_0$ Werte der Zeiträume so anpassen, dass wir einen Verlauf erhalten, der besser dem Realen Verlauf beschreibt:
Des Weiteren führen wir den Code mit verschiedenen Größen h durch, um die Unterschiede zu diskutieren.

```
#Dieser Code ist sehr unschön, leider haben wir keinen anderen Weg gefunden, das ganze
#besser aufzuschreiben, ohne einen großteil der vorherigen Funktionen umzuschreiben
# Sorry :C

#Schrittweite
h1 <- 0.05


#Simulationen für bestimmte Zeiträume


#4.1 bis 29.1
List <- Simulation(1./7.,0.925/7.,1./150.,0,8.3e7,339700, 1401200, 0, h1, 25)

I1 <- unlist(List[3])
t1 <- unlist(List[1])

#29.1 bis 26.2
List <- Simulation(1./7.,0.86/7.,1./150.,43878,8.3e7,238200, 1898900, 416646, h1, 28)

I1 <- append(I1 ,unlist(List[3]))
t1 <- append(t1 ,unlist(List[1]) + 25)
```

```
#26.2 bis 31.3   MIT ANPASSUNG VON R_0
List <- Simulation(1./7.,1.195/7.,1./150.,42536,8.3e7,119400, 2235700, 1971043, h1, 33)

I1 <- append(I1 ,unlist(List[3]))
t1 <- append(t1 ,unlist(List[1]) + 25 + 28)

#31.3 bis 30.4  MIT ANPASSUNG VON R_0
List <- Simulation(1./7.,1.21/7.,1./150.,90070,8.3e7,210700, 2521800, 4059489, h1, 30)

I1 <- append(I1 ,unlist(List[3]))
t1 <- append(t1 ,unlist(List[1]) + 25 + 28 + 33)

#30.4 bis 31.5
List <- Simulation(1./7.,0.88/7.,1./150.,126674,8.3e7,303600, 2995200, 6381397, h1, 31)

I1 <- append(I1 ,unlist(List[3]))
t1 <- append(t1 ,unlist(List[1]) + 25 + 28 + 33 + 30)

#31.5 bis zur Abklingung der Krankheit
List <- Simulation(1./7.,0.77/7.,1./150.,172147,8.3e7,106000, 3485700, 14615052, h1, 200)

I1 <- append(I1 ,unlist(List[3]))
t1 <- append(t1 ,unlist(List[1]) + 25 + 28 + 33 + 30 + 31)



#Schrittweite
h2 <- 1

#Simulationen für bestimmte Zeiträume


#4.1 bis 29.1
List <- Simulation(1./7.,0.925/7.,1./150.,0,8.3e7,339700, 1401200, 0, h2, 25)

I2 <- unlist(List[3])
t2 <- unlist(List[1])

#29.1 bis 26.2
List <- Simulation(1./7.,0.86/7.,1./150.,43878,8.3e7,238200, 1898900, 416646, h2, 28)

I2 <- append(I2 ,unlist(List[3]))
t2 <- append(t2 ,unlist(List[1]) + 25)

#26.2 bis 31.3   MIT ANPASSUNG VON R_0
List <- Simulation(1./7.,1.195/7.,1./150.,42536,8.3e7,119400, 2235700, 1971043, h2, 33)

I2 <- append(I2 ,unlist(List[3]))
t2 <- append(t2 ,unlist(List[1]) + 25 + 28)

#31.3 bis 30.4  MIT ANPASSUNG VON R_0
List <- Simulation(1./7.,1.21/7.,1./150.,90070,8.3e7,210700, 2521800, 4059489, h2, 30)
```

```
I2 <- append(I2 ,unlist(List[3]))
t2 <- append(t2 ,unlist(List[1]) + 25 + 28 + 33)

#30.4 bis 31.5
List <- Simulation(1./7.,0.88/7.,1./150.,126674,8.3e7,303600, 2995200, 6381397, h2, 31)

I2 <- append(I2 ,unlist(List[3]))
t2 <- append(t2 ,unlist(List[1]) + 25 + 28 + 33 + 30)

#31.5 bis zur Abklingung der Krankheit
List <- Simulation(1./7.,0.77/7.,1./150.,172147,8.3e7,106000, 3485700, 14615052, h2, 200)

I2 <- append(I2 ,unlist(List[3]))
t2 <- append(t2 ,unlist(List[1]) + 25 + 28 + 33 + 30 + 31)



plot(t1, I1,
     main="Simulation 2",
     ylab="Anzahl",
     xlab="Zeit in [t]",
     ylim=c(0,5e5),
     xlim=c(0,365),
     type="l",
     col="red")
lines(t2, I2, col="green")
legend("right", c("h = 0.05","h = 1"),
       fill=c("red","green"))
```




Wir erkennen allgemein nur sehr kleine Abweichungen für verschiedene h. Es ist jedoch gut zu erkennen (gerade am Ende des jeweiligen Simulationsabschnitts), dass ein größeres h zu einer größeren Schrittweite und so zu einer ungenaueren Prognose führt.

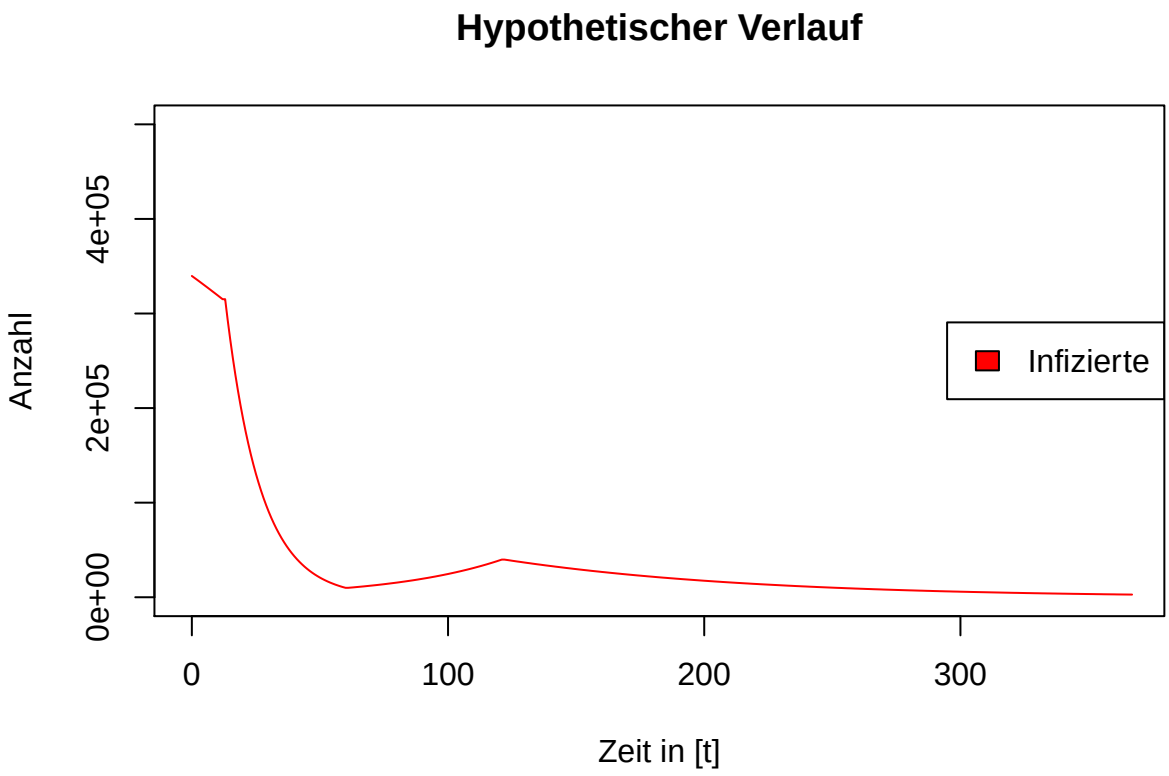
## Vergleich mit hypothetischem Verlauf

Die obige Simulation wird mit folgendem hypothetischem Verlauf verglichen:
Ab Mitte Januar wird der $R_0$-Wert so lange auf 0.5 gesenkt, bis I $<$ 10000 gilt und anschließend wird $R_0$=1.2 gehalten. Ab Anfang Mai sinkt die Reproduktionszahl wegen saisonaler Effekte auf 80% dieses Wertes.

### Code

```
#Schrittweite
h <- 1

#Simulationen für bestimmte Zeiträume


#4.1 bis 15.1
hStep1 <- Simulation(1./7.,0.98/7.,1./150.,0,8.3e7,339700, 1401200, 0, h, 11)

I <- unlist(hStep1[3])
t <- unlist(hStep1[1])
```

```
#15.1 bis I<10000
hStep2 <- Simulation(1./7.,0.5/7.,1./150.,0,8.3e7,tail(I, n=1), tail(unlist(hStep1[4]), n=1), 0, h, -1)

I <- append(I, unlist(hStep2[3]))
t <- append(t, unlist(hStep2[1]) + length(t)*h)

#jetzt bis 1.5
hStep3 <- Simulation(1./7.,1.2/7.,1./150.,0,8.3e7,tail(I, n=1), tail(unlist(hStep2[4]), n=1), 0, h, 119

I <- append(I, unlist(hStep3[3]))
t <- append(t, unlist(hStep3[1]) + length(t)*h)

#1.5 bis Ende
hStep4 <- Simulation(1./7.,0.8*1.2/7.,1./150.,0,8.3e7,tail(I, n=1), tail(unlist(hStep3[4]), n=1), 0, h,

I <- append(I, unlist(hStep4[3]))
t <- append(t, unlist(hStep4[1]) + length(t)*h)


plot(t, I,
     main="Hypothetischer Verlauf",
     ylab="Anzahl",
     xlab="Zeit in [t]",
     ylim=c(0,5e5),
     xlim=c(0,365),
     type="l",
     col="red")
legend("right", c("Infizierte"),
       fill=c("red"))
```




### Erläuterungen und Vergleich

Betrachtet man den hypothetischen Verlauf, so lässt sich erkennen, dass durch die starke Senkung des R-Wertes am Anfang (bis I<10000), die Pandemie gebrochen scheint. So kann selbst ein darauffolgender R-Wert von 1.2 nicht zu einer erneuten Welle führen.

## Fazit

Durch das Runge-Kutta-Verfahren konnte das SIR-Modell gut modelliert und ein Pandemieverlauf simuliert werden. Des Weiteren konnten die DGLs mit Gegenmaßnahmen ebenfalls numerisch genähert und dargestellt werden.
Hierbei ist zu erwähnen, dass nur fünf Phasen mit reellen Eckdaten simuliert wurden. Insbesondere eine tägliche Änderung des R-Wertes (z.B. durch Masken und Impfungen) konnte nicht gewährleist werden, was nur zu einem groben Simulation des reellen Verlaufs geführt hat. Kleine Schrittweiten stellten sich als insgesamt besser dar, da der Verlauf so besser beschrieben werden konnte (gerade wenn mehr reale Eckdaten gegeben sind). Natürlich verringert sich auch der Verfahrens-Fehler des Runge-Kutta-Verfahrens.
Ein hypothetischer Pandemieverlauf konnte entsprechend simuliert und mit dem realen Verlauf verglichen werden.

# Literatur

*1.Täglicher Lagebericht des RKI zur Coronavirus-Krankheit-2019 (COVID-19) https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/Situationsberichte/Jan_2021/2021-01-29-de.pdf?___blob=publicationFile ,Stand 06.06.2021*

*2.Täglicher Lagebericht des RKI zur Coronavirus-Krankheit-2019 (COVID-19) https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/Situationsberichte/Feb_2021/2021-02-26-de.pdf?___blob=publicationFile ,Stand 06.06.2021*

*3.Täglicher Lagebericht des RKI zur Coronavirus-Krankheit-2019 (COVID-19) https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/Situationsberichte/Maerz_2021/2021-03-31-de.pdf?___blob=publicationFile ,Stand 06.06.2021*

*4.Täglicher Lagebericht des RKI zur Coronavirus-Krankheit-2019 (COVID-19) https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/Situationsberichte/Apr_2021/2021-04-30-de.pdf?___blob=publicationFile ,Stand 06.06.2021*

*5.Täglicher Lagebericht des RKI zur Coronavirus-Krankheit-2019 (COVID-19) https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/Situationsberichte/Mai_2021/2021-05-31-de.pdf?___blob=publicationFile ,Stand 06.06.2021*

*6.Täglicher Lagebericht des RKI zur Coronavirus-Krankheit-2019 (COVID-19) https://www.rki.de/DE/Content/InfAZ/N/Neuartiges_Coronavirus/Situationsberichte/Jan_2021/2021-01-04-de.pdf?___blob=publicationFile, Stand 07.06.2021*

*7.Die wichtigsten Zahlen für Deutschland https://www.spiegel.de/wissenschaft/medizin/coronavirus-infizierte-genesene-tote-alle*

*8.Vorlesung zum Runge-Kutta Verfahren*