Chapter 2
SIR

2.1 The SIR Model

In 1927, Kermack and McKendrick (1927) published a set of general equations (Breda et al. 2012) to better understand the dynamics of an infectious disease spreading through a susceptible population. Their motivation was

> "One of the most striking features in the study of epidemics is the difficulty of finding a causal factor which appears to be adequate to account for the magnitude of the frequent epidemics of disease which visit almost every population [...] The problem may be summarized as follows: One (or more) infected person is introduced into a community of individuals, more or less susceptible to the disease in question. The disease spreads from the affected to the unaffected by contact infection. Each infected person runs through the course of his sickness, and finally is removed from the number of those who are sick, by recovery or by death. The chances of recovery or death vary from day to day during the course of his illness. The chances that the affected may convey infection to the unaffected are likewise dependent upon the stage of the sickness. As the epidemic spreads, the number of unaffected members of the community becomes reduced [...] In the course of time the epidemic may come to an end. One of the most important problems in epidemiology is to ascertain whether this termination occurs only when no susceptible individuals are left, or whether the interplay of the various factors of infectivity, recovery and mortality, may result in termination, whilst many susceptible individuals are still present in the unaffected population."

Following a general mathematical exposé, they suggested a set of pragmatic assumptions which lead to the standard SIR model of ordinary differential equations

---

This chapter uses the following R-packages: `deSolve`, `rootSolve`, `phaseR`, and `shiny`. A conceptual understanding of *reproductive ratios* and the *closed epidemic* is useful prior to this discussion. Five minute epidemics-MOOC introductions can be watched from YouTube:
Reproductive number https://www.youtube.com/watch?v=ju26rvzfFg4.
Closed epidemic https://www.youtube.com/watch?v=sSLfrSSmJZM.

O. N. Bjørnstad, *Epidemics*, Use R!, https://doi.org/10.1007/978-3-319-97487-3_2

for the flow of hosts between **S**usceptible, **I**nfectious, and **R**ecovered compartments. In modern notation, their simplest set of equations is (Fig. 2.1):

$$\frac{dS}{dt} = \mu(N - S) - \beta I \frac{S}{N} \tag{2.1}$$

$$\frac{dI}{dt} = \beta I \frac{S}{N} - (\mu + \gamma) I \tag{2.2}$$

$$\frac{dR}{dt} = \gamma I - \mu R. \tag{2.3}$$

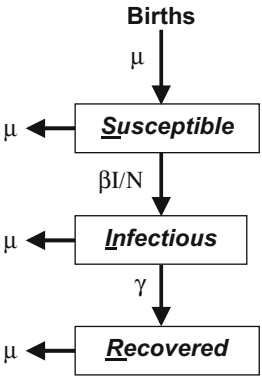

Fig. 2.1 The SIR flow diagram. Flows represent *per capita* flows from the donor compartments

The assumptions of Eqs. (2.1)–(2.3) are:

- The infection circulates in a population of size N, with a per capita "background" death rate, μ, which is balanced by a birth rate μN. From the sum of Eqs. (2.1)–(2.3), $dN/dt = 0$ and $N = S + I + R$ is thus constant.
- The infection causes acute morbidity (not mortality); That is, in this version of the SIR model we assume we can ignore disease-induced mortality. This is reasonable for certain infections like chickenpox, but certainly not for others like rabies, SARS, or ebola.
- Individuals are recruited directly into the susceptible class at birth (so we ignore perinatal maternal immunity).
- Transmission of infection from infectious to susceptible individuals is controlled by a bilinear contact term $\beta I \frac{S}{N}$. This stems from the assumption that the I infectious individuals are independently and randomly mixing with all other individuals, so the fraction S/N of the encounters is with susceptible individuals; β is the contact rate times the probability of transmission given a contact between a susceptible and an infectious individual.

- Chances of recovery or death is assumed not to change during the course of infection.
- Infectiousness is assumed not to change during the course of infection.
- Infected individuals move directly into the the infectious class (as opposed to the SEIR model; see Sect. 3.7) and remains there for an average infectious period of $1/\gamma$ (assuming $\mu << \gamma$).[1]
- The model assumes that recovered individuals are immune from reinfection for life.

The basic reproductive ratio (R_0), defined as the expected number of secondary infections from a single index case in a completely susceptible population, is a very important quantity in epidemiology. Chapter 3 is entirely devoted to this quantity. For this simple SIR model $R_0 = \frac{\beta}{\gamma+\mu}$.

2.2 Numerical Integration of the SIR Model

If there are no (or negligible) births and deaths during the duration of an epidemic ($\mu \simeq 0$), it is commonly referred to as a *closed epidemic*. While it is occasionally possible to derive analytical solutions to systems of ODEs like Eqs. (2.1)–(2.3), we generally have to resort to numerical integration to predict the numbers over time. We use the `deSolve` R-package to numerically integrate the equations. We will numerically integrate a variety of different models. While the models differ, the basic recipe is generally the same: (1) define a R-function for the general system of equations, (2) specify the time points at which we want the integrator to save the state of the system, (3) provide values for the parameters, (4) give initial values for all state variables, and finally (5) invoke the R-function that does the integration. We use the `ode`-function in the `deSolve`-package.

[1] The implicit assumptions that stem from the use of deterministic, ordinary differential equation (ODE) are that the infectious periods (and resident times in all compartments) are exponentially distributed. This is a tractable approximation for exploring overall dynamics, but observed duration of infection periods is often much less variable—the *Eimeria*-gut parasite (a relative of *Plasmodium* that cause malaria) undergoes exactly 8 replication cycles before leaving a host; or much more variable—see superspreader MOOC video: https://www.youtube.com/watch?v=3H1tG4uz9uk. Section 2.7 discusses a practical approach to model dynamics when the exponential assumption is deemed too simplistic.

```
require(deSolve)
```

Step 1: We define the function (often called the gradient-functions) for the equation systems. The `deSolve`-package requires the function to take the following parameters: time,[2] `t`, a vector with the values for the state variables (S, I, R), `y`, and parameter values (β, μ, γ, and N), `parms`:

```
sirmod = function(t, y, parms) {
    # Pull state variables from y vector
    S = y[1]
    I = y[2]
    R = y[3]
    # Pull parameter values from parms vector
    beta = parms["beta"]
    mu = parms["mu"]
    gamma = parms["gamma"]
    N = parms["N"]
    # Define equations
    dS = mu * (N - S) - beta * S * I/N
    dI = beta * S * I/N - (mu + gamma) * I
    dR = gamma * I - mu * R
    res = c(dS, dI, dR)
    # Return list of gradients
    list(res)
}
```

The `ode`-function solves differential equations numerically.

Steps 2–4: Specify the time points at which we want `ode` to record the states of the system (here we use 26 weeks with 10 time-increments per week as specified in the vector `times`), the parameter values (in this case as specified in the vector `parms`), and starting conditions (specified in `start`). In this case we model the *fraction* of individuals in each class, so we set $N = 1$, and consider a disease with an infectious period of 2 weeks ($\gamma = 1/2$), no births or deaths ($\mu = 0$) and a transmission rate of 2 ($\beta = 2$). For our starting conditions we assume that 0.1% of the initial population is infected and the remaining fraction is susceptible.

```
times = seq(0, 26, by = 1/10)
parms = c(mu = 0, N = 1, beta = 2, gamma = 1/2)
start = c(S = 0.999, I = 0.001, R = 0)
```

[2] Though, in the case of the simple SIR model there is no time-dependence in any of the parameters, so this parameter is not called within the gradient function; This will change when we consider seasonality (Chap. 5).

Step 5: Feed `start` values, `times`, the gradient-function and parameter vector to the `ode`-function as suggested by `args(ode)`.[3] For convenience we convert the output to a data frame (`ode` returns a `list`). The `head`-function shows the first 5 rows of `out`, and `round(,3)` rounds the number to three decimals.

```
out=ode(y=start, times=times, func=sirmod, parms=
    parms)
out=as.data.frame(out)
head(round(out, 3))
```

```
  ##   time     S     I R
  ## 1  0.0 0.999 0.001 0
  ## 2  0.1 0.999 0.001 0
  ## 3  0.2 0.999 0.001 0
  ## 4  0.3 0.998 0.002 0
  ## 5  0.4 0.998 0.002 0
  ## 6  0.5 0.998 0.002 0
```

We can plot the result (Fig. 2.2) to see that the model predicts an initial exponential growth of the epidemic that decelerates as susceptibles are depleted, and finally fade-out as susceptible numbers are too low to sustain the chain of transmission.

```
plot(x=out$time, y=out$S, ylab="Fraction", xlab=
    "Time", type="l")
lines(x=out$time, y=out$I, col="red")
lines(x=out$time, y=out$R, col="green")
```

R allows for a lot of customization of graphics—Rseek.org is a useful resource to find solutions to all things R... Fig. 2.2 has some added features such as a right-hand axis for the *effective* reproductive ratio (R_E)—the expected number of new cases per infected individuals in a *not* completely susceptible population—and a legend so that we can confirm that the turnover of the epidemic happens exactly when $R_E = R_0 s = 1$, where s is the fraction of remaining susceptibles. The threshold $R_0 s = 1 \Rightarrow s^* = 1/R_0$ results in the powerful rule of thumb for vaccine induced eradication and herd immunity: If we can—through vaccination —keep the susceptible population below a critical fraction, $p_c = 1 - 1/R_0$, then pathogen spread will dissipate and the pathogen will not be able to reinvade the host population (e.g., Anderson and May 1982; Roberts and Heesterbeek 1993; Ferguson et al. 2003). This rule of thumb appeared to work well for smallpox, the only vaccine-eradicated human disease; Its R_0 was commonly around 5, and most countries saw elimination once vaccine cover exceeded 80% (Anderson and May 1982). The actual code used to produce Fig. 2.2 is:

[3] For further details on usage, do `?function` on the R command-line, i.e., `?ode` in this instance.

```
#Calculate R0
R0=parms["beta"]/(parms["gamma"]+parms["mu"])

#Adjust margins to accommodate a second right axis
par(mar = c(5,5,2,5))
#Plot state variables
plot(x=out$time, y=out$S, ylab="Fraction", xlab="Time",
     type="l")
lines(x=out$time, y=out$I, col="red")
lines(x=out$time, y=out$R, col="green")
#Add vertical line at turnover point
xx=out$time[which.max(out$I)]
lines(c(xx,xx), c(1/R0,max(out$I)), lty=3)

#prepare to superimpose 2nd plot
par(new=TRUE)
#plot effective reproductive ratio  (w/o axes)
plot(x=out$time, y=R0*out$S, type="l", lty=2, lwd=2,
     col="black", axes=FALSE, xlab=NA, ylab=NA,
     ylim=c(-.5, 4.5))
lines(c(xx, 26), c(1,1), lty=3)
#Add right-hand axis for RE
axis(side = 4)
mtext(side = 4, line = 4, expression(R[E]))
#Add legend
legend("right", legend=c("S", "I", "R",
     expression(R[E])), lty=c(1,1,1, 2),
     col=c("black", "red", "green", "black"))
```

2.3 Final Epidemic Size

The closed epidemic model has two equilibria $\{S = 1, I = 0, R = 0\}$ which is unstable when $R_0 > 1$, and the $\{S^*, I^*, R^*\}$-equilibrium which reflects the *final epidemic size*, for which $I^* = 0$ as the epidemic eventually self-extinguish in the absence of susceptible recruitment; S^* is the fraction of susceptibles that escape infection altogether; and R^* is the *final epidemic size*—the fraction of susceptibles that will be infected before the epidemic self-extinguish. For the closed epidemic, there is an exact mathematical solution to the final epidemic size (below). It is nevertheless useful to consider computational ways of finding equilibria in the absence of exact solutions.

The `rootSolve`-package will attempt to find equilibria of systems of differential equations through numerical integration. The function `runsteady` is really just a wrapper function around the `ode`-function that integrates until the system settles on some steady-state (if it exists). It takes the same arguments as `ode`. By varying initial conditions `rootSolve` should find multiple *stable* equilibria if there are more than one stable solution.[4]

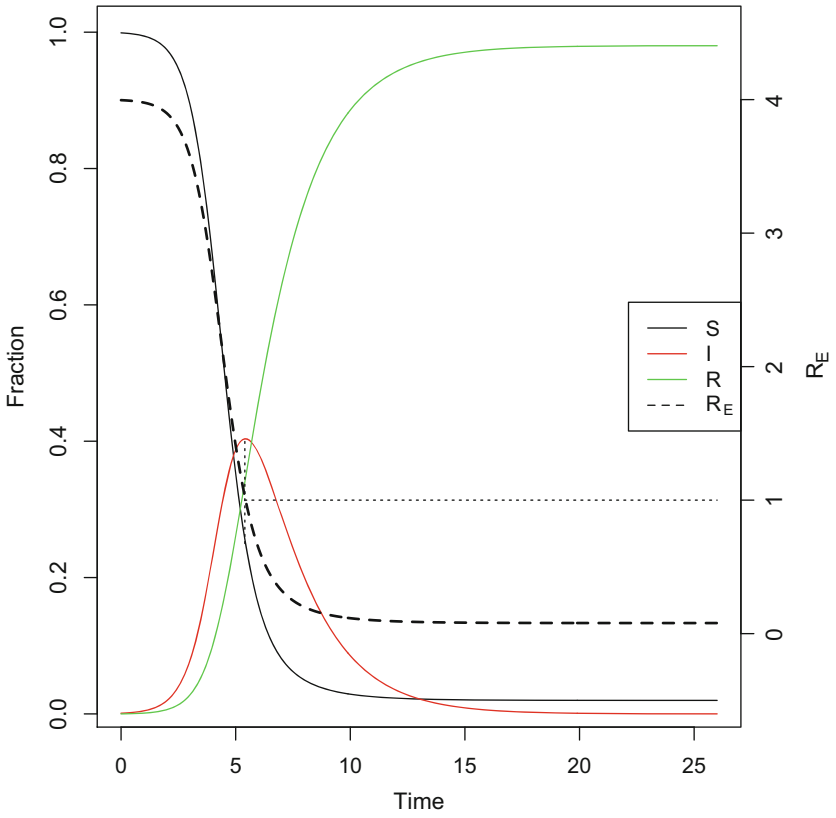

Fig. 2.2 The closed SIR epidemic with left and right axes and effective reproductive ratio, R_E. The epidemic turns over at $R_E = 1$

```
require(rootSolve)
equil=runsteady(y=c(S=1-1E-5, I=1E-5, R=0),
times=c(0,1E5), func=sirmod, parms=parms)
round(equil$y, 3)
```

[4] It will not find unstable equilibria, for these we will need to use other strategies. We will consider finding all equilibria in more depth in Sect. 9.3.

```
##    S    I    R
## 0.02 0.00 0.98
```

So for these parameters, 2% of susceptibles are expected to escape infection altogether and 98%—the final epidemic size—are expected to be infected during the course of the epidemic.

Let us explore numerically how the final epidemic size depends on R_0. Recall that for the specific SIR variant we are working with $R_0 = \beta/(\gamma+\mu)$, and since we are studying the closed epidemic $\mu = 0$. In the above example we assume an infectious period of 2 weeks (i.e., $\gamma = 1/2$), so we may vary β so R_0 goes from 0.1 to 5. For moderate to large R_0 this fraction has been shown to be approximately $1-\exp(-R_0)$ (e.g., Anderson and May 1982). We can check how well this approximation holds (Fig. 2.3).[5]

```
#Candidate values for R0 and beta
R0 = seq(0.1, 5, length=50)
betas= R0 * 1/2
#Vector of NAs to be filled with numbers
f = rep(NA, 50)
#Loop over i from 1, 2, ..., 50
for(i in seq(from=1, to=50, by=1)){
     equil=runsteady(y=c(S=1-1E-5, I=1E-5,
       R=0), times=c(0,1E5), func=sirmod,
       parms=c(mu=0, N=1, beta=betas[i], gamma=1/2))
     f[i]=equil$y["R"]
}
plot(R0, f, type="l", xlab=expression(R[0]))
curve(1-exp(-x), from=1, to=5, add=TRUE, col="red")
```

We see that the approximation is good for $R_0 > 2.5$ but overestimates the final epidemic size for smaller R_0 (and is terrible for $R_0 < 1$).

For the closed epidemic SIR model, there is an exact mathematical solution to the fraction of susceptibles that escapes infection $(1-f)$ given by the implicit equation $f = \exp(-R_0(1-f))$ or equivalently $\exp(-R_0(1-f)) - f = 0$ (Swinton 1998). So we can also find the final size by applying the `uniroot`-function to the equation. The `uniroot`-function finds numerical solutions to equations with one unknown variable (which has to be named `x`).

```
#Define function
fn=function(x, R0){
 exp(-(R0*(1-x))) - x
```

[5] We use a `for`-loop here to calculate the final epidemic size for a range of values of R_0; A loop works by repeating calculations (in this case 50 times), after each repeat the value of the looping variable (in this case `i`) is changed to the next value in the looping vector. So in this example `i` will be 1 first, then 2, then ... until the loop ends after `i`= 50.

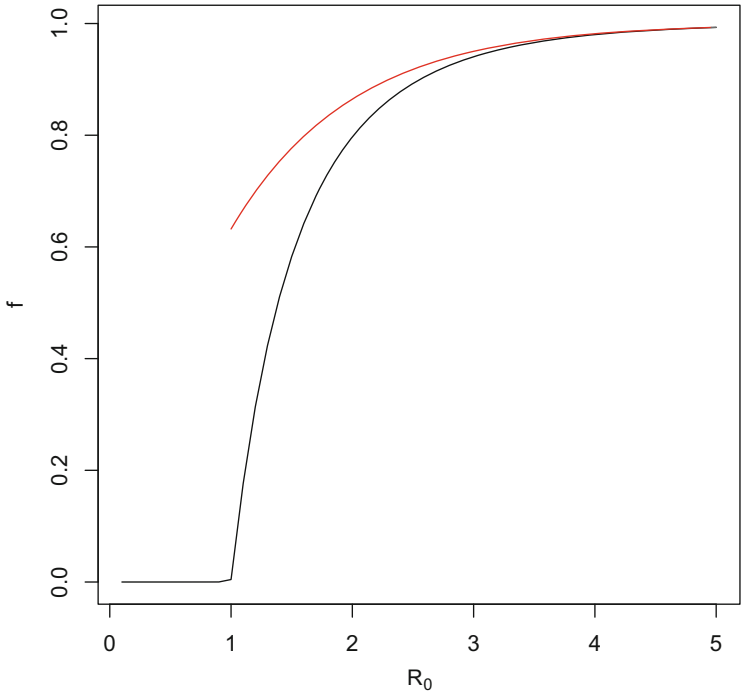

Fig. 2.3 The final epidemic size as a function of R_0. The black line is the solution based on numerically integrating the closed epidemic, and the red line is the approximation $f \simeq 1 - \exp(-R_0)$

```
}
1-uniroot(fn, lower = 0, upper = 1-1E-9,
      tol = 1e-9, R0=2)$root

  ## [1] 0.7968121

#check accuracy of approximation:
exp(-2)-uniroot(fn, lower = 0, upper = 1-1E-9,
      tol = 1e-9, R0=2)$root

  ## [1] -0.06785259
```

So for $R_0 = 2$ the final epidemic size is 79.6% and the approximation is off by around 6.7% points.

2.4 Open Epidemic

The *open epidemic* has recruitment of new susceptibles (i.e., $\mu > 0$). As long as $R_0 > 1$, the open epidemic has an "endemic equilibrium" were the pathogen and host coexist. If we use the SIR equations to model fractions (i.e., set $N = 1$), Eq. (2.2) of the SIR model implies that $S^* = (\gamma + \mu)/\beta = 1/R_0$ is the endemic S-equilibrium, which when substituted into Eq. (2.1) gives $I^* = \mu(R_0 - 1)/\beta$, and finally, $R^* = N - I^* - S^*$ as the I and R endemic equilibria. We can study the predicted dynamics of the open epidemic using the `sirmod`-function. Let us assume a life expectancy of 50 years, a stable population size, and thus a weekly birth rate of $\mu = 1/(50 * 52)$. Let's assume that 19% of the initial population is susceptible and 1% is infected and numerically integrate the model for 50 years (Fig. 2.4).

```
times  = seq(0, 52*50, by=.1)
parms  = c(mu = 1/(50*52), N = 1, beta =  2,
     gamma = 1/2)
start = c(S=0.19, I=0.01, R = 0.8)
out = as.data.frame(ode(y=start, times=times,
     func=sirmod, parms=parms))
par(mfrow=c(1,2)) #Make room for side-by-side plots
plot(times, out$I, ylab="Fraction", xlab="Time",
     type="l")
plot(out$S, out$I, type="l", xlab="Susceptible",
     ylab="Infected")
```

2.5 Phase Analyses

When working with dynamical systems we are often interested in studying the dynamics in the phase plane and derive the isoclines that divide this plane in regions of increase and decrease of the various state variables. The `phaseR` package is a wrapper around `ode` that makes it easy to analyze 1D and 2D ode's.[6] The R-state in the SIR model does not influence the dynamics, so we can rewrite the SIR model as a 2D system.

```
simod = function(t, y, parameters) {
    S = y[1]
    I = y[2]
```

[6] The `phaseR` package requires the gradient function to take the arguments `t`, `y`, and `parameters`.

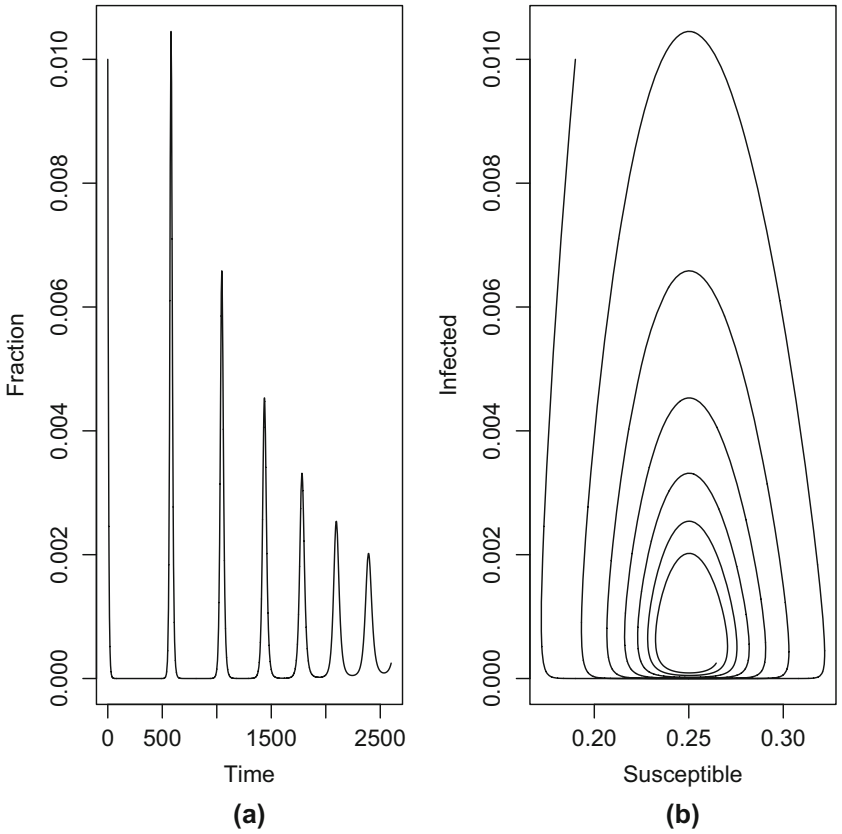

Fig. 2.4 The open SIR epidemic. (**a**) The fraction infected over time. (**b**) The joint time series of infecteds and susceptibles in the S-I phase plane. The trajectory forms a counter-clockwise inwards spiral in the S-I plane (note that the 50-year simulation is not long enough for the system to reach the steady-state endemic equilibrium at the center of the spiral)

```
    beta = parameters["beta"]
    mu = parameters["mu"]
    gamma = parameters["gamma"]
    N = parameters["N"]

    dS = mu * (N - S) - beta * S * I/N
    dI = beta * S * I/N - (mu + gamma) * I
    res = c(dS, dI)
    list(res)
}
```

The isoclines (sometimes called the nullclines) in this system are given by the solution to the equations $dS/dt = 0$ and $dI/dt = 0$ and partitions the phase plane into regions were S and I are increasing and decreasing. For $N = 1$, the I-isocline is $S = (\gamma+\mu)/\beta = 1/R_0$ and the S-isocline is $I = \mu(1/S-1)/\beta$. We can draw these in the phase plane and add a simulated trajectory to the plot (Fig. 2.5). The trajectory cycles in a counter-clockwise dampened fashion towards the endemic equilibrium (Fig. 2.5). To visualize the expected change to the system at arbitrary points in the phase plane, we can further use the function `flowField` in the `phaseR`-package to superimpose predicted arrows of change (vectors).

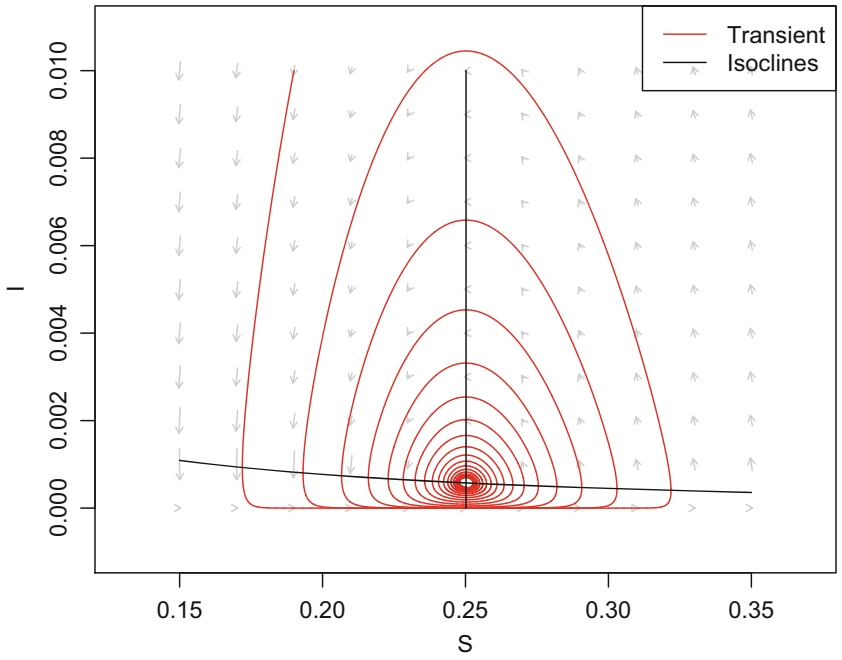

Fig. 2.5 The S-I phase plane with isoclines and the predicted anti-clockwise trajectory towards the equilibrium

```
require(phaseR)
#Plot vector field
fld=flowField(simod, x.lim=c(0.15,0.35), y.lim=c(0,.01),
     parameters=parms, system="two.dim", add=FALSE,
     ylab="I", xlab="S")
#Add trajectory
out = as.data.frame(ode(y = c(S=0.19, I=0.01), times=
     seq(0, 52*100, by=.1), func=simod, parms=parms))
 lines(out$S, out$I, col="red")
```

```
#Add S-isocline
curve(parms["mu"]*(1/x-1)/parms["beta"], 0.15, 0.35,
     xlab="S", ylab="I", add=TRUE)
#Add I-isocline
shat=(parms["gamma"]+parms["mu"])/parms["beta"]
lines(rep(shat, 2), c(0,0.01))

legend("topright", legend=c("Transient", "Isoclines"),
     lty=c(1, 1), col=c("red", "black"))
```

2.6 Stability and Periodicity

If we work with continuous-time ODE models like the SIR, equilibria are locally stable if (and only if) all the real part of the eigenvalues of the Jacobian matrix—when evaluated at the equilibrium—are smaller than zero. We will discuss stability and resonant periodicity in detail in Chap. 9, so this section is just a teaser... An equilibrium is (1) a node (i.e., all trajectories moves monotonically towards/away from the equilibrium) if the largest eigenvalue has only real parts, or (2) a focus (trajectories spiral towards or away from the equilibrium) if the largest eigenvalues are a conjugate pair of complex numbers ($a \pm b\imath$).[7] For a focus the imaginary part determines the dampening period of the cycle according to $2\pi/b$. We can thus use the Jacobian matrix to study the SIR model's equilibria. If we let $F = dS/dt = \mu(N-S) - \beta SI/N$ and $G = dI/dt = \beta SI/N - (\mu+\gamma)I$, the Jacobian of the SIR system is

$$J = \begin{pmatrix} \frac{\partial F}{\partial S} & \frac{\partial F}{\partial I} \\ \frac{\partial G}{\partial S} & \frac{\partial G}{\partial I} \end{pmatrix}, \tag{2.4}$$

and the two equilibria are the disease-free equilibrium and the endemic equilibrium as defined above.

R can help with all of this. We first calculate the equilibria:

```
# Pull values from parms vector
gamma = parms["gamma"]
beta = parms["beta"]
mu = parms["mu"]
N = parms["N"]


# Endemic equilibrium
```

[7] And a center—like the Lotka-Volterra predator-prey model—if conjugate pair only has imaginary parts.

```
Sstar = (gamma + mu)/beta
Istar = mu * (beta/(gamma + mu) - 1)/beta
eq1 = list(S = Sstar, I = Istar)
```

We then calculate the elements of the Jacobian using R's `D`-function:

```
# Define equations
dS = expression(mu * (N - S) - beta * S * I/N)
dI = expression(beta * S * I/N - (mu + gamma) * I)
# Differentiate w.r.t. S and I
j11 = D(dS, "S")
j12 = D(dS, "I")
j21 = D(dI, "S")
j22 = D(dI, "I")
```

We pass the values for S^* and I^* in the `eq1`-list to the Jacobian,[8] and use `eigen`-function to calculate the eigenvalues:

```
#Evaluate Jacobian  at equilibrium
J=with(data=eq1, expr=matrix(c(eval(j11),eval(j12),
     eval(j21),eval(j22)), nrow=2, byrow=TRUE))
#Calculate eigenvalues
eigen(J)$values

  ## [1] -0.00076864+0.02400384i -0.00076864-0.02400384i
```

For the endemic equilibrium, the eigenvalues are a pair of complex conjugates which real parts are negative, so it is a stable focus. The period of the inwards spiral is:

```
2 * pi/(Im(eigen(J)$values[1]))

  ## [1] 261.7575
```

So with these parameters the dampening period is predicted to be 261 weeks (just over 5 years). Thus, during disease invasion we expect this system to exhibit initial outbreaks every 5 years. A further significance of this number is that if the system is stochastically perturbed by, say, environmental variability affecting transmission, we expect the system to exhibit low amplitude "phase-forgetting" cycles (Nisbet

[8] In previous coding like for the `sirmod`-function, we "pulled" parameter values from the input arguments inside the function to make the code as transparent as possible; while it makes the code easy to read, it makes for extra coding, and can clutter up the workspace with variables that are defined in multiple locations. The `with`-function allows the evaluation of an expression using variables defined in a data list.

and Gurney 1982) with approximately this period in the long run (see Chap. 9). We can make more accurate calculations of the stochastic system using *transfer functions* (Nisbet and Gurney 1982; Priestley 1981). We will visit on this slightly more advanced topic in Sect. 9.7.

The same protocol can be used on the disease-free equilibrium $\{S^* = 1, I^* = 0\}$.

```
eq2=list(S=1,I=0)
J=with(eq2,
     matrix(c(eval(j11),eval(j12),eval(j21),
     eval(j22)), nrow=2, byrow=TRUE))
eigen(J)$values

  ## [1]  1.4996153846 -0.0003846154
```

The eigenvalues are strictly real and the largest value is greater than zero, so it is an unstable node (a "saddle"); The epidemic trajectory is predicted to move monotonically away from this disease free equilibrium if infection is introduced into the system. This makes sense because with the parameter values used, $R_0 = 3.99$ which is greater than the invasion-threshold value of 1.

2.7 Advanced: More Realistic Infectious Periods

The S(E)IR-type differential equation models assumes that rate of exit from the infectious classes are constant, the implicit assumption is thus that the infectious period is exponentially distributed among infected individuals; The average infectious period is $1/(\gamma+\mu)$, but an exponential fraction is infectious much shorter/longer than this. The chain-binomial model (see Sect. 3.4), in contrast, assumes that everybody is infectious for a fixed period and then all instantaneously recover (or die). These assumptions are mathematically convenient, but in reality neither are particularly realistic. Hope-Simpson (1952) traced the chains of transmission of measles in multi-sibling household. The timing of secondary and tertiary cases was analyzed in detail by Bailey (1956) and Bailey and Alff-Steinberger (1970). The average latent and infectious periods were calculated to be 8.58 and 6.57 days, respectively. While the distribution around each of these averages were not estimated separately (the latent period was assumed to be distributed and the infectious period assumed fixed), the variance around the roughly fortnight period of infection was estimated to be 3.13. The mean duration of infection is thus 15.15 days with a standard deviation of 1.77 (Fig. 2.6). So neither a fixed nor an exponential distribution is very accurate (Keeling and Grenfell 1997; Lloyd 2001).

Kermack and McKendrick's (1927) original model allows for arbitrary infectious-period distributions. We can write Kermack and McKendrick's origi-

nal equations as "renewal equations" (Breda et al. 2012), introducing the additional notation of $k(t)$ being the (instantaneous) *incidence* at time t (i.e., flux into the I-class at time t).

$$\frac{dS}{dt} = \mu(N - S) - k(t) \tag{2.5}$$

$$k(t) = \beta I \frac{S}{N} \tag{2.6}$$

$$\frac{dI}{dt} = k(t) - \mu I - \int_0^\infty \frac{h(\tau)}{1 - H(\tau)} k(t - \tau) d\tau \tag{2.7}$$

$$\frac{dR}{dt} = \int_0^\infty \frac{h(\tau)}{1 - H(\tau)} k(t - \tau) d\tau - \mu R, \tag{2.8}$$

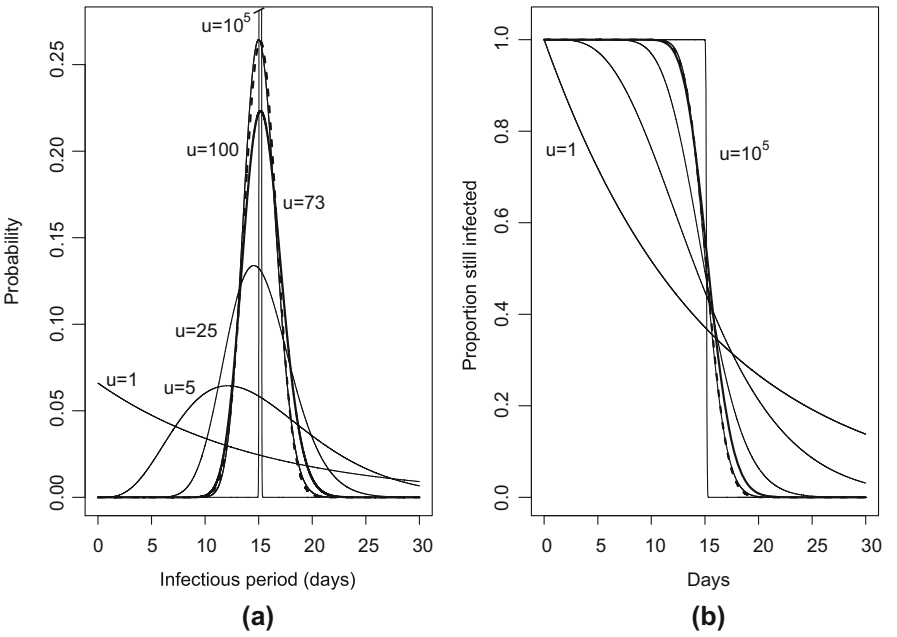

Fig. 2.6 Gamma distributed infectious periods: (**a**) The predicted infectious period distribution based on a Gamma distribution with shape $u = 1$, 5, 25, 100, and 100,000; $u = 1$ corresponds to the exponential distribution implicit in the standard SIR model; the bold line ($u = 73$) is the one corresponding to the variance observed in Hope-Simpson's (1952) study of measles. The dotted line (virtually indistinguishable from the $u = 100$) is a Gaussian distribution intended to show that when u is large the Gamma distribution converges on the Gaussian. (**b**) The probability of still being infectious as a function of time for the different distributions; as u becomes large, the distribution converges on a fixed infectious period. Note that the empirical distribution (bold) is quite different from the exponential

where $k(t-\tau)$ is the number of individuals that was infected τ time units ago, $h(\tau)$ is the probability of recovering on infection-day τ, and $H(\tau)$ is the cumulative probability of having recovered by infection-day τ; $k(t-\tau)/(1-H(\tau))$ is thus the fraction of individuals infected at time $t-\tau$ that still remains in the infected class on day t and the integral is over all previous infections so as to quantify the total flux into the removed class at time t. Though intuitive, these general integro-differential equations (Eqs. (2.5)–(2.8)) are not easy to work with in general. For a restricted set of distributions for the $h()$-function, however—the Erlang distribution (the Gamma distribution with an integer shape parameter)—the model can be numerically integrated using a "Gamma-chain" model (referred to as "linear chain trickery" by Metz and Diekmann 1991) of coupled ordinary differential equations (e.g., Blythe et al. 1984; de Valpine et al. 2014; Bjørnstad et al. 2016). The trick is to separate any distributed-delay compartment into u sub-compartments through which individuals pass through at a rate overallrate $*u$. The resultant infectious period will have a mean of $1/$overallrate and a coefficient-of-variation of $1/\sqrt{u}$.

We can write a chain-SIR model to simulate $S \to I \to R$ flows with more realistic infectious period distributions[9]:

```
chainSIR=function(t, logx, params){
x=exp(logx)
u=params["u"]
S=x[1]
I=x[2:(u+1)]
R=x[u+2]
with(as.list(params),{
  dS = mu * (N - S)  - sum(beta * S * I) / N
  dI = rep(0, u)
  dI[1] = sum(beta * S * I) / N - (mu + u*gamma) * I[1]
  if(u>1){
     for(i in 2:u){
         dI[i]= u*gamma * I[i-1] - (mu+u*gamma)* I[i]
      }
   }
  dR = u*gamma * I[u] - mu * R
  res=c(dS/S, dI/I, dR/R)
  list(res)
})
}
```

[9] With high number of compartments this system of equations can become "stiff" with the computer potentially making rounding errors leading to erroneous negative numbers. We use a "log-trick" (Ellner and Guckenheimer 2011) available for systems where all state variables are strictly positive: we solve the system in log-coordinates to smooth abrupt changes and force all number to be greater than zero. To employ this technique we log-transform all initial values in `start`, change the first line in the function to `x = exp(logx)` and the last line to return `dS/S`, etc. in place of `dS` which comes from the chain-rule of differentiation and the fact that $D(\log x) = 1/x$.

We can compare the predicted dynamics of the simple SIR model with the $u = 2$ chain model, the $u = 500$ chain model (which is effectively the fixed-period delayed-differential model) and the "measles-realistic" $u = 73$ model.

```
times = seq(0, 10, by=1/52)
paras2 = c(mu = 1/75, N = 1, beta =  625,
     gamma = 365/14, u=1)
xstart2 = log(c(S=.06, I=c(0.001, rep(0.0001,
     paras2["u"]-1)), R = 0.0001))
out = as.data.frame(ode(xstart2, times, chainSIR,
     paras2))
plot(times, exp(out[,3]), ylab="Infected", xlab=
     "Time", ylim=c(0, 0.01), type='l')

paras2["u"] =2
xstart2 = log(c(S=.06, I=c(0.001, rep(0.0001/
     paras2["u"], paras2["u"]-1)), R = 0.0001))
out2 = as.data.frame(ode(xstart2, times, chainSIR,
     paras2))
lines(times, apply(exp(out2[,-c(1:2,length(out2))]),
     1 ,sum), col='blue')

paras2["u"] =73
xstart2 = log(c(S=.06, I=c(0.001, rep(0.0001/
     paras2["u"], paras2["u"]-1)), R = 0.0001))
out3 = as.data.frame(ode(xstart2, times, chainSIR,
     paras2))
lines(times, apply(exp(out3[,-c(1:2,length(out3))]),
     1, sum), col='red', lwd=2, lty=2)

paras2["u"] =500
xstart2 = log(c(S=.06, I=c(0.001, rep(0.0001/
     paras2["u"], paras2["u"]-1)), R = 0.0001))
out4 = as.data.frame(ode(xstart2, times, chainSIR,
     paras2))
lines(times, apply(exp(out4[,-c(1:2,length(out4))]),
     1,sum, na.rm=TRUE), col='green')

legend("topright", legend=c("SIR", "u=2", "u=500",
     "u=73 (H-S)"), lty=c(1,1,1,2), lwd=c(1,1,1, 2),
     col=c("black", "blue", "green", "red"))
```

The more narrow the infectious-period distribution, the more punctuated the predicted epidemics. However, infectious-period narrowing—alone—cannot sustain recurrent epidemics; In the absence of stochastic or seasonal forcing epidemics will

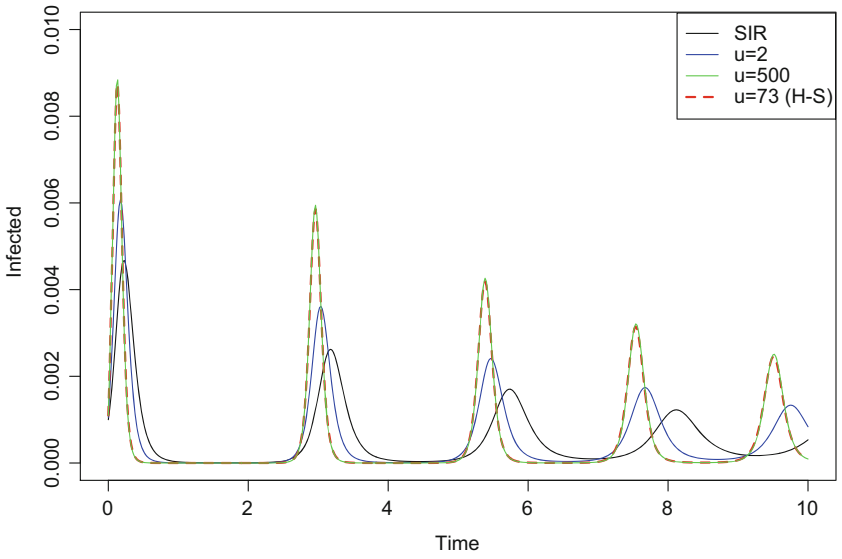

Fig. 2.7 Chain-SIR models with different infectious period distributions

dampen to the endemic equilibrium (though the damping period is slightly accelerated and the convergence on the equilibrium is slightly slower with narrowing infectious period distributions) (Fig. 2.7).

In the above we considered non-exponential infectious-period distributions. However, the general ODE chain method can be used for any compartment. Lavine et al. (2011), for example, used it to model non-exponential waning of natural and vaccine-induced immunity to whooping cough.

2.8 ShinyApp

The following code will launch a local shinyApp of the SIR model in your local browser. This App can also be launched by calling `SIR.app` in the `epimdr`-package. Several of the subsequent chapters also have associated shinyApps. Those will only be accessible from the package (because the code is long and a bit tedious). We quote an annotated version of the `SIR.app` in full.

```
require(shiny)
require(deSolve)
require(phaseR)

#This creates the User Interface (UI)
ui = pageWithSidebar(
```

```
#The title
headerPanel("The SIR  model"),
#The sidebar for parameter input
sidebarPanel(
   #Sliders:
   sliderInput("beta", "Transmission (yr^-1):", 300,
               min = 0, max = 1000),
   sliderInput("infper", "Infectious period (days)", 5,
               min = 1, max = 100),
   sliderInput("mu", "birth rate:", 5,
               min = 0, max = 100),
   sliderInput("T", "Time range:",
                  min = 0, max = 1, value = c(0,1))
),
#Main panel for figures and equations
mainPanel(
  #Multiple tabs in main panel
  tabsetPanel(
    #Tab 1: Time plot (plot1 from server)
    tabPanel("Time", plotOutput("plot1")),
    #Tab 2: Phase plot (plot2 from server)
    tabPanel("Phase plane", plotOutput("plot2",
         height = 500)),
    #Tab 3: MathJax typeset equations
    tabPanel("Equations",
       withMathJax(
         helpText("Susceptible $$\\frac{dS}{dt} =
            \\mu (N - S) - \\frac{\\beta I S}{N}$$"),
         helpText("Infecitous $$\\frac{dI}{dt} =
            \\frac{\\beta I S}{N} - (\\mu+\\sigma) I$$"),
         helpText("Removed $$\\frac{dR}{dt} =
            \\gamma I - \\mu R$$"),
         helpText("Reproductive ratio $$R_0 =
            \\frac{1}{\\gamma+\\mu} \\frac{\\beta N}{N}$$")
         ))
))) #End of ui()

# This creates the 'behind the scenes' code (Server)
server = function(input, output) {
  #Gradient function for SIR  model
  sirmod=function(t, x, parms){
    S=x[1]
    I=x[2]
    R=x[3]
```

```
  beta=parms["beta"]
  mu=parms["mu"]
  gamma=parms["gamma"]
  N=parms["N"]
  dS = mu * (N  - S)  - beta * S * I / N
  dI = beta * S * I / N - (mu + gamma) * I
  dR = gamma * I - mu * R
  res=c(dS, dI, dR)
  list(res)
 }

#Gradient function used for phaseR phase-plot
simod=function(t, y, parameters){
 S=y[1]
 I=y[2]
 beta=parameters["beta"]
 mu=parameters["mu"]
 gamma=parameters["gamma"]
 N=parameters["N"]
 dS = mu * (N  - S)  - beta * S * I / N
 dI = beta * S * I / N - (mu + gamma) * I
 res=c(dS, dI)
 list(res)
}

#Plot1: renderPlot to be passed to UI tab 1
output$plot1 = renderPlot({
#input\$xx's are pulled from UI
times  = seq(0, input$T[2], by=1/1000)
parms  = c(mu = input$mu, N = 1, beta =  input$beta,
  gamma = 365/input$infper)
start = c(S=0.999, I=0.001, R = 0)
R0 = round(with(as.list(parms), beta/(gamma+mu)), 1)

#Integrate ode with parameters pulled from UI
out=ode(y=start, times=times, func=sirmod,
  parms=parms)
out=as.data.frame(out)

#Plot1
sel=out$time>input$T[1]&out$time<input$T[2]
plot(x=out$time[sel], y=out$S[sel], ylab="fraction",
  xlab="time", type="l", ylim=range(out[sel,-c(1,4)]))
title(paste("R0=", R0))
```

```
  lines(x=out$time[sel], y=out$I[sel], col="red")
  lines(x=out$time[sel], y=out$R[sel], col="green")
  legend("right", legend=c("S", "I", "R"),
         lty=c(1,1,1), col=c("black", "red", "green"))
   })

  #Plot2: renderPlot to be passed to UI tab 2
  output$plot2 = renderPlot({
  times  = seq(0, input$T[2], by=1/1000)
  parms  = c(mu = input$mu, N = 1, beta =  input$beta,
    gamma = 365/input$infper)
  start = c(S=0.999, I=0.001, R = 0)
  R0 = round(with(as.list(parms), beta/(gamma+mu)), 1)

  #Integrate simod
  out=ode(y=start[-3], times=times, func=simod,
    parms=parms)
  out=as.data.frame(out)

  #Plot2
  plot(x=out$S, y=out$I, xlab="Fraction suceptible",
    ylab="Fraction infected", type="l")
  title(paste("R0=", R0))
  #Add vector field
  fld=flowField(simod, x.lim=range(out$S), y.lim=
    range(out$I), parameters=parms, system="two.dim",
    add=TRUE, ylab="I", xlab="S")
  #Add isoclines
  abline(v=1/R0, col="green")
  curve(parms["mu"]*(1-x)/(parms["beta"]*x), min(out$S),
      max(out$S), add=TRUE, col="red")
  legend("topright",  legend=c("I-socline",
      "S-isocline"), lty=c(1,1), col=c("red", "green"))
   })
  } #End of server()

shinyApp(ui, server)
```