Bayesian workflow for disease transmission modeling in Stan

Eustat – XXXIII International Statistical Seminar

Julien Riou, MD PhD
Institute of Social and Preventive Medicine, University of Bern, Switzerland

Preface

- Objective: fit transmission models in Stan
- Based on Grinsztajn et al., 2020 (link)
- Prerequisites:
  - general understanding of Bayesian inference
  - basic programming with `R` and `Stan`
- All material is available on
  `https://github.com/jriou/bayesian_workflow`

Outline

- Introduction
- (Quick notice: Bayesian inference with Stan)
- Simple SIR
- Using simulated data
- Scaling up ODE-based models
- Extensions

Ref: Grinsztajn et al. (2020)

UNIVERSITÄT BERN

Introduction

Models of disease transmission:

- Interpretability: mechanistic, phenomenological

UNIVERSITÄT BERN

Introduction

Models of disease transmission:

- Interpretability: mechanistic, phenomenological
- Data-generating mechanisms: incubation, contagion, immunity...

Introduction

Models of disease transmission:

- Interpretability: mechanistic, phenomenological
- Data-generating mechanisms: incubation, contagion, immunity...
- Scale: agent-based, population-based

Introduction

Models of disease transmission:

- Interpretability: mechanistic, phenomenological
- Data-generating mechanisms: incubation, contagion, immunity...
- Scale: agent-based, population-based
- Framework: deterministic, stochastic

Introduction

Models of disease transmission:

- Interpretability: mechanistic, phenomenological
- Data-generating mechanisms: incubation, contagion, immunity...
- Scale: agent-based, population-based
- Framework: deterministic, stochastic

Mechanistic + population-based + deterministic

$\rightarrow$ ODE-based compartmental model (e.g., SIR)

Introduction

ODE-based compartmental model:

- Divide the population into homogeneous groups (compartments)

Introduction

ODE-based compartmental model:

- Divide the population into homogeneous groups (compartments)
- Define flows between compartments with differential equations

Introduction

ODE-based compartmental model:

- Divide the population into homogeneous groups (compartments)
- Define flows between compartments with differential equations
- Define initial conditions

Introduction

ODE-based compartmental model:

- Divide the population into homogeneous groups (compartments)
- Define flows between compartments with differential equations
- Define initial conditions
- Solve for the time-dependent volume in each compartment

Introduction

ODE-based compartmental model:

- Divide the population into homogeneous groups (compartments)
- Define flows between compartments with differential equations
- Define initial conditions
- Solve for the time-dependent volume in each compartment

$$\begin{tikzcd} S \arrow[r, "\beta S \frac{I}{N}"] & I \arrow[r, "\gamma I"] & R \end{tikzcd}$$

$$\frac{dS}{dt} = -\beta S \frac{I}{N}$$

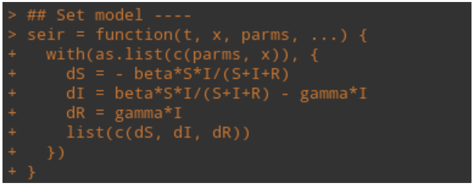
$$\frac{dI}{dt} = \beta S \frac{I}{N} - \gamma I$$

$$\frac{dR}{dt} = \gamma I$$

Introduction

Simulate in `R` with package `deSolve`:

- set compartments and differential equations

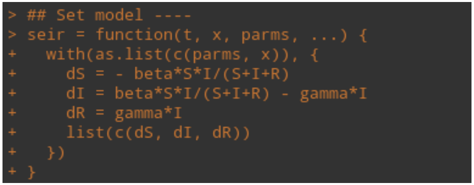


```
> ## Set model ----
> seir = function(t, x, parms, ...) {
+   with(as.list(c(parms, x)), {
+     dS = - beta*S*I/(S+I+R)
+     dI = beta*S*I/(S+I+R) - gamma*I
+     dR = gamma*I
+     list(c(dS, dI, dR))
+   })
+ }
```

Introduction

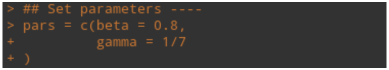
Simulate in `R` with package `deSolve`:

- set compartments and differential equations

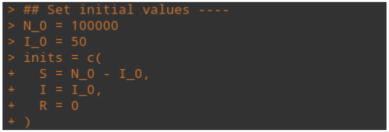


```
> ## Set model ----
> seir = function(t, x, parms, ...) {
+   with(as.list(c(parms, x)), {
+     dS = - beta*S*I/(S+I+R)
+     dI = beta*S*I/(S+I+R) - gamma*I
+     dR = gamma*I
+     list(c(dS, dI, dR))
+   })
+ }
```

- set (fixed) values for β, ρ and initial conditions
  $\beta = 0.8; \gamma = 1/7; S(0) = 100,000 - 50; I(0) = 50; R(0) = 0$



```
> ## Set parameters ----
> pars = c(beta = 0.8,
+          gamma = 1/7
+ )
```

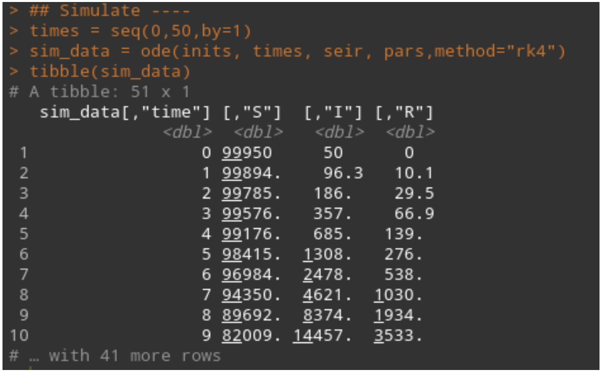


```
> ## Set initial values ----
> N_0 = 100000
> I_0 = 50
> inits = c(
+   S = N_0 - I_0,
+   I = I_0,
+   R = 0
+ )
```

u^b UNIVERSITÄT BERN

Introduction

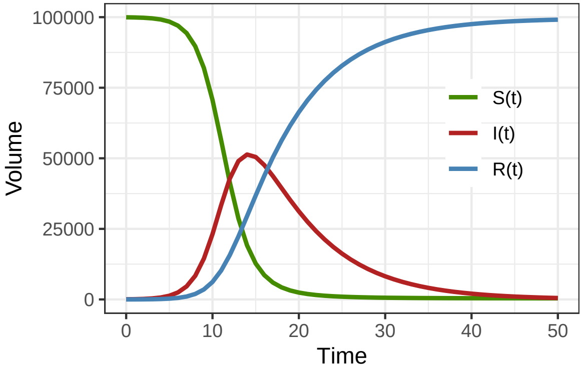
- solve the ODE system numerically (Runge-Kutta 4th order)



```
> ## Simulate ----
> times = seq(0,50,by=1)
> sim_data = ode(inits, times, seir, pars,method="rk4")
> tibble(sim_data)
# A tibble: 51 x 1
   sim_data[,"time"] [,"S"]  [,"I"] [,"R"]
               <dbl>  <dbl>   <dbl>  <dbl>
 1                 0 99950      50      0
 2                 1 99894.     96.3   10.1
 3                 2 99785.    186.    29.5
 4                 3 99576.    357.    66.9
 5                 4 99176.    685.   139.
 6                 5 98415.   1308.   276.
 7                 6 96984.   2478.   538.
 8                 7 94350.   4621.  1030.
 9                 8 89692.   8374.  1934.
10                 9 82009.  14457.  3533.
# … with 41 more rows
```

Introduction

- we obtain (deterministic) values for $S(t)$, $I(t)$ and $R(t)$

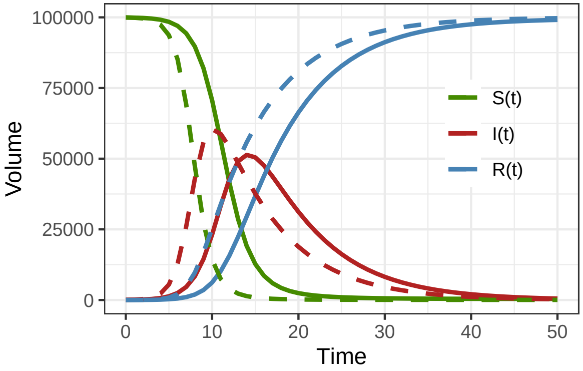


- these quantities have real-world interpretations (respectively susceptibility, prevalence, and cumulative attack rate)

Introduction

- the result is entirely determined by the chosen values for β, ρ and the initial conditions

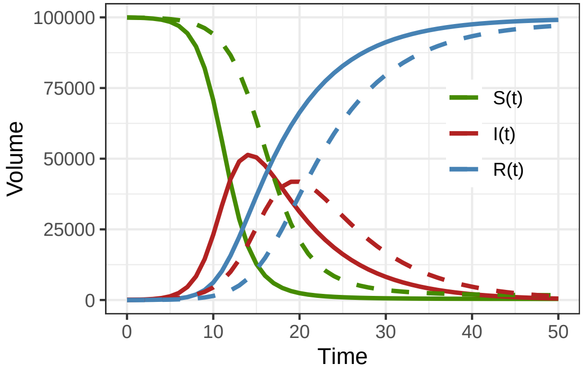
with $\beta = 1.1$ instead of 0.8, we get



Introduction

- the result is entirely determined by the chosen values for β, ρ and the initial conditions

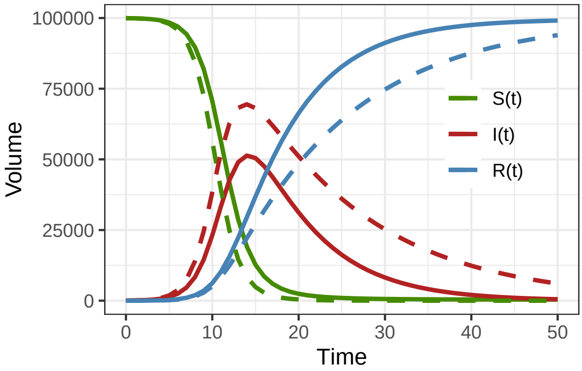
with $\beta = 0.6$ instead of 0.8, we get



Introduction

- the result is entirely determined by the chosen values for β, ρ and the initial conditions

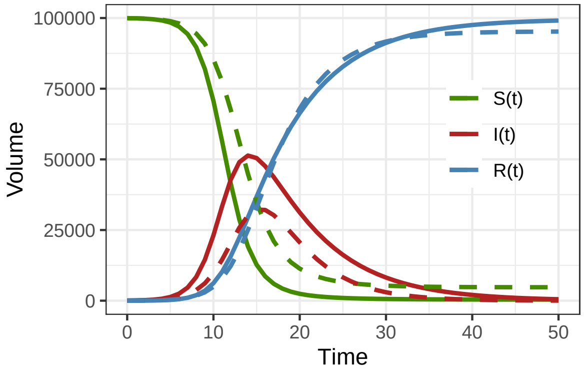
with $\gamma = 1/14$ instead of $1/7$, we get



Introduction

- the result is entirely determined by the chosen values for β, ρ and the initial conditions

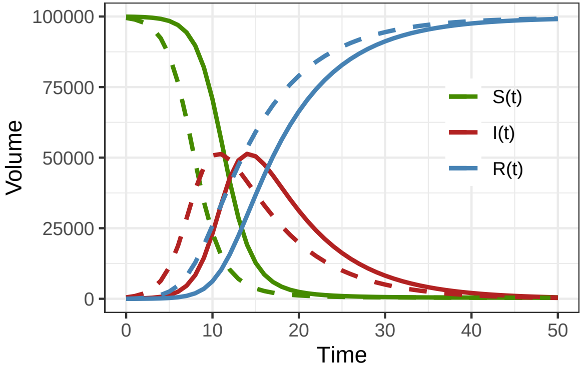
with $\gamma = 1/4$ instead of $1/7$, we get



Introduction

- the result is entirely determined by the chosen values for β, ρ and the initial conditions

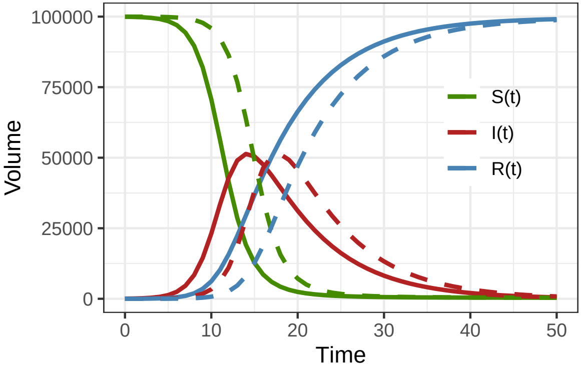
with $I(0) = 500$ instead of 50, we get



Introduction

- the result is entirely determined by the chosen values for β, ρ and the initial conditions

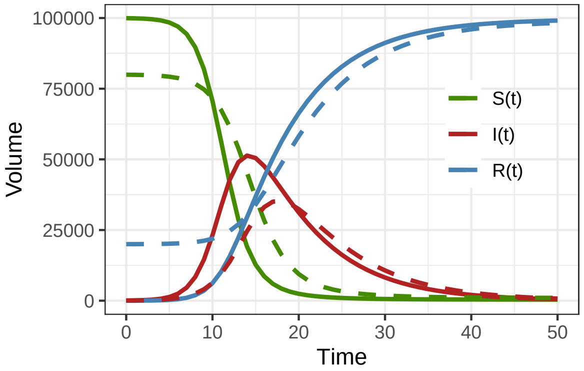
with $I(0) = 5$ instead of 50, we get



Introduction

- the result is entirely determined by the chosen values for β, ρ and the initial conditions

with $R(0) = 20,000$ instead of 0, we get



Introduction

Compartmental models have many uses:

- formalize and quantify general concepts (herd immunity, vaccination threshold...)
- get mechanistic insight about an epidemic ($\mathcal{R}_0$, $\mathcal{R}_t$, impact of interventions...)
- produce forecasts (based on mechanisms)

Introduction

Compartmental models have many uses:

- formalize and quantify general concepts (herd immunity, vaccination threshold...)
- get mechanistic insight about an epidemic ($\mathcal{R}_0$, $\mathcal{R}_t$, impact of interventions...)
- produce forecasts (based on mechanisms)

$\rightarrow$ based on numerical values for β, ρ and the initial conditions

Introduction

Enters Bayesian inference:

- make the best use of information from data
- easily incorporate prior knowledge
- infer the value (posterior) of the parameters
- propagate uncertainty from data and priors

Introduction

Enters Bayesian inference:

- make the best use of information from data
- easily incorporate prior knowledge
- infer the value (posterior) of the parameters
- propagate uncertainty from data and priors

$\rightarrow$ Markov Chain Monte Carlo methods and Stan

(Bayesian inference with Stan)

General principle of Bayesian inference:

- specify a complete Bayesian model
  - consider data $y = \{y_1, ..., y_n\}$ and parameter θ
  - specify an observation model

$$\Pr(y|\theta) = \prod_n \text{normal}(y_n|\theta, 1)$$

  - complete the model with a prior on the parameter

$$\Pr(\theta) = \text{normal}(0, 1)$$

- estimate the joint probability density function of the model

UNIVERSITÄT BERN

(Bayesian inference with Stan)

The joint probability density function of the model is given by

$$\Pr(y, \theta) = \prod_{n=1}^{N} \text{normal\_pdf}\left(y_n \mid \theta, 1\right) \cdot \text{normal\_pdf}\left(\theta \mid 0, 1\right)$$

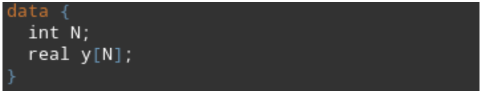
or on the log scale

$$\log \Pr(y, \theta) = \sum_{n=1}^{N} \text{normal\_lpdf}\left(y_n \mid \theta, 1\right) + \text{normal\_lpdf}\left(\theta \mid 0, 1\right)$$

(Bayesian inference with Stan)

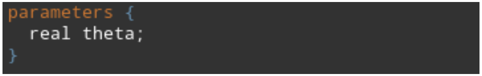
Programming in Stan is structured in blocks:

- the `data` block defines data variables



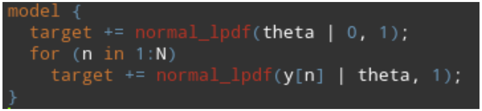
```
data {
  int N;
  real y[N];
}
```

- the `parameters` block defines parameters



```
parameters {
  real theta;
}
```

- the `model` block defines the target log probability density function

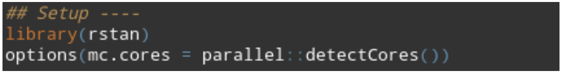


```
model {
  target += normal_lpdf(theta | 0, 1);
  for (n in 1:N)
    target += normal_lpdf(y[n] | theta, 1);
}
```

(Bayesian inference with Stan)

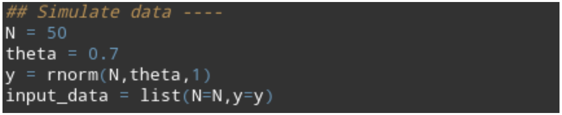
We then explore the target with Hamiltonian Monte Carlo:

- load `rstan` package



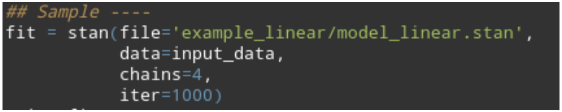
```
## Setup ----
library(rstan)
options(mc.cores = parallel::detectCores())
```

- simulate $N = 50$ data points with $\theta = 0.7$



```
## Simulate data ----
N = 50
theta = 0.7
y = rnorm(N,theta,1)
input_data = list(N=N,y=y)
```

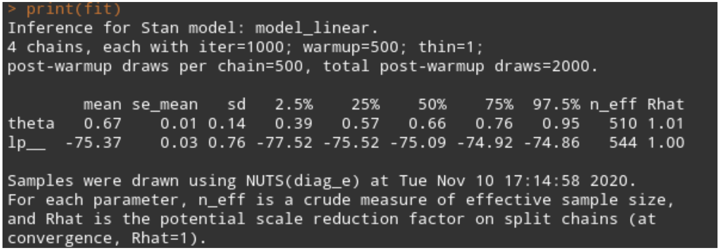
- run sampling



```
## Sample ----
fit = stan(file='example_linear/model_linear.stan',
           data=input_data,
           chains=4,
           iter=1000)
```

(Bayesian inference with Stan)

- print results

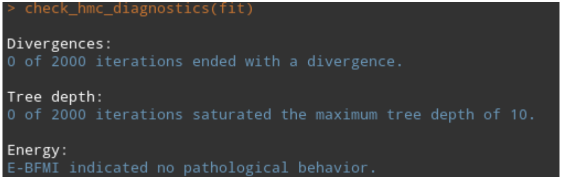


```
> print(fit)
Inference for Stan model: model_linear.
4 chains, each with iter=1000; warmup=500; thin=1;
post-warmup draws per chain=500, total post-warmup draws=2000.

        mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
theta   0.67    0.01 0.14   0.39   0.57   0.66   0.76   0.95   510 1.01
lp__  -75.37    0.03 0.76 -77.52 -75.52 -75.09 -74.92 -74.86   544 1.00

Samples were drawn using NUTS(diag_e) at Tue Nov 10 17:14:58 2020.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

- diagnostics: $\hat{R}$, divergences, tree depth, energy



```
> check_hmc_diagnostics(fit)

Divergences:
0 of 2000 iterations ended with a divergence.

Tree depth:
0 of 2000 iterations saturated the maximum tree depth of 10.

Energy:
E-BFMI indicated no pathological behavior.
```

Acknowledgements & ressources

- Michael Betancourt's *Introduction to Stan*
  `https://betanalpha.github.io/assets/case_studies/stan_intro.html`
- Daniel Lee's *ODEs in Stan*
  `https://youtu.be/hJ34_xJhYeY`
- Richard McElreath's *Statistical rethinking*
  `https://youtu.be/4WVelCswXo4`