

---

# **Compartment Model Helper Documentation**

***Release 0.1***

**SeHyoung Ahn**

**Mar 27, 2020**



**CONTENTS:**

<b>1</b>	<b>Tutorials</b>	<b>3</b>
1.1	SIR Model . . . . .	3
1.2	SEIR Model . . . . .	4
<b>2</b>	<b>Technical Documentation</b>	<b>5</b>
2.1	Methods . . . . .	5
2.2	Properties . . . . .	6
<b>3</b>	<b>License</b>	<b>7</b>
<b>4</b>	<b>Indices and tables</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



Due to the COVID-19, many people would write different iteration of the compartment model. This is written to reduce the repetitive coding of people. For example, the SEIR model can be written as

```
linear.exposed.infectious = 1;
linear.infectious.recovered = 1;
interaction.susceptible.infectious.exposed = 1;
seir_model = epimodel(linear, interaction);

init_dist.infectious = 1e-4;
init_dist.susceptible = 1 - 1e-4;
seir_model.set_initial_dist(init_dist);

seir_model.simulate(10);

plot(seir_model.time_knots, seir_model.results.susceptible);
title('susceptible');
```

One should be able to infer the syntax easily, but see [Tutorials](#) for further explanations of the syntax.



## TUTORIALS

## 1.1 SIR Model

SIR model is given by

$$\begin{aligned}\frac{dS}{dt} &= -\frac{\beta IS}{N}, \\ \frac{dI}{dt} &= \frac{\beta IS}{N} - \gamma I, \\ \frac{dR}{dt} &= \gamma I.\end{aligned}$$

As a compartment model, the flow is defined from one compartment into another at a given rate potentially given with an interaction with another variable. Hence, all the dynamics can be summarized by

$$\text{origin} \rightarrow \text{target} : \text{flowrate} \cdot \text{origin}$$

and

$$\text{origin} \rightarrow \text{target} : \text{flowrate} \cdot \text{interaction} \cdot \text{origin}.$$

These can be more concisely summarized as

$$\begin{aligned}\text{origin.target} &= \text{flowrate} \\ \text{origin.interaction.target} &= \text{flowrate}\end{aligned}$$

Using this syntax with a struct, we can build an `epimodel` by:

```
gamma = 1;
beta_N = 1;

linear.infectious.recovered = gamma;

interaction.susceptible.infectious.infectious = beta_N;

sir_model = epimodel(linear, interaction);
```

The `epimodel` class parses and builds the dynamics matrix under the hood. To simulate the dynamics, we need to define a initial distribution.

```
init_dist.infectious = 1e-4;
init_dist.susceptible = 1 - 1e-4;
sir_model.set_initial_dist(init_dist);
```

The values that are not fed in are automatically filled in as zero. This is sufficient to simulate the model

```
sir_model.simulate(10);
```

where one feed in the length of the simulation (see *simulate* for one other implicit parameter). Simulation populates the *results* struct with the simulation results and also fills in *time\_knots* of time of the variables. Hence, for example,

```
plot(sir_model.time_knots, sir_model.results.susceptible);  
title('susceptible');
```

plots the number of susceptible people.

## 1.2 SEIR Model

SEIR model is similar to SIR model, but with exposed state added. This just requires an addition of one more line. One can summarize these with a struct:

```
E2I = 1;  
I2R = 1;  
beta_N = 1;  
  
linear.exposed.infectious = E2I;  
linear.infectious.recovered = I2R;  
  
interaction.susceptible.infectious.exposed = beta_N;  
  
seir_model = epimodel(linear, interaction);
```

and set the initial distribution

```
init_dist.infectious = 1e-4;  
init_dist.susceptible = 1 - 1e-4;  
seir_model.set_initial_dist(init_dist);
```

Again, “exposed” and “recovered” states are implicitly set to have zero initial distribution.

```
seir_model.simulate(10);
```

simulate for 10 time period.

Finally,

```
plot(seir_model.time_knots, seir_model.results.susceptible);  
title('susceptible');  
  
plot(seir_model.time_knots, seir_model.results.exposed);  
title('exposed');  
  
plot(seir_model.time_knots, seir_model.results.infectious);  
title('infectious');  
  
plot(seir_model.time_knots, seir_model.results.recovered);  
title('recovered');
```

makes plots.



## TECHNICAL DOCUMENTATION

This is the technical documentation. See [Tutorials](#) instead to get started.

### 2.1 Methods

`epimodel.epimodel(struct_linear, struct_interaction)`

Parses struct into matrices

#### Parameters

- **struct\_linear** (*struct*) – struct corresponding to linear flows. The syntax is

```
struct.origin.target = flowrate
```

- **struct\_interaction** (*struct*) – struct corresponding to interacted flows. The syntax is

```
struct.origin.interaction.target = flowrate
```

#### Example

```
% SEIR model
struct_linear.exposed.infectious = 1;
struct_linear.infectious.recovered = 1;
struct_interaction.susceptible.infectious.exposed = 1;
seir_model = epimodel(struct_linear, struct_interaction);
```

`epimodel.set_initial_dist(struct_dist)`

Sets initial distribution

**Parameters** **struct\_dist** (*struct*) – struct of initial distribution. Unfilled values default to zero

#### Example

```
struct_dist.infectious = 1e-4;
struct_dist.susceptible = 1 - 1e-4;
seir_model.set_initial_dist(struct_dist);
```

`epimodel.simulate(end_time, time_step)`

Runs simulation/ODE forward

**Parameters**

- **end\_time** (*double*) – end time of simulation
- **time\_step** (*double*) – [default 1e-3] time step of discretization

**Example**

```
seir_model.simulate(10);
```

**See also:**

- *results*
- *set\_initial\_dist*

---

**Note:** Note that The values are computed using an “explicit” udpate, so *time\_step* needs to small enough to satisfy the CFL condition.

---

## 2.2 Properties

`epimodel.results`

**Type** struct

**Description** Struct of simulation results

**Example**

```
plot(model.results.time, model.results.infectious);
```

`epimodel.init_dist`

**Type** double

**Description** vector of initial distribution

**Note** Set it using *set\_initial\_dist*

`epimodel.name2loc`

**Type** struct

**Description** (INTERNAL ATTRIBUTE) struct of location for different variables

`epimodel.loc2name`

**Type** cell array

**Description** (INTERNAL ATTRIBUTE) cell array of variable names

`epimodel.time_knots`

**Type** double

**Description** time knot points of simulation

## LICENSE

### BSD 2-Clause License

Copyright (c) 2020, SeHyouun Ahn

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.



## INDICES AND TABLES

- `genindex`
- `search`



## INDEX

### E

`epimodel()` (`epimodel.epimodel` method), 5

### I

`init_dist` (in module `epimodel`), 6

### L

`loc2name` (in module `epimodel`), 6

### N

`name2loc` (in module `epimodel`), 6

### R

`results` (in module `epimodel`), 6

### S

`set_initial_dist()` (`epimodel.epimodel` method), 5

`simulate()` (`epimodel.epimodel` method), 5

### T

`time_knots` (in module `epimodel`), 6