DiseaseModelPINN_notebook2

December 16, 2022

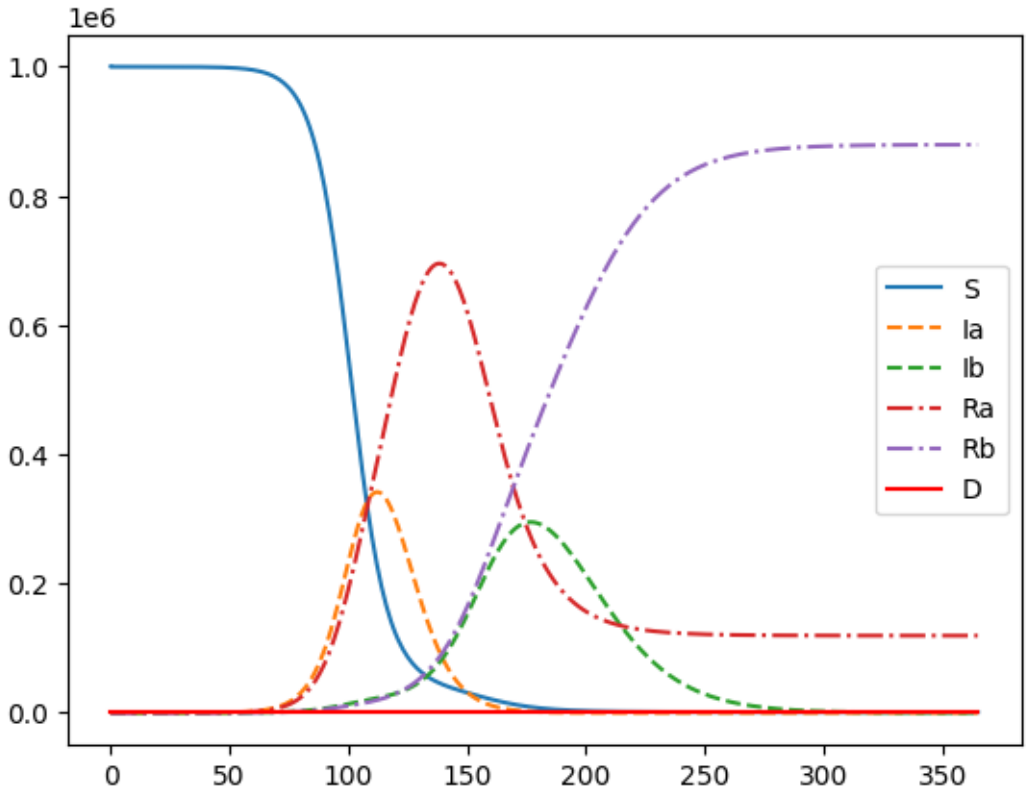
[ ]:
```
import deepxde as dde
from deepxde.backend import pytorch
import torch
import matplotlib.pyplot as plt
import numpy as np

from SIRD_deepxde_DiseaseModel import SIRD_deepxde_net
from DiseaseModel import SIR,SIRD,SIRDIm,SIRDImRel, SIRDImRelSimple, SIRD2Var,␣
↪GeneralModelSolver
from Plot import Plot

# %matplotlib widget

seed = 1
np.random.seed(seed)
dde.config.set_random_seed(seed)
```

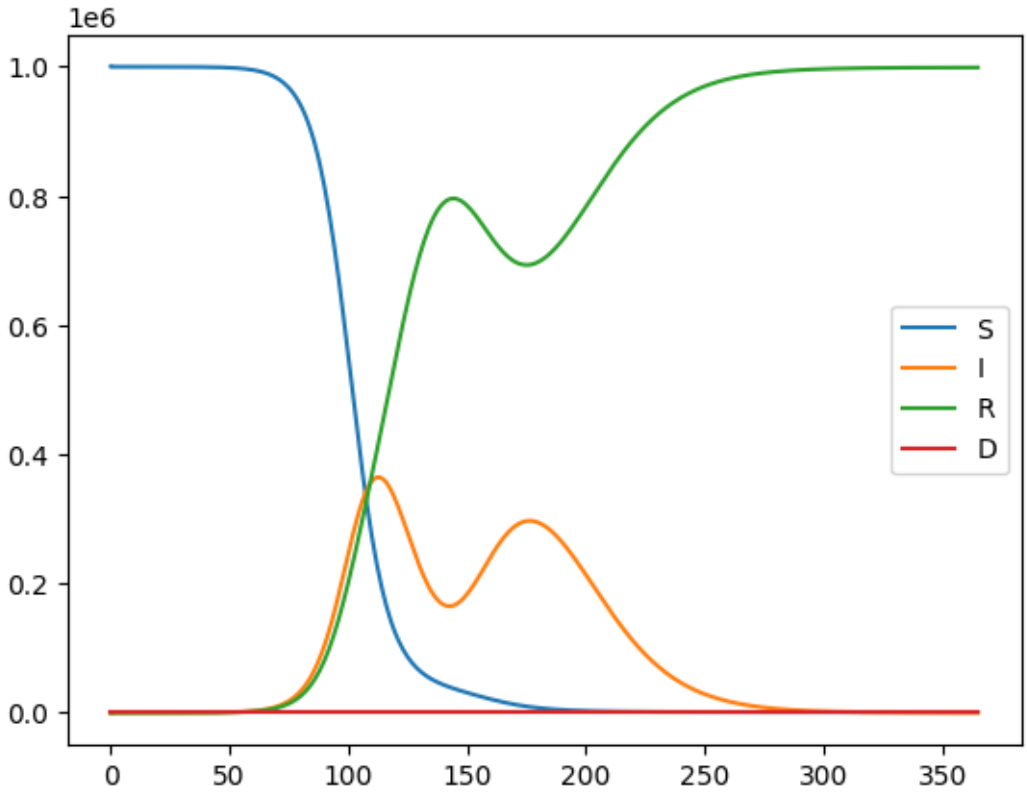
```
Using backend: pytorch


default Torch device: cpu
```

[ ]:
```
time_delta = [0,365] # use three values here for intro time of second variant

# initial_conditions = {
#     "S": 1000000,
#     "I": 1,
#     "R": 0,
#     }
# static_parameters = {
#     "alpha": (0.15),
#     "beta": (0.07),
#     }
# sird_model = SIR(initial_conditions, static_parameters, time_delta)

# initial_conditions = {
#     "S": 1000000,
#     "I": 1,
```

```
#     "R": 0,
#     "D": 0,
#     }
# static_parameters = {
#     "alpha": (0.2),
#     "beta": (0.05),
#     "gamma": (0.001),
#     }
# sird_model = SIRD(initial_conditions, static_parameters, time_delta)

# initial_conditions = {
#     "S": 1000000,
#     "I": 15,
#     "R": 0,
#     "D": 0,
#     "Im": 0, # should be between 0 and 1
#     }
# static_parameters = {
#     "alpha": 0.12,
#     "beta": 0.07,
#     "gamma": 0.02,
#     "kappa": 0.2,
#     }
# sird_model = SIRDIm(initial_conditions, static_parameters, time_delta)
# initial_conditions = {
#     "S": 1000000,
#     "I": 15,
#     "R": 0,
#     "D": 0,
#     "Im": 0, # should be between 0 and 1
#     }
# static_parameters = {
#     "lambda_": 1.5,
#     "gamma": 0.000,
#     "kappa": 0.2,
#     }
# sird_model = SIRDImRel(initial_conditions, static_parameters, time_delta)
# initial_conditions = {
#     "S": 1000000,
#     "I": 15,
#     "R": 0,
#     "Im": 0, # should be between 0 and 1
#     }
# static_parameters = {
#     "lambda_": 1.5,
#     "kappa": 0.2,
#     }
```

```
# sird_model = SIRDImRelSimple(initial_conditions, static_parameters,␣
 ↪time_delta)

initial_conditions = {
    "S": 1000000,
    "Ia": 1,
    "Ib": 0,
    "Ra": 0,
    "Rb": 0,
    "D": 0,
    "Im_a": 0, # should be between 0 and 1
    "Im_b": 0, # should be between 0 and 1
    }
static_parameters = {
    "alpha_a": 0.23  ,
    "alpha_b": 0.18,
    "beta_a": 0.1,
    "beta_b": 0.08,
    "gamma_a": 0.00,
    "gamma_b": 0.00,
    "kappa_a": 0.8,
    "kappa_b": 0.5,
    }
# static_parameters = {
#     "alpha_a": 0.11,
#     "alpha_b": 0.12,
#     "beta_a": 0.08,
#     "beta_b": 0.08,
#     "gamma_a": 0.00,
#     "gamma_b": 0.00,
#     "kappa_a": 0.1,
#     "kappa_b": 0.2,
#     }
sird_model = SIRD2Var(initial_conditions, static_parameters, time_delta)


t_synth, solution_synth_full = sird_model.simulate()
t_synth, solution_synth = sird_model.get_solution_as_sird()
sird_model.plot_solution()
sird_model.plot_sird()
```


1e6
1.0
0.8
0.6
0.4
0.2
0.0
0
50
100
150
200
250
300
350
S
Ia
Ib
Ra
Rb
D



[ ]:
```
print(sird_model)
```

```
A Disease Model with description: 'A model that simulates two concurrent
diseases and natural herd immunity as a factor of the amount of recovered for
each variant':
Parameters:
        alpha_a = 0.23
        alpha_b = 0.18
        beta_a = 0.1
        beta_b = 0.08
        gamma_a = 0.0
        gamma_b = 0.0
        kappa_a = 0.8
        kappa_b = 0.5

PDE groups and initial conditions:
        S = 1000000
        Ia = 1
        Ib = 0
        Ra = 0
        Rb = 0
```

```
        D = 0
        Im_a = 0
        Im_b = 0

PDE equations:
        dS/dt = -(alpha_a/N)*Ia*S  -(alpha_b/N)*Ib*S
        dIa/dt = (alpha_a/N)*S*Ia + (alpha_a/N)*(1 - Im_a)*(Ra + Rb - D)*Ia -
beta_a*Ia - gamma_a*Ia
        dIb/dt = (alpha_b/N)*S*Ib + (alpha_b/N)*(1 - Im_b)*(Ra + Rb - D)*Ib -
beta_b*Ib - gamma_b*Ib
        dRa/dt = beta_a*Ia - (alpha_a/N)*(1 - (Im_a))*(Ra)*(Ia) - (alpha_b/N)*(1
- (Im_b))*(Ra)*(Ib)
        dRb/dt = beta_b*Ib - (alpha_a/N)*(1 - (Im_a))*(Rb)*(Ia) - (alpha_b/N)*(1
- (Im_b))*(Rb)*(Ib)
        dD/dt = gamma_a*Ia + gamma_b*Ib
        dIm_a/dt = kappa_a*beta_a*Ia/N
        dIm_b/dt = kappa_b*beta_b*Ib/N
PINN PDE loss equations:
        dS_t - (-(alpha_a/N)*Ia*S  -(alpha_b/N)*Ib*S)
        dIa_t - ((alpha_a/N)*S*Ia + (alpha_a/N)*(1 - Im_a)*(Ra + Rb - D)*Ia -
beta_a*Ia - gamma_a*Ia)
        dIb_t - ((alpha_b/N)*S*Ib + (alpha_b/N)*(1 - Im_b)*(Ra + Rb - D)*Ib -
beta_b*Ib - gamma_b*Ib)
        dRa_t - (beta_a*Ia - (alpha_a/N)*(1 - (Im_a))*(Ra)*(Ia) - (alpha_b/N)*(1
- (Im_b))*(Ra)*(Ib))
        dRb_t - (beta_b*Ib - (alpha_a/N)*(1 - (Im_a))*(Rb)*(Ia) - (alpha_b/N)*(1
- (Im_b))*(Rb)*(Ib))
        dD_t - (gamma_a*Ia + gamma_b*Ib)
        dIm_a_t - (kappa_a*beta_a*Ia/N)
        dIm_b_t - (kappa_b*beta_b*Ib/N)
```

[ ]:
```
# keep this even if not subsetting
t = t_synth
wsol = solution_synth

solver = GeneralModelSolver(sird_model)
# subset
# max_timestep = 300
# t_bool = t_synth < max_timestep
# t = t_synth[t_bool]
# wsol = wsol_synth[t_bool]
```

[ ]:
```
model = SIRD_deepxde_net(t, wsol,disease_model=sird_model, with_neumann=True,␣
 ↪model_name="diseasemodel_test", with_softadapt=True)
print(model)
hyper_print_every = 100
```

```
model.init_model(lr=0.01, print_every=hyper_print_every, activation="tanh",␣
 ↪nn_layers=2, nn_layer_width=32)
```

```
PINN model:
Parameters: ['alpha_a', 'alpha_b', 'beta_a', 'beta_b', 'gamma_a', 'gamma_b',
'kappa_a', 'kappa_b']
Loss measures: ['dS_t', 'dIa_t', 'dIb_t', 'dRa_t', 'dRb_t', 'dD_t', 'dIm_a_t',
'dIm_b_t', 'ic_Ia', 'ic_Ib', 'ic_Ra', 'ic_Rb', 'ic_D', 'ic_Im_a', 'ic_Im_b',
'ic_S', 'observe_S', 'observe_I', 'observe_R', 'observe_D', 'observe_SUM',
'sign_Ia', 'sign_Ib', 'sign_Ra', 'sign_Rb', 'sign_D', 'sign_Im_a', 'sign_Im_b',
'L1_norm_Ia', 'L1_norm_Ib', 'L1_norm_Ra', 'L1_norm_Rb', 'L1_norm_D',
'L1_norm_Im_a', 'L1_norm_Im_b', 'ic_neumann_S', 'ic_neumann_Ia',
'ic_neumann_Ib', 'ic_neumann_Ra', 'ic_neumann_Rb', 'ic_neumann_D',
'ic_neumann_Im_a', 'ic_neumann_Im_b']
Compiling model…
'compile' took 0.000073 s
```

[ ]:
```
prev_best_step = 0
iters = 0
plot_every=1000
```

[ ]:
```
TOTAL_ITER = 100_000
while True:
    # for n in range(TOTAL_ITER//plot_every):
    model.train_model(iterations=plot_every, print_every=hyper_print_every,␣
 ↪use_LBFGSB=False)
    params_nn, best_step = model.get_best_params(out_func=np.exp) # parameters␣
 ↪need to be extracted with the exponential functino as they have been␣
 ↪modelled in logspace
    if best_step > prev_best_step:
        break
    elif iters >= TOTAL_ITER:
        break

params_nn, best_step = model.get_best_params(out_func=np.exp) # parameters need␣
 ↪to be extracted with the exponential functino as they have been modelled in␣
 ↪logspace
t_nn_param, wsol_nn_param, wsol_sird_nn_param = solver(*params_nn)
# params_nn= tuple(np.exp([*params_nn]))
# print(*params_nn)
model.set_synthetic_data(t_synth, solution_synth_full)
model.set_nn_synthetic_data(t_nn_param, wsol_nn_param, wsol_sird_nn_param)
print(static_parameters, sep="\n")
plot = Plot(model, values_to_plot=sird_model.initial_conditions_keys) # class␣
 ↪that contains plotting functions
plot.show_known_and_prediction()
```

```
plot.plot_param_history()
plot.plot_loss_history()
```

```
Training model…

Step      Train loss
Test loss
Test metric
42000     [6.57e-06, 1.30e-06, 2.21e-06, 2.77e-06, 5.79e-06, 2.20e-07, 3.13e-06,
1.92e-06, 1.19e-06, 4.35e-06, 4.10e-06, 3.03e-05, 9.10e-06, 3.08e-06, 5.19e-06,
3.11e-05, 4.61e-04, 1.40e-03, 6.02e-04, 7.67e-05, 6.35e-05, 3.50e-06, 2.21e-05,
7.68e-05, 5.07e-07, 2.69e-05, 5.42e-08, 1.19e-07, 1.14e-05, 1.44e-06, 2.24e-06,
1.11e-05, 1.62e-05, 1.15e-06, 2.35e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06,
2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06]    [6.57e-06, 1.30e-06, 2.21e-06,
2.77e-06, 5.79e-06, 2.20e-07, 3.13e-06, 1.92e-06, 1.19e-06, 4.35e-06, 4.10e-06,
3.03e-05, 9.10e-06, 3.08e-06, 5.19e-06, 3.11e-05, 4.61e-04, 1.40e-03, 6.02e-04,
7.67e-05, 6.35e-05, 3.50e-06, 2.21e-05, 7.68e-05, 5.07e-07, 2.69e-05, 5.42e-08,
1.19e-07, 1.14e-05, 1.44e-06, 2.24e-06, 1.11e-05, 1.62e-05, 1.15e-06, 2.35e-06,
2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06]
[]
42000     [6.57e-06, 1.30e-06, 2.21e-06, 2.77e-06, 5.79e-06, 2.20e-07, 3.13e-06,
1.92e-06, 1.19e-06, 4.35e-06, 4.10e-06, 3.03e-05, 9.10e-06, 3.08e-06, 5.19e-06,
3.11e-05, 4.61e-04, 1.40e-03, 6.02e-04, 7.67e-05, 6.35e-05, 3.50e-06, 2.21e-05,
7.68e-05, 5.07e-07, 2.69e-05, 5.42e-08, 1.19e-07, 1.14e-05, 1.44e-06, 2.24e-06,
1.11e-05, 1.62e-05, 1.15e-06, 2.35e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06,
2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06]    [6.57e-06, 1.30e-06, 2.21e-06,
2.77e-06, 5.79e-06, 2.20e-07, 3.13e-06, 1.92e-06, 1.19e-06, 4.35e-06, 4.10e-06,
3.03e-05, 9.10e-06, 3.08e-06, 5.19e-06, 3.11e-05, 4.61e-04, 1.40e-03, 6.02e-04,
7.67e-05, 6.35e-05, 3.50e-06, 2.21e-05, 7.68e-05, 5.07e-07, 2.69e-05, 5.42e-08,
1.19e-07, 1.14e-05, 1.44e-06, 2.24e-06, 1.11e-05, 1.62e-05, 1.15e-06, 2.35e-06,
2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06, 2.21e-06]
[]
42100     [6.41e-06, 1.45e-06, 2.10e-06, 2.67e-06, 5.62e-06, 2.21e-07, 3.21e-06,
1.88e-06, 1.84e-06, 9.79e-06, 2.29e-06, 2.68e-05, 5.40e-06, 5.04e-07, 2.60e-06,
6.76e-07, 4.56e-04, 1.30e-03, 5.71e-04, 6.57e-05, 8.61e-05, 3.91e-06, 2.03e-05,
7.52e-05, 6.94e-07, 2.35e-05, 1.52e-08, 6.14e-08, 1.30e-05, 3.15e-06, 4.84e-06,
1.38e-05, 1.46e-05, 3.93e-07, 1.26e-06, 2.16e-07, 2.16e-07, 2.16e-07, 2.16e-07,
2.16e-07, 2.16e-07, 2.16e-07, 2.16e-07]    [6.41e-06, 1.45e-06, 2.10e-06,
2.67e-06, 5.62e-06, 2.21e-07, 3.21e-06, 1.88e-06, 1.84e-06, 9.79e-06, 2.29e-06,
2.68e-05, 5.40e-06, 5.04e-07, 2.60e-06, 6.76e-07, 4.56e-04, 1.30e-03, 5.71e-04,
6.57e-05, 8.61e-05, 3.91e-06, 2.03e-05, 7.52e-05, 6.94e-07, 2.35e-05, 1.52e-08,
6.14e-08, 1.30e-05, 3.15e-06, 4.84e-06, 1.38e-05, 1.46e-05, 3.93e-07, 1.26e-06,
2.16e-07, 2.16e-07, 2.16e-07, 2.16e-07, 2.16e-07, 2.16e-07, 2.16e-07, 2.16e-07]
[]
42200     [6.22e-06, 1.64e-06, 2.00e-06, 2.61e-06, 5.39e-06, 2.17e-07, 3.25e-06,
1.86e-06, 1.64e-07, 4.75e-06, 6.68e-11, 3.84e-05, 6.76e-06, 5.38e-07, 2.64e-06,
1.43e-06, 4.80e-04, 1.20e-03, 5.44e-04, 5.72e-05, 1.05e-04, 5.11e-06, 1.64e-05,
```

7.33e-05, 1.10e-06, 1.95e-05, 1.35e-08, 5.69e-08, 1.50e-05, 3.53e-06, 7.14e-06,
1.99e-05, 1.55e-05, 3.95e-07, 1.10e-06, 5.44e-07, 5.44e-07, 5.44e-07, 5.44e-07,
5.44e-07, 5.44e-07, 5.44e-07, 5.44e-07] [6.22e-06, 1.64e-06, 2.00e-06,
2.61e-06, 5.39e-06, 2.17e-07, 3.25e-06, 1.86e-06, 1.64e-07, 4.75e-06, 6.68e-11,
3.84e-05, 6.76e-06, 5.38e-07, 2.64e-06, 1.43e-06, 4.80e-04, 1.20e-03, 5.44e-04,
5.72e-05, 1.05e-04, 5.11e-06, 1.64e-05, 7.33e-05, 1.10e-06, 1.95e-05, 1.35e-08,
5.69e-08, 1.50e-05, 3.53e-06, 7.14e-06, 1.99e-05, 1.55e-05, 3.95e-07, 1.10e-06,
5.44e-07, 5.44e-07, 5.44e-07, 5.44e-07, 5.44e-07, 5.44e-07, 5.44e-07, 5.44e-07]
[]
42300 [5.90e-06, 1.83e-06, 1.93e-06, 2.62e-06, 5.16e-06, 2.07e-07, 3.26e-06,
1.84e-06, 5.20e-06, 8.16e-06, 7.89e-06, 1.06e-05, 4.15e-06, 6.47e-09, 1.53e-06,
2.32e-06, 4.32e-04, 1.13e-03, 5.38e-04, 4.96e-05, 1.16e-04, 4.85e-06, 1.47e-05,
8.25e-05, 2.33e-07, 1.59e-05, 4.60e-09, 3.86e-08, 1.88e-05, 5.76e-06, 5.04e-06,
4.39e-06, 1.36e-05, 2.86e-07, 6.77e-07, 1.89e-09, 1.89e-09, 1.89e-09, 1.89e-09,
1.89e-09, 1.89e-09, 1.89e-09, 1.89e-09] [5.90e-06, 1.83e-06, 1.93e-06,
2.62e-06, 5.16e-06, 2.07e-07, 3.26e-06, 1.84e-06, 5.20e-06, 8.16e-06, 7.89e-06,
1.06e-05, 4.15e-06, 6.47e-09, 1.53e-06, 2.32e-06, 4.32e-04, 1.13e-03, 5.38e-04,
4.96e-05, 1.16e-04, 4.85e-06, 1.47e-05, 8.25e-05, 2.33e-07, 1.59e-05, 4.60e-09,
3.86e-08, 1.88e-05, 5.76e-06, 5.04e-06, 4.39e-06, 1.36e-05, 2.86e-07, 6.77e-07,
1.89e-09, 1.89e-09, 1.89e-09, 1.89e-09, 1.89e-09, 1.89e-09, 1.89e-09, 1.89e-09]
[]
42400 [5.50e-06, 1.94e-06, 1.68e-06, 2.44e-06, 5.02e-06, 1.87e-07, 3.29e-06,
1.85e-06, 1.05e-07, 2.49e-07, 1.64e-05, 5.77e-05, 8.33e-06, 1.22e-06, 3.03e-06,
6.27e-06, 2.77e-04, 1.13e-03, 6.12e-04, 4.17e-05, 1.18e-04, 4.19e-06, 1.15e-05,
5.20e-05, 1.90e-06, 1.26e-05, 3.70e-08, 6.95e-08, 2.42e-05, 7.27e-06, 4.85e-05,
3.10e-05, 1.80e-05, 8.02e-07, 1.14e-06, 8.60e-09, 8.60e-09, 8.60e-09, 8.60e-09,
8.60e-09, 8.60e-09, 8.60e-09, 8.60e-09] [5.50e-06, 1.94e-06, 1.68e-06,
2.44e-06, 5.02e-06, 1.87e-07, 3.29e-06, 1.85e-06, 1.05e-07, 2.49e-07, 1.64e-05,
5.77e-05, 8.33e-06, 1.22e-06, 3.03e-06, 6.27e-06, 2.77e-04, 1.13e-03, 6.12e-04,
4.17e-05, 1.18e-04, 4.19e-06, 1.15e-05, 5.20e-05, 1.90e-06, 1.26e-05, 3.70e-08,
6.95e-08, 2.42e-05, 7.27e-06, 4.85e-05, 3.10e-05, 1.80e-05, 8.02e-07, 1.14e-06,
8.60e-09, 8.60e-09, 8.60e-09, 8.60e-09, 8.60e-09, 8.60e-09, 8.60e-09, 8.60e-09]
[]
42500 [2.07e-05, 2.51e-06, 2.72e-06, 2.55e-06, 1.41e-05, 1.65e-07, 4.06e-06,
2.51e-06, 2.52e-04, 2.88e-04, 8.49e-05, 5.97e-04, 6.40e-05, 6.21e-05, 9.80e-06,
9.08e-04, 6.51e-04, 1.65e-03, 7.74e-04, 7.83e-05, 2.05e-05, 1.90e-06, 1.00e-04,
7.17e-05, 8.28e-06, 8.16e-06, 0.00e+00, 6.10e-08, 3.11e-04, 4.90e-05, 7.83e-05,
1.25e-04, 1.70e-04, 3.57e-04, 2.82e-05, 2.01e-07, 2.01e-07, 2.01e-07, 2.01e-07,
2.01e-07, 2.01e-07, 2.01e-07, 2.01e-07] [2.07e-05, 2.51e-06, 2.72e-06,
2.55e-06, 1.41e-05, 1.65e-07, 4.06e-06, 2.51e-06, 2.52e-04, 2.88e-04, 8.49e-05,
5.97e-04, 6.40e-05, 6.21e-05, 9.80e-06, 9.08e-04, 6.51e-04, 1.65e-03, 7.74e-04,
7.83e-05, 2.05e-05, 1.90e-06, 1.00e-04, 7.17e-05, 8.28e-06, 8.16e-06, 0.00e+00,
6.10e-08, 3.11e-04, 4.90e-05, 7.83e-05, 1.25e-04, 1.70e-04, 3.57e-04, 2.82e-05,
2.01e-07, 2.01e-07, 2.01e-07, 2.01e-07, 2.01e-07, 2.01e-07, 2.01e-07, 2.01e-07]
[]
42600 [7.54e-06, 1.49e-06, 2.57e-06, 2.28e-06, 6.81e-06, 1.21e-07, 3.53e-06,
2.06e-06, 7.96e-06, 8.41e-07, 7.91e-06, 9.83e-06, 2.24e-06, 2.25e-06, 1.35e-06,
8.59e-06, 5.97e-04, 1.29e-03, 5.24e-04, 3.37e-05, 4.45e-05, 1.42e-05, 1.72e-05,

```
1.20e-04, 3.69e-06, 1.51e-05, 1.09e-06, 3.14e-07, 1.24e-05, 2.98e-06, 5.83e-06,
9.66e-06, 2.61e-06, 1.83e-06, 1.19e-06, 8.46e-06, 8.46e-06, 8.46e-06, 8.46e-06,
8.46e-06, 8.46e-06, 8.46e-06, 8.46e-06]    [7.54e-06, 1.49e-06, 2.57e-06,
2.28e-06, 6.81e-06, 1.21e-07, 3.53e-06, 2.06e-06, 7.96e-06, 8.41e-07, 7.91e-06,
9.83e-06, 2.24e-06, 2.25e-06, 1.35e-06, 8.59e-06, 5.97e-04, 1.29e-03, 5.24e-04,
3.37e-05, 4.45e-05, 1.42e-05, 1.72e-05, 1.20e-04, 3.69e-06, 1.51e-05, 1.09e-06,
3.14e-07, 1.24e-05, 2.98e-06, 5.83e-06, 9.66e-06, 2.61e-06, 1.83e-06, 1.19e-06,
8.46e-06, 8.46e-06, 8.46e-06, 8.46e-06, 8.46e-06, 8.46e-06, 8.46e-06, 8.46e-06]
[]
42700     [6.49e-06, 1.53e-06, 1.71e-06, 2.33e-06, 5.98e-06, 1.09e-07, 3.46e-06,
2.01e-06, 7.90e-06, 2.52e-06, 1.02e-05, 5.75e-05, 5.32e-06, 8.17e-07, 2.06e-06,
6.60e-07, 5.65e-04, 1.19e-03, 5.03e-04, 3.10e-05, 5.37e-05, 1.42e-05, 1.44e-05,
7.11e-05, 1.04e-05, 1.30e-05, 6.64e-07, 3.75e-07, 2.76e-06, 1.30e-06, 1.97e-05,
1.85e-05, 7.45e-06, 1.31e-06, 6.04e-07, 2.57e-07, 2.57e-07, 2.57e-07, 2.57e-07,
2.57e-07, 2.57e-07, 2.57e-07, 2.57e-07]    [6.49e-06, 1.53e-06, 1.71e-06,
2.33e-06, 5.98e-06, 1.09e-07, 3.46e-06, 2.01e-06, 7.90e-06, 2.52e-06, 1.02e-05,
5.75e-05, 5.32e-06, 8.17e-07, 2.06e-06, 6.60e-07, 5.65e-04, 1.19e-03, 5.03e-04,
3.10e-05, 5.37e-05, 1.42e-05, 1.44e-05, 7.11e-05, 1.04e-05, 1.30e-05, 6.64e-07,
3.75e-07, 2.76e-06, 1.30e-06, 1.97e-05, 1.85e-05, 7.45e-06, 1.31e-06, 6.04e-07,
2.57e-07, 2.57e-07, 2.57e-07, 2.57e-07, 2.57e-07, 2.57e-07, 2.57e-07, 2.57e-07]
[]
42800     [6.13e-06, 1.68e-06, 1.64e-06, 2.32e-06, 5.59e-06, 1.01e-07, 3.38e-06,
1.99e-06, 8.95e-06, 1.34e-06, 2.28e-05, 1.10e-04, 4.11e-06, 1.75e-06, 1.76e-06,
2.83e-07, 5.10e-04, 1.10e-03, 4.94e-04, 2.76e-05, 6.37e-05, 1.24e-05, 1.39e-05,
5.13e-05, 1.64e-05, 1.04e-05, 6.84e-07, 2.62e-07, 2.68e-06, 5.81e-07, 3.65e-05,
5.33e-05, 5.97e-06, 1.02e-06, 6.48e-07, 1.85e-08, 1.85e-08, 1.85e-08, 1.85e-08,
1.85e-08, 1.85e-08, 1.85e-08, 1.85e-08]    [6.13e-06, 1.68e-06, 1.64e-06,
2.32e-06, 5.59e-06, 1.01e-07, 3.38e-06, 1.99e-06, 8.95e-06, 1.34e-06, 2.28e-05,
1.10e-04, 4.11e-06, 1.75e-06, 1.76e-06, 2.83e-07, 5.10e-04, 1.10e-03, 4.94e-04,
2.76e-05, 6.37e-05, 1.24e-05, 1.39e-05, 5.13e-05, 1.64e-05, 1.04e-05, 6.84e-07,
2.62e-07, 2.68e-06, 5.81e-07, 3.65e-05, 5.33e-05, 5.97e-06, 1.02e-06, 6.48e-07,
1.85e-08, 1.85e-08, 1.85e-08, 1.85e-08, 1.85e-08, 1.85e-08, 1.85e-08, 1.85e-08]
[]
42900     [5.47e-06, 1.86e-06, 1.57e-06, 2.36e-06, 5.15e-06, 9.34e-08, 3.25e-06,
1.97e-06, 4.67e-06, 2.46e-06, 3.21e-06, 6.62e-05, 3.96e-06, 2.92e-07, 1.24e-06,
4.19e-07, 4.55e-04, 1.02e-03, 4.86e-04, 2.39e-05, 6.80e-05, 1.19e-05, 9.89e-06,
5.31e-05, 8.15e-06, 9.01e-06, 4.01e-08, 1.23e-07, 1.29e-06, 9.71e-07, 9.30e-06,
2.41e-05, 5.66e-06, 3.74e-06, 7.18e-07, 9.34e-07, 9.34e-07, 9.34e-07, 9.34e-07,
9.34e-07, 9.34e-07, 9.34e-07, 9.34e-07]    [5.47e-06, 1.86e-06, 1.57e-06,
2.36e-06, 5.15e-06, 9.34e-08, 3.25e-06, 1.97e-06, 4.67e-06, 2.46e-06, 3.21e-06,
6.62e-05, 3.96e-06, 2.92e-07, 1.24e-06, 4.19e-07, 4.55e-04, 1.02e-03, 4.86e-04,
2.39e-05, 6.80e-05, 1.19e-05, 9.89e-06, 5.31e-05, 8.15e-06, 9.01e-06, 4.01e-08,
1.23e-07, 1.29e-06, 9.71e-07, 9.30e-06, 2.41e-05, 5.66e-06, 3.74e-06, 7.18e-07,
9.34e-07, 9.34e-07, 9.34e-07, 9.34e-07, 9.34e-07, 9.34e-07, 9.34e-07, 9.34e-07]
[]
43000     [3.95e-06, 2.11e-06, 1.41e-06, 2.49e-06, 4.37e-06, 9.22e-08, 3.01e-06,
1.92e-06, 1.96e-06, 9.44e-06, 1.44e-05, 2.60e-06, 4.43e-07, 1.40e-06, 7.51e-08,
3.11e-05, 3.62e-04, 8.90e-04, 4.81e-04, 2.10e-05, 7.13e-05, 1.22e-06, 4.93e-06,
```

6.83e-05, 3.54e-08, 6.29e-06, 1.89e-08, 0.00e+00, 2.10e-05, 2.37e-05, 3.36e-06,
3.78e-06, 1.67e-06, 5.09e-07, 1.58e-06, 2.09e-06, 2.09e-06, 2.09e-06, 2.09e-06,
2.09e-06, 2.09e-06, 2.09e-06, 2.09e-06]   [3.95e-06, 2.11e-06, 1.41e-06,
2.49e-06, 4.37e-06, 9.22e-08, 3.01e-06, 1.92e-06, 1.96e-06, 9.44e-06, 1.44e-05,
2.60e-06, 4.43e-07, 1.40e-06, 7.51e-08, 3.11e-05, 3.62e-04, 8.90e-04, 4.81e-04,
2.10e-05, 7.13e-05, 1.22e-06, 4.93e-06, 6.83e-05, 3.54e-08, 6.29e-06, 1.89e-08,
0.00e+00, 2.10e-05, 2.37e-05, 3.36e-06, 3.78e-06, 1.67e-06, 5.09e-07, 1.58e-06,
2.09e-06, 2.09e-06, 2.09e-06, 2.09e-06, 2.09e-06, 2.09e-06, 2.09e-06, 2.09e-06]
[]
43100   [3.30e-06, 2.42e-06, 1.31e-06, 2.82e-06, 3.78e-06, 7.96e-08, 2.69e-06,
1.85e-06, 4.69e-06, 7.68e-07, 2.08e-05, 3.96e-06, 1.11e-06, 2.48e-07, 9.57e-08,
2.23e-06, 2.95e-04, 8.26e-04, 4.66e-04, 1.66e-05, 7.80e-05, 3.59e-06, 1.29e-06,
4.05e-05, 4.01e-08, 5.25e-06, 3.16e-09, 0.00e+00, 1.14e-05, 5.36e-06, 3.92e-06,
1.53e-06, 3.38e-06, 3.70e-07, 1.48e-06, 1.50e-08, 1.50e-08, 1.50e-08, 1.50e-08,
1.50e-08, 1.50e-08, 1.50e-08, 1.50e-08]   [3.30e-06, 2.42e-06, 1.31e-06,
2.82e-06, 3.78e-06, 7.96e-08, 2.69e-06, 1.85e-06, 4.69e-06, 7.68e-07, 2.08e-05,
3.96e-06, 1.11e-06, 2.48e-07, 9.57e-08, 2.23e-06, 2.95e-04, 8.26e-04, 4.66e-04,
1.66e-05, 7.80e-05, 3.59e-06, 1.29e-06, 4.05e-05, 4.01e-08, 5.25e-06, 3.16e-09,
0.00e+00, 1.14e-05, 5.36e-06, 3.92e-06, 1.53e-06, 3.38e-06, 3.70e-07, 1.48e-06,
1.50e-08, 1.50e-08, 1.50e-08, 1.50e-08, 1.50e-08, 1.50e-08, 1.50e-08, 1.50e-08]
[]
43200   [3.22e-06, 2.60e-06, 1.31e-06, 2.88e-06, 3.61e-06, 6.77e-08, 2.50e-06,
1.82e-06, 7.83e-07, 5.57e-08, 5.44e-06, 1.06e-04, 2.55e-06, 7.04e-07, 2.09e-07,
1.51e-05, 2.70e-04, 7.87e-04, 4.61e-04, 1.32e-05, 7.71e-05, 4.25e-06, 1.33e-06,
1.49e-05, 3.50e-06, 4.46e-06, 5.79e-09, 1.61e-09, 9.94e-06, 4.15e-06, 1.89e-05,
5.25e-05, 4.49e-06, 7.50e-07, 9.87e-07, 1.21e-06, 1.21e-06, 1.21e-06, 1.21e-06,
1.21e-06, 1.21e-06, 1.21e-06, 1.21e-06]   [3.22e-06, 2.60e-06, 1.31e-06,
2.88e-06, 3.61e-06, 6.77e-08, 2.50e-06, 1.82e-06, 7.83e-07, 5.57e-08, 5.44e-06,
1.06e-04, 2.55e-06, 7.04e-07, 2.09e-07, 1.51e-05, 2.70e-04, 7.87e-04, 4.61e-04,
1.32e-05, 7.71e-05, 4.25e-06, 1.33e-06, 1.49e-05, 3.50e-06, 4.46e-06, 5.79e-09,
1.61e-09, 9.94e-06, 4.15e-06, 1.89e-05, 5.25e-05, 4.49e-06, 7.50e-07, 9.87e-07,
1.21e-06, 1.21e-06, 1.21e-06, 1.21e-06, 1.21e-06, 1.21e-06, 1.21e-06, 1.21e-06]
[]
43300   [3.08e-06, 2.65e-06, 1.42e-06, 2.92e-06, 3.48e-06, 5.93e-08, 2.42e-06,
1.82e-06, 2.86e-05, 1.60e-05, 9.93e-05, 8.58e-08, 4.41e-07, 1.11e-06, 4.48e-07,
4.90e-06, 2.39e-04, 7.84e-04, 4.71e-04, 1.03e-05, 8.20e-05, 1.00e-06, 1.12e-07,
5.32e-05, 1.68e-10, 3.05e-06, 1.28e-08, 4.21e-09, 1.29e-05, 5.47e-06, 5.47e-05,
7.31e-06, 1.70e-06, 4.42e-07, 5.23e-07, 9.84e-07, 9.84e-07, 9.84e-07, 9.84e-07,
9.84e-07, 9.84e-07, 9.84e-07, 9.84e-07]   [3.08e-06, 2.65e-06, 1.42e-06,
2.92e-06, 3.48e-06, 5.93e-08, 2.42e-06, 1.82e-06, 2.86e-05, 1.60e-05, 9.93e-05,
8.58e-08, 4.41e-07, 1.11e-06, 4.48e-07, 4.90e-06, 2.39e-04, 7.84e-04, 4.71e-04,
1.03e-05, 8.20e-05, 1.00e-06, 1.12e-07, 5.32e-05, 1.68e-10, 3.05e-06, 1.28e-08,
4.21e-09, 1.29e-05, 5.47e-06, 5.47e-05, 7.31e-06, 1.70e-06, 4.42e-07, 5.23e-07,
9.84e-07, 9.84e-07, 9.84e-07, 9.84e-07, 9.84e-07, 9.84e-07, 9.84e-07, 9.84e-07]
[]
43400   [3.05e-03, 2.45e-05, 2.11e-03, 3.19e-04, 3.32e-04, 2.47e-07, 1.80e-05,
1.30e-05, 4.93e-02, 5.17e-02, 5.20e-02, 5.14e-02, 2.15e-02, 2.28e-05, 6.16e-05,
1.89e-02, 4.88e-02, 2.01e-01, 1.57e-01, 2.37e-02, 1.43e+00, 0.00e+00, 0.00e+00,

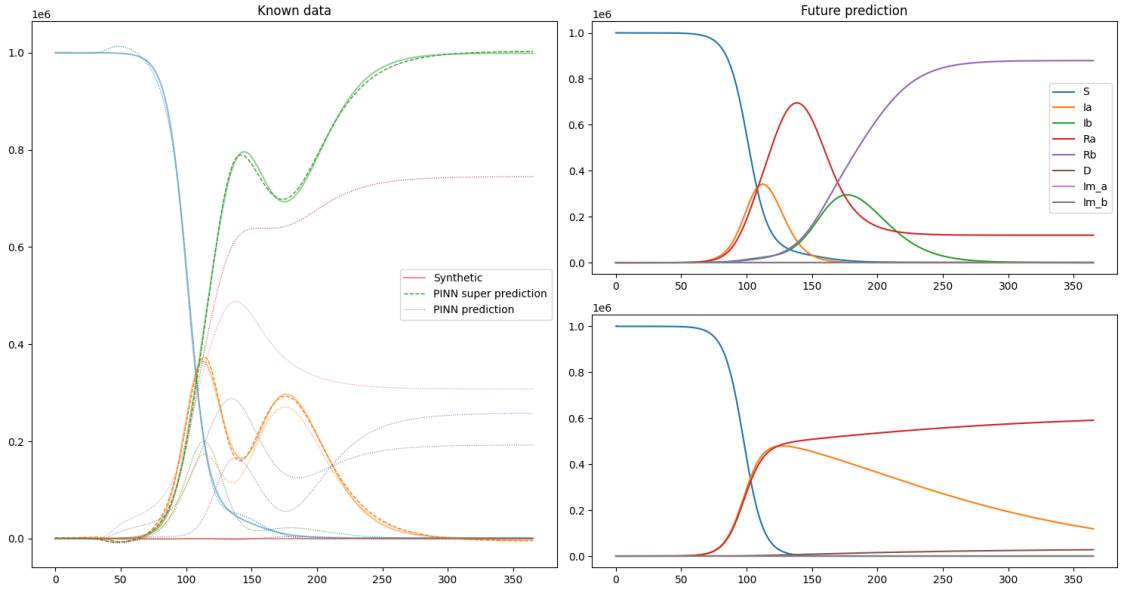
0.00e+00, 0.00e+00, 0.00e+00, 1.52e-05, 2.23e-05, 4.50e-02, 4.87e-02, 5.37e-02,
4.97e-02, 2.12e-02, 4.58e-05, 8.17e-05, 1.07e-04, 1.07e-04, 1.07e-04, 1.07e-04,
1.07e-04, 1.07e-04, 1.07e-04, 1.07e-04] [3.05e-03, 2.45e-05, 2.11e-03,
3.19e-04, 3.32e-04, 2.47e-07, 1.80e-05, 1.30e-05, 4.93e-02, 5.17e-02, 5.20e-02,
5.14e-02, 2.15e-02, 2.28e-05, 6.16e-05, 1.89e-02, 4.88e-02, 2.01e-01, 1.57e-01,
2.37e-02, 1.43e+00, 0.00e+00, 0.00e+00, 0.00e+00, 0.00e+00, 0.00e+00, 1.52e-05,
2.23e-05, 4.50e-02, 4.87e-02, 5.37e-02, 4.97e-02, 2.12e-02, 4.58e-05, 8.17e-05,
1.07e-04, 1.07e-04, 1.07e-04, 1.07e-04, 1.07e-04, 1.07e-04, 1.07e-04, 1.07e-04]
[]
43500 [3.44e-05, 5.67e-06, 4.46e-06, 3.78e-06, 9.63e-06, 1.23e-08, 6.18e-06,
4.46e-06, 8.46e-06, 8.23e-06, 2.71e-05, 9.05e-06, 3.19e-06, 5.52e-07, 3.46e-07,
1.48e-05, 5.17e-04, 6.10e-03, 6.24e-03, 1.69e-06, 2.68e-05, 0.00e+00, 0.00e+00,
1.29e-04, 0.00e+00, 1.67e-06, 0.00e+00, 0.00e+00, 6.60e-06, 6.95e-06, 2.87e-05,
8.47e-06, 3.27e-06, 4.51e-07, 2.98e-07, 9.38e-07, 9.38e-07, 9.38e-07, 9.38e-07,
9.38e-07, 9.38e-07, 9.38e-07, 9.38e-07] [3.44e-05, 5.67e-06, 4.46e-06,
3.78e-06, 9.63e-06, 1.23e-08, 6.18e-06, 4.46e-06, 8.46e-06, 8.23e-06, 2.71e-05,
9.05e-06, 3.19e-06, 5.52e-07, 3.46e-07, 1.48e-05, 5.17e-04, 6.10e-03, 6.24e-03,
1.69e-06, 2.68e-05, 0.00e+00, 0.00e+00, 1.29e-04, 0.00e+00, 1.67e-06, 0.00e+00,
0.00e+00, 6.60e-06, 6.95e-06, 2.87e-05, 8.47e-06, 3.27e-06, 4.51e-07, 2.98e-07,
9.38e-07, 9.38e-07, 9.38e-07, 9.38e-07, 9.38e-07, 9.38e-07, 9.38e-07, 9.38e-07]
[]
43600 [1.28e-05, 2.26e-06, 2.42e-06, 1.68e-06, 7.88e-06, 4.38e-08, 2.94e-06,
2.31e-06, 3.42e-05, 2.03e-04, 2.07e-06, 1.51e-06, 3.34e-04, 4.18e-06, 2.55e-05,
5.33e-04, 1.72e-04, 1.07e-03, 6.59e-04, 2.09e-04, 1.49e-04, 3.79e-07, 2.11e-05,
4.85e-05, 8.64e-09, 0.00e+00, 3.04e-08, 7.70e-07, 2.91e-05, 7.33e-05, 1.42e-05,
4.02e-05, 3.49e-04, 5.31e-06, 7.04e-06, 1.99e-06, 1.99e-06, 1.99e-06, 1.99e-06,
1.99e-06, 1.99e-06, 1.99e-06, 1.99e-06] [1.28e-05, 2.26e-06, 2.42e-06,
1.68e-06, 7.88e-06, 4.38e-08, 2.94e-06, 2.31e-06, 3.42e-05, 2.03e-04, 2.07e-06,
1.51e-06, 3.34e-04, 4.18e-06, 2.55e-05, 5.33e-04, 1.72e-04, 1.07e-03, 6.59e-04,
2.09e-04, 1.49e-04, 3.79e-07, 2.11e-05, 4.85e-05, 8.64e-09, 0.00e+00, 3.04e-08,
7.70e-07, 2.91e-05, 7.33e-05, 1.42e-05, 4.02e-05, 3.49e-04, 5.31e-06, 7.04e-06,
1.99e-06, 1.99e-06, 1.99e-06, 1.99e-06, 1.99e-06, 1.99e-06, 1.99e-06, 1.99e-06]
[]
43700 [5.30e-06, 2.21e-06, 2.01e-06, 1.91e-06, 4.79e-06, 3.42e-08, 2.52e-06,
1.89e-06, 2.62e-06, 2.49e-06, 1.15e-05, 1.33e-05, 2.04e-06, 3.34e-07, 2.29e-07,
9.77e-06, 2.87e-04, 7.96e-04, 4.59e-04, 4.96e-06, 2.76e-05, 4.23e-06, 1.17e-05,
3.03e-05, 3.96e-07, 5.55e-07, 3.88e-09, 1.84e-09, 5.25e-06, 7.91e-07, 8.77e-06,
6.79e-06, 1.96e-06, 2.63e-06, 1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06,
1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06] [5.30e-06, 2.21e-06, 2.01e-06,
1.91e-06, 4.79e-06, 3.42e-08, 2.52e-06, 1.89e-06, 2.62e-06, 2.49e-06, 1.15e-05,
1.33e-05, 2.04e-06, 3.34e-07, 2.29e-07, 9.77e-06, 2.87e-04, 7.96e-04, 4.59e-04,
4.96e-06, 2.76e-05, 4.23e-06, 1.17e-05, 3.03e-05, 3.96e-07, 5.55e-07, 3.88e-09,
1.84e-09, 5.25e-06, 7.91e-07, 8.77e-06, 6.79e-06, 1.96e-06, 2.63e-06, 1.75e-06,
1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06, 1.75e-06]
[]
43800 [3.04e-06, 2.45e-06, 1.27e-06, 2.18e-06, 3.90e-06, 3.41e-08, 2.38e-06,
1.82e-06, 2.33e-06, 2.92e-08, 1.53e-05, 8.88e-06, 7.34e-07, 3.48e-08, 7.71e-08,
1.46e-06, 2.15e-04, 7.28e-04, 4.48e-04, 4.30e-06, 3.72e-05, 8.30e-07, 4.62e-06,

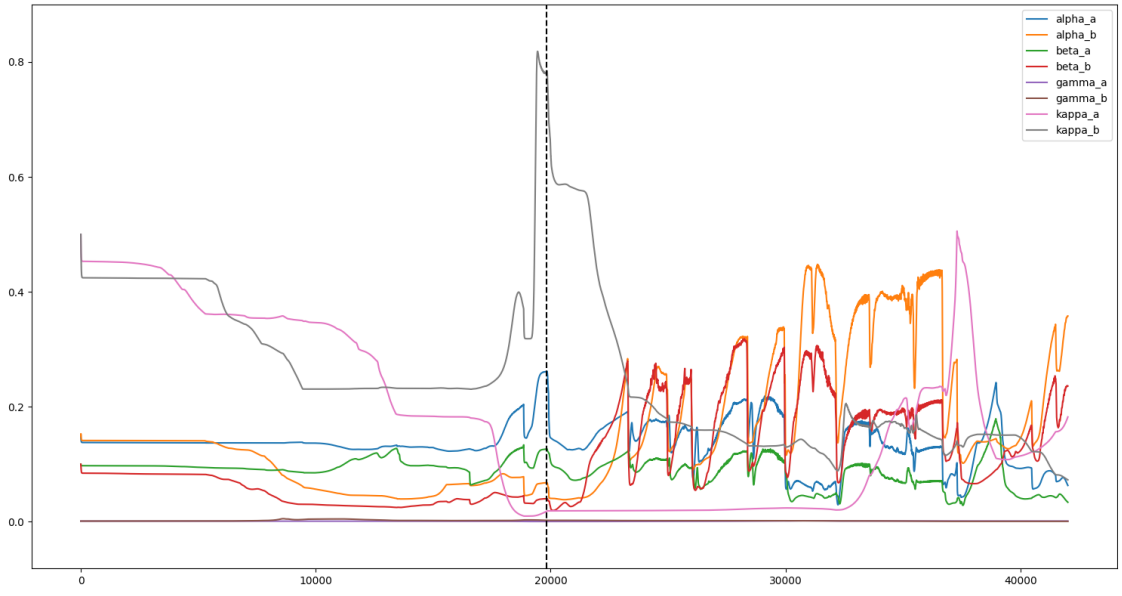
```
2.24e-05, 8.95e-08, 1.21e-06, 5.48e-10, 0.00e+00, 6.36e-06, 5.33e-06, 4.87e-06,
1.23e-06, 3.43e-07, 9.01e-07, 1.64e-06, 1.28e-06, 1.28e-06, 1.28e-06, 1.28e-06,
1.28e-06, 1.28e-06, 1.28e-06, 1.28e-06]    [3.04e-06, 2.45e-06, 1.27e-06,
2.18e-06, 3.90e-06, 3.41e-08, 2.38e-06, 1.82e-06, 2.33e-06, 2.92e-08, 1.53e-05,
8.88e-06, 7.34e-07, 3.48e-08, 7.71e-08, 1.46e-06, 2.15e-04, 7.28e-04, 4.48e-04,
4.30e-06, 3.72e-05, 8.30e-07, 4.62e-06, 2.24e-05, 8.95e-08, 1.21e-06, 5.48e-10,
0.00e+00, 6.36e-06, 5.33e-06, 4.87e-06, 1.23e-06, 3.43e-07, 9.01e-07, 1.64e-06,
1.28e-06, 1.28e-06, 1.28e-06, 1.28e-06, 1.28e-06, 1.28e-06, 1.28e-06, 1.28e-06]
[]
43900     [2.42e-06, 2.64e-06, 1.14e-06, 2.50e-06, 3.47e-06, 3.20e-08, 2.30e-06,
1.80e-06, 1.53e-06, 8.50e-08, 9.93e-06, 1.49e-05, 4.84e-08, 7.03e-07, 1.18e-08,
3.50e-06, 1.79e-04, 7.00e-04, 4.42e-04, 3.44e-06, 4.11e-05, 1.82e-06, 1.64e-06,
1.51e-05, 1.66e-07, 1.10e-06, 6.11e-09, 9.87e-11, 5.84e-06, 2.70e-06, 2.47e-06,
2.86e-06, 2.12e-08, 6.06e-07, 1.16e-06, 1.22e-07, 1.22e-07, 1.22e-07, 1.22e-07,
1.22e-07, 1.22e-07, 1.22e-07, 1.22e-07]    [2.42e-06, 2.64e-06, 1.14e-06,
2.50e-06, 3.47e-06, 3.20e-08, 2.30e-06, 1.80e-06, 1.53e-06, 8.50e-08, 9.93e-06,
1.49e-05, 4.84e-08, 7.03e-07, 1.18e-08, 3.50e-06, 1.79e-04, 7.00e-04, 4.42e-04,
3.44e-06, 4.11e-05, 1.82e-06, 1.64e-06, 1.51e-05, 1.66e-07, 1.10e-06, 6.11e-09,
9.87e-11, 5.84e-06, 2.70e-06, 2.47e-06, 2.86e-06, 2.12e-08, 6.06e-07, 1.16e-06,
1.22e-07, 1.22e-07, 1.22e-07, 1.22e-07, 1.22e-07, 1.22e-07, 1.22e-07, 1.22e-07]
[]
44000     [2.46e-06, 3.06e-06, 1.10e-06, 3.63e-06, 4.12e-06, 3.09e-08, 2.25e-06,
1.82e-06, 7.19e-08, 7.09e-08, 3.44e-06, 3.05e-05, 1.57e-06, 1.21e-06, 3.90e-07,
1.66e-05, 1.44e-04, 6.77e-04, 4.16e-04, 6.11e-06, 4.57e-05, 4.54e-05, 1.21e-07,
1.11e-05, 3.99e-07, 4.16e-06, 8.94e-09, 2.71e-09, 5.38e-06, 1.56e-06, 7.30e-07,
6.07e-06, 1.91e-06, 9.58e-07, 1.13e-06, 2.76e-06, 2.76e-06, 2.76e-06, 2.76e-06,
2.76e-06, 2.76e-06, 2.76e-06, 2.76e-06]    [2.46e-06, 3.06e-06, 1.10e-06,
3.63e-06, 4.12e-06, 3.09e-08, 2.25e-06, 1.82e-06, 7.19e-08, 7.09e-08, 3.44e-06,
3.05e-05, 1.57e-06, 1.21e-06, 3.90e-07, 1.66e-05, 1.44e-04, 6.77e-04, 4.16e-04,
6.11e-06, 4.57e-05, 4.54e-05, 1.21e-07, 1.11e-05, 3.99e-07, 4.16e-06, 8.94e-09,
2.71e-09, 5.38e-06, 1.56e-06, 7.30e-07, 6.07e-06, 1.91e-06, 9.58e-07, 1.13e-06,
2.76e-06, 2.76e-06, 2.76e-06, 2.76e-06, 2.76e-06, 2.76e-06, 2.76e-06, 2.76e-06]
[]

Best model at step 19822:
  train loss: 1.72e-04
  test loss: 1.72e-04
  test metric: []

'train' took 52.550540 s
```

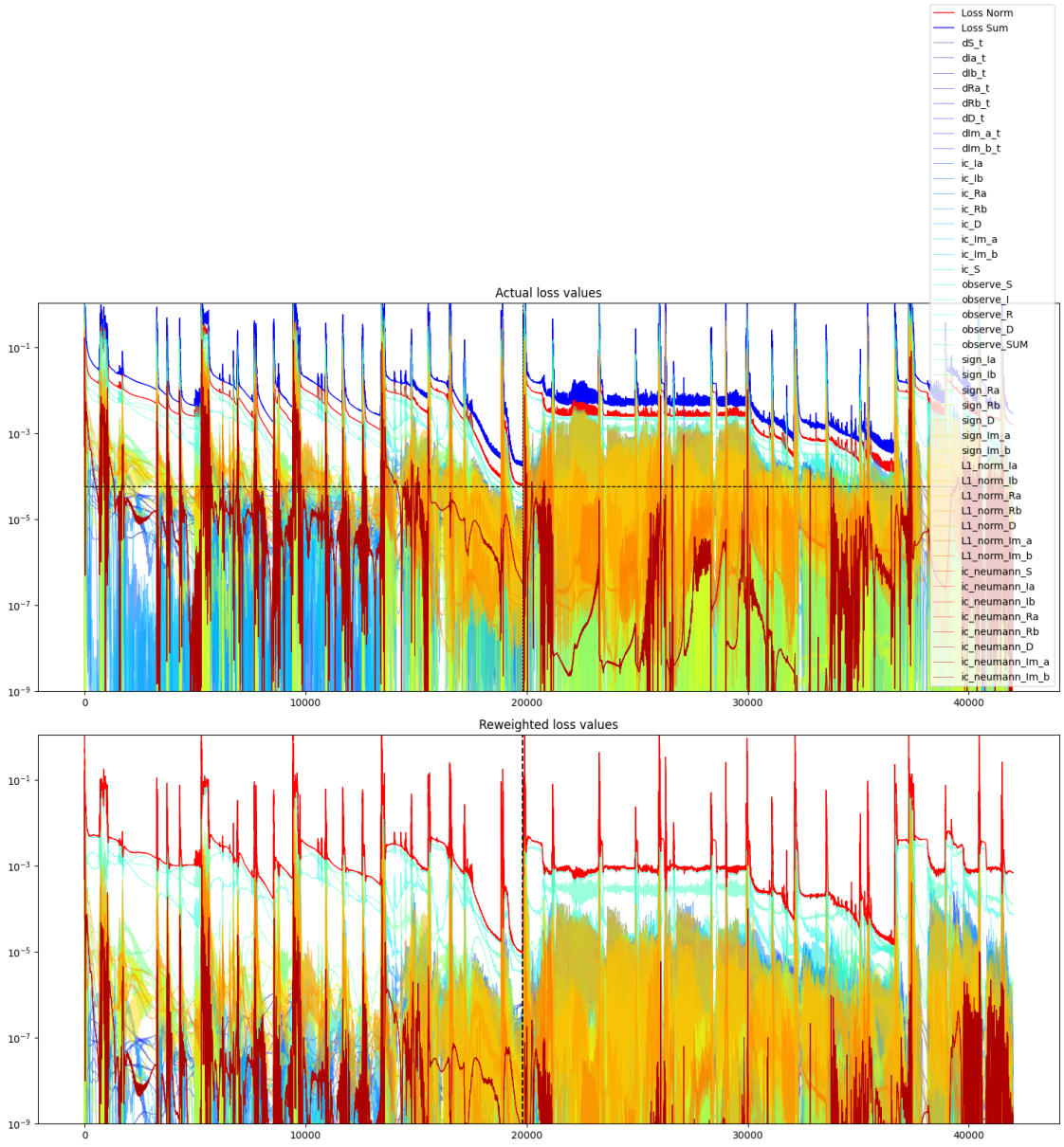
[ ]:
```
params_nn, best_step = model.get_best_params(out_func=np.exp) # parameters need␣
 ↪to be extracted with the exponential functino as they have been modelled in␣
 ↪logspace
t_nn_param, wsol_nn_param, wsol_sird_nn_param = solver(*params_nn)
# params_nn= tuple(np.exp([*params_nn]))
```

```
# print(*params_nn)
model.set_synthetic_data(t_synth, solution_synth_full)
model.set_nn_synthetic_data(t_nn_param, wsol_nn_param, wsol_sird_nn_param)
print(static_parameters, sep="\n")
plot = Plot(model, values_to_plot=sird_model.initial_conditions_keys) # class␣
 ↪that contains plotting functions
plot.show_known_and_prediction()
plot.plot_param_history()
plot.plot_loss_history()
```

```
Best train step: 19822
alpha_a: 0.2612912224939594
alpha_b: 0.067623923243795
beta_a: 0.12640536541080158
beta_b: 0.03993615570019145
gamma_a: 0.0003307768160708097
gamma_b: 0.0023793391400612515
kappa_a: 0.01818129764622366
kappa_b: 0.7802866784337122
{'alpha_a': 0.23, 'alpha_b': 0.18, 'beta_a': 0.1, 'beta_b': 0.08, 'gamma_a':
0.0, 'gamma_b': 0.0, 'kappa_a': 0.8, 'kappa_b': 0.5}
```




alpha_a
alpha_b
beta_a
beta_b
gamma_a
gamma_b
kappa_a
kappa_b
0.8
0.6
0.4
0.2
0.0
0
10000
20000
30000
40000



[ ]:

[ ]:
```
# from SIRD_normal_nn import SIRD_net
# max_timestep = 100000
# t_bool = t_synth < max_timestep
# t = t_synth[t_bool]
# wsol = solution_synth[t_bool]

# net = SIRD_net(t, wsol, init_num_people=1e6)
# net.train()
```

```
# net.plot(t_synth, solution_synth)
```

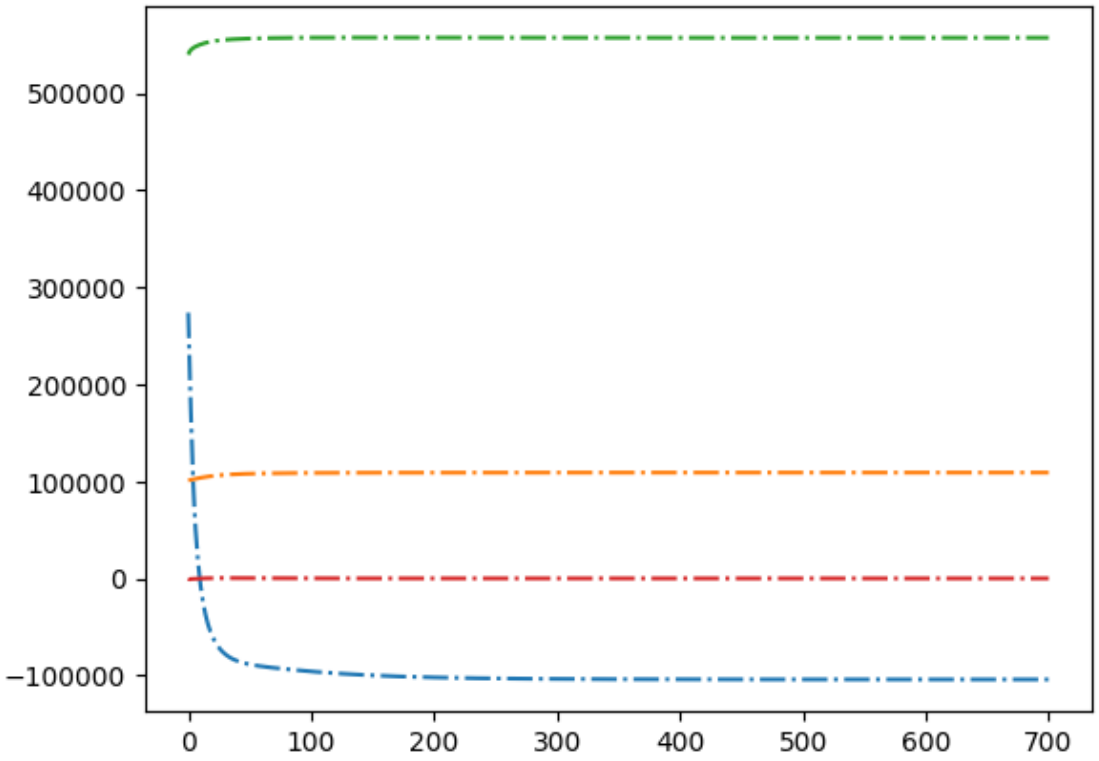
```
Epoch  1 : Train Loss 0.196515
Epoch 101 : Train Loss 1.731265
Epoch 201 : Train Loss 1.556680
Epoch 301 : Train Loss 1.514396
Epoch 401 : Train Loss 1.433833
Epoch 501 : Train Loss 1.515973
Epoch 601 : Train Loss 1.775939
Epoch 701 : Train Loss 0.587267
Epoch 801 : Train Loss 1.529438
Epoch 901 : Train Loss 1.554050
Epoch 1001 : Train Loss 1.645776
Epoch 1101 : Train Loss 1.636393
Epoch 1201 : Train Loss 1.485456
Epoch 1301 : Train Loss 1.512465
Epoch 1401 : Train Loss 1.869238
Epoch 1501 : Train Loss 1.595362
Epoch 1601 : Train Loss 1.961178
Epoch 1701 : Train Loss 1.520701
Epoch 1801 : Train Loss 1.552181
Epoch 1901 : Train Loss 1.191245
Epoch 2001 : Train Loss 1.267965
Epoch 2101 : Train Loss 0.744110
Epoch 2201 : Train Loss 0.897524
Epoch 2301 : Train Loss 1.668186
Epoch 2401 : Train Loss 1.638371
Epoch 2501 : Train Loss 1.439987
Epoch 2601 : Train Loss 1.501633
Epoch 2701 : Train Loss 1.618403
Epoch 2801 : Train Loss 1.379131
Epoch 2901 : Train Loss 1.559366
Epoch 3001 : Train Loss 1.849974
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[8], line 8
      5 wsol = solution_synth[t_bool]
      7 net = SIRD_net(t, wsol, init_num_people=1e6)
----> 8 net.train()
      9 net.plot(t_synth, solution_synth)

File ~/Library/Mobile Documents/com~apple~CloudDocs/DTU/11Semester Msc/Deep␣
 ↪Learning/Deep_Learning_Project_PINN/exercises_Jakob/SIRD_normal_nn.py:119, in␣
 ↪SIRD_net.train(self)
   115     batch_loss = criterion(output.flatten(), target.flatten())
   117     batch_loss.backward()#retain_graph=True)
```

```
--> 119     optimizer.step()
    120     cur_loss += batch_loss
    121 losses.append(cur_loss / batch_size)

File ~/Library/Mobile Documents/com~apple~CloudDocs/DTU/11Semester Msc/Deep␣
 ↪Learning/Deep_Learning_Project_PINN/.venv/lib/python3.9/site-packages/torch/
 ↪optim/optimizer.py:198, in Optimizer.profile_hook_step.<locals>.wrapper(*args ␣
 ↪**kwargs)
    194         else:
    195             raise RuntimeError(f"{func} must return None or a tuple of␣
 ↪(new_args, new_kwargs),"
    196                                f"but got {result}.")
--> 198 out = func(*args, **kwargs)
    199 self._optimizer_step_code()
    201 # call optimizer step post hooks

File ~/Library/Mobile Documents/com~apple~CloudDocs/DTU/11Semester Msc/Deep␣
 ↪Learning/Deep_Learning_Project_PINN/.venv/lib/python3.9/site-packages/torch/
 ↪optim/optimizer.py:29, in _use_grad_for_differentiable.<locals>.
 ↪_use_grad(self, *args, **kwargs)
     27 try:
     28     torch.set_grad_enabled(self.defaults['differentiable'])
---> 29     ret = func(self, *args, **kwargs)
     30 finally:
     31     torch.set_grad_enabled(prev_grad)

File ~/Library/Mobile Documents/com~apple~CloudDocs/DTU/11Semester Msc/Deep␣
 ↪Learning/Deep_Learning_Project_PINN/.venv/lib/python3.9/site-packages/torch/
 ↪optim/adam.py:258, in Adam.step(self, closure, grad_scaler)
    246     beta1, beta2 = group['betas']
    248     grad_scale, found_inf = self._init_group(
    249         group,
    250         grad_scaler,
   (…)
    255         max_exp_avg_sqs,
    256         state_steps)
--> 258     adam(params_with_grad,
    259          grads,
    260          exp_avgs,
    261          exp_avg_sqs,
    262          max_exp_avg_sqs,
    263          state_steps,
    264          amsgrad=group['amsgrad'],
    265          beta1=beta1,
    266          beta2=beta2,
    267          lr=group['lr'],
    268          weight_decay=group['weight_decay'],
    269          eps=group['eps'],
    270          maximize=group['maximize'],
```

```
    271          foreach=group['foreach'],
    272          capturable=group['capturable'],
    273          differentiable=group['differentiable'],
    274          fused=group['fused'],
    275          grad_scale=grad_scale,
    276          found_inf=found_inf)
    278 return loss

File ~/Library/Mobile Documents/com~apple~CloudDocs/DTU/11Semester Msc/Deep␣
 ↪Learning/Deep_Learning_Project_PINN/.venv/lib/python3.9/site-packages/torch/
 ↪optim/adam.py:324, in adam(params, grads, exp_avgs, exp_avg_sqs,␣
 ↪max_exp_avg_sqs, state_steps, foreach, capturable, differentiable, fused,␣
 ↪grad_scale, found_inf, amsgrad, beta1, beta2, lr, weight_decay, eps, maximize
    321 else:
    322     func = _single_tensor_adam
--> 324 func(params,
    325      grads,
    326      exp_avgs,
    327      exp_avg_sqs,
    328      max_exp_avg_sqs,
    329      state_steps,
    330      amsgrad=amsgrad,
    331      beta1=beta1,
    332      beta2=beta2,
    333      lr=lr,
    334      weight_decay=weight_decay,
    335      eps=eps,
    336      maximize=maximize,
    337      capturable=capturable,
    338      differentiable=differentiable,
    339      grad_scale=grad_scale,
    340      found_inf=found_inf)

File ~/Library/Mobile Documents/com~apple~CloudDocs/DTU/11Semester Msc/Deep␣
 ↪Learning/Deep_Learning_Project_PINN/.venv/lib/python3.9/site-packages/torch/
 ↪optim/adam.py:436, in _single_tensor_adam(params, grads, exp_avgs,␣
 ↪exp_avg_sqs, max_exp_avg_sqs, state_steps, grad_scale, found_inf, amsgrad,␣
 ↪beta1, beta2, lr, weight_decay, eps, maximize, capturable, differentiable)
    433 else:
    434     denom = (exp_avg_sq.sqrt() / bias_correction2_sqrt).add_(eps)
--> 436 param.addcdiv_(exp_avg, denom, value=-step_size)

KeyboardInterrupt:
```

[ ]:
```
# fig, ax = plt.subplots()
# net.plot(ax, t_synth)
```


500000
400000
300000
200000
100000
0
−100000
0
100
200
300
400
500
600
700

[ ]: