frontiers

Supplementary Material

From Epidemic to Pandemic Modelling
by Shannon Connolly, David Gilbert, Monika Heiner
Frontiers in Systems Biology, submitted January 2022

1 PETRINUTS PLATFORM – QUICK REFERENCE

1.1 The Framework

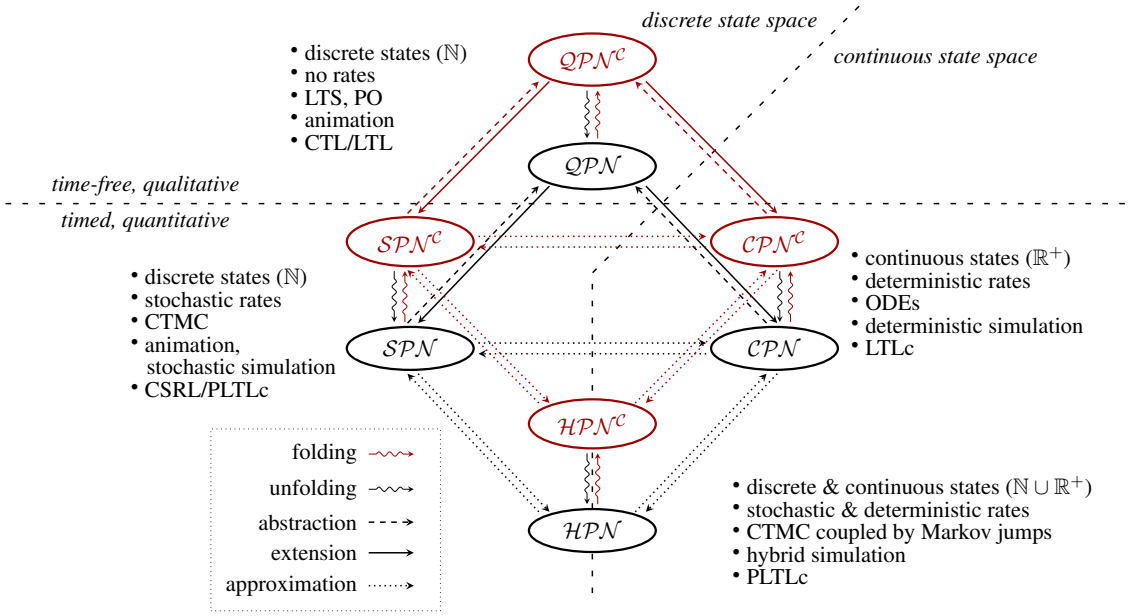

Figure S1. Paradigms of the unifying framework. Paradigms integrated in the unifying framework of the *PetriNuts* platform (adapted from Heiner et al. (2012)). The arrows describe the relations supported by automatic file transformation (export, unfolding); except *folding*, which in most cases has to be done manually.

Net classes given in Figure S1:

- $\mathcal{QPN}$, $\mathcal{QPN}^{\mathcal{C}}$- (coloured) qualitative Petri nets (standard Petri nets)
- $\mathcal{SPN}$, $\mathcal{SPN}^{\mathcal{C}}$- (coloured) stochastic Petri nets
- $\mathcal{CPN}$, $\mathcal{CPN}^{\mathcal{C}}$- (coloured) continuous Petri nets
- $\mathcal{HPN}$, $\mathcal{HPN}^{\mathcal{C}}$- (coloured) hybrid Petri nets

Remark: There are also fuzzy extensions of all six quantitative Petri nets Assaf et al. (2019).

Annotation index for Figure S1:

- interpretation of marking
- reaction rates
- semantics
- execution type: animation/simulation type
- temporal logic type corresponding to the given semantics to describe expected properties

For newcomers to the *PetriNuts* platform we provide guidelines for some basic use cases.

1.2 How to use *Snoopy*

Snoopy Heiner et al. (2012) is written in C++, using the Standard Template Library and the cross-platform toolkit wxWidgets WxWidgets (2021). The installation files, together with many other resources, are available at `https://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy`. *Snoopy* comes with a graphical user interface; thus all the following use cases start with *Open Snoopy*.

(1) How to obtain a graphical representation for a CANDL file:

- $File \rightarrow Import$ x.candl
- $Edit \rightarrow Layout \rightarrow$ chose layout algorithm
- $View \rightarrow Show\ attributes \rightarrow$ for all net elements, select the attributes you want to see
- $File \rightarrow Save$ x (as colspn or colcpn)

(2) How to obtain the unfolded net:

- $File \rightarrow Open$ x (colspn or colcpn)
- $File \rightarrow Export\ as$ andl file using dssd_util // involves unfolding
- $File \rightarrow Import$ x.andl $\rightarrow$ select *new model*
- $Edit \rightarrow Layout \rightarrow$ chose layout algorithm
- $File \rightarrow Save$ x (as spn or cpn)

Remarks:

- export of colspn (colcpn) to spn (cpn) also involves unfolding;
- alternatively, unfolding can be done via the command line tool *Spike*:

```
spike load -f=x.candl unfold save -f=x.andl
```

(3) How to convert $\mathcal{SPN} \Leftrightarrow \mathcal{CPN}$ or $\mathcal{SPN}^{\mathcal{C}} \Leftrightarrow \mathcal{CPN}^{\mathcal{C}}$:

- $File \rightarrow Open$ x.spn
- $File \rightarrow Export\ as$ x.cpn
- $File \rightarrow Open$ x.cpn

Remarks: or vice versa; likewise for $\mathcal{SPN}^{\mathcal{C}}$, $\mathcal{CPN}^{\mathcal{C}}$;

(4) How to obtain (C)ANDL files:

- $File \rightarrow Open$ x (spn/cpn)
- $File \rightarrow Export$ to ANDL
  or
- $File \rightarrow Open$ x (colspn/colcpn)
- $File \rightarrow Export$ to CANDL

(5) How to obtain the ODEs of a given $\mathcal{CPN}$ or $\mathcal{CPN}^{\mathcal{C}}$:

- $File \rightarrow Open$ x (cpn/colcpn)
- $File \rightarrow Export \rightarrow$ *ODEs to LaTeX* or *Text, Octave, Matlab*

Remarks:

- ODEs can also be exported as plain text by the simulation window's export options;
- $\mathcal{CPN}^{\mathcal{C}}$ have to be unfolded to $\mathcal{CPN}$, before generating the ODEs;

(6) How to reuse colour definitions:

- construct your (epidemic) model as $\mathcal{SPN}^{\mathcal{C}}$/$\mathcal{CPN}^{\mathcal{C}}$;
  alternatively, if you already have an uncoloured version x, do
  - $File \rightarrow Open$ x
  - $File \rightarrow export$ to corresponding coloured net class; i.e.,
    $\mathcal{SPN}$ can be exported to $\mathcal{SPN}^{\mathcal{C}}$, and $\mathcal{CPN}$ to $\mathcal{CPN}^{\mathcal{C}}$.
  - $File \rightarrow Open$ the coloured version of x
- $File \rightarrow Import$ y.candl // the file providing the required colour definitions
  - chose *Selective import*
  - chose *Select all* or individually select the subset of required definitions
  - press OK
- check if the colour definitions were imported as expected;

Remark: Advanced users may alternatively edit CANDL files with their favourite text editor.

(7) How to simulate with Snoopy:

- $File \rightarrow Open$ x (spn/colspn/cpn/colcpn)
- $View \rightarrow Start\ Simulation\ Mode$
- chose *Model Configuration*
- chose *Simulator Configuration*
- open some *Views* by double click
- click *Start Simulation*

Remark: For more details see Snoopy's Manuals Liu et al. (2012); Herajy et al. (2017) and related textbook chapters Marwan et al. (2012); Blätke et al. (2015).

1.3 How to use *Patty*

Patty is a JavaScript to support Petri net animation (token flow) in a web browser; it does not require any installation on the user's site. *Patty* reads (uncoloured) Petri nets in Snoopy's proprietary format.

All uncoloured Petri nets given in this paper can be animated in a purely qualitative manner at `https://www-dssz.informatik.tu-cottbus.de/DSSZ/Research/ModellingEpidemics`.

- A click on the picture opens a new window with a Petri net, directly executable in your web brower by help of Patty; nothing needs to be installed.
- Then, the PN can be animated by either clicking on individual transitions (boxes) or one of the smallish triangles in the control panel.
- Clicking on a place adds a token.

1.4 How to use *Charlie*

Charlie Heiner et al. (2015) is written in Java; thus it requires a Java runtime environment. The jar file is available at `https://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Charlie`. Charlie comes with a graphical user interface; thus all the following use cases start with *Open Charlie*.

- $file \rightarrow open$ x (andl)
- *net properties* – summary vector of the results achieved so far; acronyms are explained by a tool tip feature.
- *IM-based analysis* – computation of P-/T-invariants, which can be written to a text file (see *options*) and are read by *Snoopy* $\rightarrow Extras \rightarrow Load\ node\ set\ file$.
- *siphon/trap computation* – among others, bad siphons can be identified and written to a text file (click *export siphons*) and are read by *Snoopy* $\rightarrow Extras \rightarrow Load\ node\ set\ file$.
- *protocol* – a more verbose summary of the analysis results achieved so far.
- $file \rightarrow exit$ – all analysis results can be found in a log file, written to the folder from where the ANDL file had been loaded.

Remark: For more details see Franzke (2009); Heiner et al. (2015) and *Charlie*'s webpage.

1.5 How to use *Spike*

Spike Chodak and Heiner (2019) is a command line tool written in C++, available at `https://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Spike`. Thus all commands have to be entered in a terminal window.

(1) How to unfold:

Enter

```
spike load -f=x.candl unfold save -f=x.andl
```

(2) How to simulate:

For reproducible simulation experiments, *Spike* builds on a configuration file (to be written in its own configuration language), where all details of how to configure the model and the simulator(s) to be used need to be specified. Assuming the configuration file has the name *config.spc*, enter

```
spike exe -f=config.spc
```

Remarks:

- We provide configuration files to simulate P_{10}SIR with optimised kinetic rate constants and to produce Figure 11 illustrating dynamic rate constants.
- For more details see Chodak and Heiner (2019) and *Spike*'s webpage.

1.6 How to use *MC2*

MC2 Donaldson and Gilbert (2008) is written in Java; thus it requires a Java runtime environment. The jar file is available at `https://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/MC2`. *MC2* is a command line tool and performs offline simulative model checking of Probabilistic Linear-time Temporal Logic (PLTL). It requires as input a file of time series traces in CSV format, and a file of the PLTL properties to be checked. The property file contains one property per line, in the form

$P_{\geq 1}[\, G([x] = 0\,)\,]$ where x is an entity named (coloured/uncoloured place/transition/observer) in the traces file. The call is:

java -jar MC2v2.0beta2.jar det *TracesFile PropertiesFile*

and the output comprises, for each property checked, the property formula and on the next line the result (*true* or *false*).

We have compiled a library of useful properties which can be used directly or adapted as required. These library properties employ *meta variables* indicated by $x in the properties, to be substituted one by one by all observables in the input time series csv file of behaviour traces, e.g.

P>=1 [G([$x]>=0)]

We have developed a Unix script to automatically expand these meta variables to the observables in a given csv file so that there is a version of the properties for each observable, and then to run the model checker on the csv file and the expanded property file. The property library and Unix script are provided.

2 EXCHANGE FORMATS

2.1 ANDL – Abstract Net Description Language

ANDL has been designed as a concise, but human-readable exchange format for (uncoloured) Petri nets, written as plain ASCII text. For illustration of the ANDL syntax, we give a couple of examples; for formal definitions see Schwarick et al. (2016).

The very first keyword (*spn*/*cpn*) determines the net class – $\mathcal{SPN}$ or $\mathcal{CPN}$. An ANDL specification consists basically of three lists: the constants, the places and the transitions. Constants can hold several value sets and can be organised in groups (here: *marking* and *parameter*, which are pre-defined groups, extended by the user-defined groups *par_ratio*, *standard-incidence*). Models can be configured by choosing for each constant group a specific value set. For example, to apply the standard incidence, choose the *Main* value set (N will be set to $I0 + S0$), to apply Mass-action kinetics, choose *V_Set_0* (N will be set to 1).

Example 1: SIR.spn - Figure 1 in ANDL notation.

```
spn [SIR]
{
constants:
  valuesets[Main:V_Set_0:V_Set_1:V_Set_2:V_Set_3:V_Set_4:V_Set_5:V_Set_6]
marking:
  int S0 = [100:1e3:1e4:1e5:1e6:1e7:1e8:];
  int I0 = [1:S0/100::::::];
standardIncidence:
  int N = [I0+S0:1::::::];
par_ratio:
  double ratio = [1:0.1:0.01:1.0e-3:2:3:4:5];
parameter:
  double k_infect = [1:0.1:0.01:1.0e-3:1.0e-4:1.0e-5:1.0e-6:];
  double k_recover = [k_infect*ratio:::::::];

places:
```

```
  Susceptible = S0;
  Infectious = I0;
  Recovered = 0;

transitions:
  Infect
    :
    : [Infectious + 2] & [Susceptible - 1] & [Infectious - 1]
    : MassAction(k_infect/N)
    ;
  Recover
    :
    : [Recovered + 1] & [Infectious - 1]
    : MassAction(k_recover)
    ;

}
```

Example 2: SIR.cpn - Figure 1 in ANDL notation.

In terms of ANDL syntax, the only difference between $\mathcal{SPN}$ and $\mathcal{CPN}$ (besides the very first keyword) is the keyword *continuous* (occurring after *places* and *transitions*) which reminds us that places and transitions of a $\mathcal{CPN}$ are treated as continuous nodes.

```
cpn [SIR]
{
constants:
  valuesets[Main:V_Set_0:V_Set_1:V_Set_2:V_Set_3:V_Set_4:V_Set_5:V_Set_6]
marking:
  int S0 = [100:1e3:1e4:1e5:1e6:1e7:1e8:];
  int I0 = [1:S0/100::::::];
standardIncidence:
  int N = [I0+S0:1::::::];
par_ratio:
  double ratio = [1:0.1:0.01:1.0e-3:2:3:4:5];
parameter:
  double k_infect = [1:0.1:0.01:1.0e-3:1.0e-4:1.0e-5:1.0e-6:];
  double k_recover = [k_infect*ratio:::::::];

places:
continuous:
  Susceptible = S0;
  Infectious = I0;
  Recovered = 0;

transitions:
```

```
continuous:
  Infect
    :
    : [Infectious + 2] & [Susceptible - 1] & [Infectious - 1]
    : MassAction(k_infect/N)
    ;
  Recover
    :
    : [Recovered + 1] & [Infectious - 1]
    : MassAction(k_recover)
    ;

}
```

Example 3: SIR-S2.spn - Figure 4 (Top) in ANDL notation.

```
spn  [SIR-S2]
{
constants:
  valuesets[Main:V_Set_0:V_Set_1:V_Set_2:V_Set_3:V_Set_4:V_Set_5:V_Set_6]
marking:
  int SY0 = [50000:::::::];
  int SO0 = [50000:::::::];
  int IY0 = [1:::::::];
  int IO0 = [0:::::::];
standardIncidence:
  int N = [SY0+SO0+IY0+IO0:1::::::];
par_ratio:
  double ratio = [1:0.1:0.01:1.0e-3:2:3:4:5];
parameter:
  double k_infect = [1:0.1:0.01:1.0e-3:1.0e-4:1.0e-5:1.0e-6:];
  double k_recover = [k_infect*ratio:::::::];
  double k_crossInfect = [k_infect/0.1:::::::];

places:
  Susceptible_Young = SY0;
  Susceptible_Old = SO0;
  Infectious_Young = IY0;
  Infectious_Old = IO0;
  Recovered_Young = 0;
  Recovered_Old = 0;

transitions:
  InfectYoung
    :
    : [Infectious_Young + 2] & [Susceptible_Young - 1] & [Infectious_Young - 1]
```

```
    : MassAction(k_infect/N)
    ;
  RecoverYoung
    :
    : [Recovered_Young + 1] & [Infectious_Young - 1]
    : MassAction(k_recover)
    ;
  RecoverOld
    :
    : [Recovered_Old + 1] & [Infectious_Old - 1]
    : MassAction(k_recover)
    ;
  InfectOld
    :
    : [Infectious_Old + 2] & [Infectious_Old - 1] & [Susceptible_Old - 1]
    : MassAction(k_infect/N)
    ;
  OldInfectYoung
    :
    : [Infectious_Young + 1] & [Infectious_Old + 1] &
      [Susceptible_Young - 1] & [Infectious_Old - 1]
    : MassAction(k_crossInfect/N)
    ;
  YoungInfectOld
    :
    : [Infectious_Young + 1] & [Infectious_Old + 1] &
      [Susceptible_Old - 1] & [Infectious_Young - 1]
    : MassAction(k_crossInfect/N)
    ;

}
```

2.2 CANDL – Coloured ANDL

CANDL is an extension of ANDL; it has been designed as a concise, but human-readable exchange format for coloured Petri nets, written as plain ASCII text. For illustration of the CANDL syntax, we give two examples; for formal definitions see Assaf et al. (2021).

Example 4 shows the coloured version of Example 3, i.e., unfolding the CANDL code in Example 4 gives the ANDL code in Example 3. The naming convention for unfolded places is exactly as shown here (name of the coloured place followed by one of its colours, separated by an underscore), the naming of unfolded transitions slightly differs.

The rates of the transition *Infect* are colour-dependent; the appropriate colours are determined by a Boolean expression, forming a guard, given in square brackets.

Example 4 - SIR-S_enum.colspn - Figure 4 (Middle) in CANDL notation.

```
colspn  [SIR-S2_enum]
{
constants:
    valuesets[Main:V_Set_0:V_Set_1:V_Set_2:V_Set_3:V_Set_4:V_Set_5:V_Set_6]
marking:
  int SY0 = [50000:::::::];
  int SO0 = [50000:::::::];
  int IY0 = [1:::::::];
  int IO0 = [0:::::::];
standardIncidence:
  int N = [SY0+SO0+IY0+IO0:1::::::];
par_ratio:
  double ratio = [1:0.1:0.01:1.0e-3:2:3:4:5];
parameter:
  double k_infect = [1:0.1:0.01:1.0e-3:1.0e-4:1.0e-5:1.0e-6:];
  double k_crossInfect = [k_infect/0.1:::::::];
  double k_recover = [k_infect*ratio:::::::];

colorsets:
  Dot = {dot};
  enum Strata = {Young,Old};

variables:
  Strata : x;
  Strata : y;

places:
discrete:
  Strata Infectious = IY0‘Young++IO0‘Old;
  Strata Recovered = 0‘Young++0‘Old;
  Strata Susceptible = SY0‘Young++SO0‘Old;

transitions:
  Infect
    :
    : [Infectious + {x++y}] & [Susceptible - {x}] & [Infectious - {y}]
    : [(x=Young&y=Young)|(x=Old&y=Old)] MassAction(k_infect/N) ++
      [(x=Young&y=Old)|(x=Old&y=Young)] MassAction(k_crossInfect/N)
    ;
  Recover
    :
    : [Recovered + {x}] & [Infectious - {x}]
    : MassAction(k_recover)
    ;
```

```
}
```

Example 5 is structurally equivalent to Example 4, but uses a different colour set to encode stratification. Example 4 defines an enumeration set, which can be extended by adding further keywords. In contrast, Example 5 defines an integer subset, ranging from 1 to an upper bound given by the constant *StrataNum*, which simplifies scaling to an arbitrary number of strata. In terms of implementation, there isn't much of a difference as enumeration types are internally mapped to integer types. Thus the keywords of an enumeration type (here: *Young*, *Old*) are merely syntactic sugar, which may improve readability. For further details of the supported colour sets and related colouring principles, see Liu et al. (2012).

Example 5 - SIR-S_int.colspn - Figure 4 (Middle) in CANDL notation.

```
colspn  [SIR-S2_int]
{
constants:
  valuesets[Main:V_Set_0:V_Set_1:V_Set_2:V_Set_3:V_Set_4:V_Set_5:V_Set_6]
coloring:
  int StrataNum = [2:::::::];
marking:
  int SY0 = [50000:::::::];
  int SO0 = [50000:::::::];
  int IY0 = [1:::::::];
  int IO0 = [0:::::::];
standardIncidence:
  int N = [SY0+SO0+IY0+IO0:1::::::];
par_ratio:
  double ratio = [1:0.1:0.01:1.0e-3:2:3:4:5];
parameter:
  double k_infect = [1:0.1:0.01:1.0e-3:1.0e-4:1.0e-5:1.0e-6:];
  double k_crossInfect = [k_infect/0.1:::::::];
  double k_recover = [k_infect*ratio:::::::];

colorsets:
  Dot = {dot};
  Strata = {1..StrataNum};

variables:
  Strata : x;
  Strata : y;

places:
discrete:
  Strata Infectious = IY0`1++IO0`2;
  Strata Recovered = 0`1++0`2;
  Strata Susceptible = SY0`1++SO0`2;

transitions:
```

```
  Infect
    :
    : [Infectious + {x++y}] & [Susceptible - {x}] & [Infectious - {y}]
    : [(x=1&y=1)|(x=2&y=2)] MassAction(k_infect/N) ++
      [(x=1&y=2)|(x=2&y=1)] MassAction(k_crossInfect/N)
    ;
  Recover
    :
    : [Recovered + {x}] & [Infectious - {x}]
    : MassAction(k_recover)
    ;

}
```

2.3 ODEs generated

For illustration we provide here for selected $\mathcal{CPN}$/$\mathcal{CPN}^{\mathcal{C}}$ the ODEs which are automatically generated when it comes to simulating a model; for $\mathcal{CPN}^{\mathcal{C}}$ this involves unfolding. To feed the ODEs to another simulator tool, consider the option to export to Matlab, Octave, or ERODE Cardelli et al. (2017); otherwise you may start from the plain text representation to obtain your required representation style; all of these exports are supported by *Snoopy*.

(1) ODEs - SIQR - Figure 3 (Top).

$$
\begin{aligned}
\frac{d\text{Susceptible}}{d\text{t}} &= -(\text{k\_infect}/\text{N} * \text{Susceptible} * \text{Infectious}) \\
\frac{d\text{Recovered}}{d\text{t}} &= +(\text{k\_recover} * \text{Infectious}) + (\text{k\_recover} * \text{Quarantine}) \\
\frac{d\text{Infectious}}{d\text{t}} &= +(k\text{\_infect} * \text{Susceptible} * \text{Infectious}) \\
&\quad -(k\text{\_recover} * \text{Infectious}) - (\text{k\_quar} * \text{Infectious}) \\
\frac{d\text{Quarantine}}{d\text{t}} &= +(\text{k\_quar} * \text{Infectious}) - (\text{k\_recover} * \text{Quarantine})
\end{aligned}
$$

(2) ODEs - SIAR - Figure 3 (Bottom).

$$\begin{aligned}
\frac{d\text{InfectiousSymptomatic}}{d\text{t}} &= +(k_\text{infect}/\text{N} * \text{Susceptible} * \text{InfectiousSymptomatic}) \\
&\quad +(k_\text{infect}/\text{N} * \text{Susceptible} * \text{InfectiousAsymptomatic}) \\
&\quad +(k_\text{symptoms} * \text{InfectiousAsymptomatic}) \\
&\quad -(k_\text{recover} * \text{InfectiousSymptomatic}) \\
\frac{d\text{RecoveredSymptomatic}}{d\text{t}} &= (\text{k_recover} * \text{InfectiousSymptomatic}) \\
\frac{d\text{Susceptible}}{d\text{t}} &= -2(\text{k_infect}/\text{N} * \text{Susceptible} * \text{InfectiousSymptomatic}) \\
&\quad +(k_\text{infect}/\text{N} * \text{Susceptible} * \text{InfectiousAsymptomatic}) \\
\frac{d\text{InfectiousAsymptomatic}}{d\text{t}} &= +(\text{k_infect}/\text{N} * \text{Susceptible} * \text{InfectiousSymptomatic}) \\
&\quad +(k_\text{infect}/\text{N} * \text{Susceptible} * \text{InfectiousAsymptomatic}) \\
&\quad -(k_\text{symptoms} * \text{InfectiousAsymptomatic}) \\
&\quad -(k_\text{recover} * \text{InfectiousAsymptomatic}) \\
\frac{d\text{RecoveredAsymptomatic}}{d\text{t}} &= (\text{k_recover} * \text{InfectiousAsymptomatic})
\end{aligned}$$

(3) ODEs - SIR-S2 - Figure 4.

$$\begin{aligned}
\frac{d\text{Infectious}_{\text{Young}}}{d\text{t}} &= +(k_\text{crossInfect}/\text{N} * \text{Susceptible_Young} * \text{Infectious_Old}) \\
&\quad +(k_\text{infect}/\text{N} * \text{Susceptible_Young} * \text{Infectious_Young}) \\
&\quad -(k_\text{recover} * \text{Infectious_Young}) \\
\frac{d\text{Recovered}_{\text{Young}}}{d\text{t}} &= +(\text{k_recover} * \text{Infectious_Young}) \\
\frac{d\text{Susceptible}_{\text{Young}}}{d\text{t}} &= -(\text{k_infect}/\text{N} * \text{Susceptible_Young} * \text{Infectious_Young}) \\
&\quad -(k_\text{crossInfect}/\text{N} * \text{Susceptible_Young} * \text{Infectious_Old}) \\
\frac{d\text{Susceptible}_{\text{Old}}}{d\text{t}} &= -(\text{k_infect}/\text{N} * \text{Infectious_Old} * \text{Susceptible_Old}) \\
&\quad -(k_\text{crossInfect}/\text{N} * \text{Susceptible_Old} * \text{Infectious_Young}) \\
\frac{d\text{Recovered}_{\text{Old}}}{d\text{t}} &= (\text{k_recover} * \text{Infectious_Old}) \\
\frac{d\text{Infectious}_{\text{Old}}}{d\text{t}} &= +(\text{k_crossInfect}/\text{N} * \text{Susceptible_Old} * \text{Infectious_Young}) \\
&\quad +(k_\text{infect}/\text{N} * \text{Infectious_Old} * \text{Susceptible_Old}) \\
&\quad -(k_\text{recover} * \text{Infectious_Old})
\end{aligned}$$

(4) ODEs - P_2SIR - Figure 5.

$$
\begin{aligned}
\frac{d\mathrm{S_{DE}}}{d\mathrm{t}} &= -(k\_\mathrm{infect}/\mathrm{N} * \mathrm{I\_DE} * \mathrm{S\_DE}) + (\mathrm{k\_travel} * \mathrm{S\_FR}) - (\mathrm{k\_travel} * \mathrm{S\_DE}) \\
\frac{d\mathrm{R_{DE}}}{d\mathrm{t}} &= +(\mathrm{k\_recover} * \mathrm{I\_DE}) + (\mathrm{k\_travel} * \mathrm{R\_FR}) - (\mathrm{k\_travel} * \mathrm{R\_DE}) \\
\frac{d\mathrm{I_{DE}}}{d\mathrm{t}} &= +(k\_\mathrm{infect}/\mathrm{N} * \mathrm{I\_DE} * \mathrm{S\_DE}) \\
&\quad +(\mathrm{k\_travel} * \mathrm{I\_FR}) \\
&\quad -(k\_\mathrm{recover} * \mathrm{I\_DE}) \\
&\quad -(k\_\mathrm{travel} * \mathrm{I\_DE}) \\
\frac{d\mathrm{S_{FR}}}{d\mathrm{t}} &= -(k\_\mathrm{infect}/\mathrm{N} * \mathrm{I\_FR} * \mathrm{S\_FR}) + (\mathrm{k\_travel} * \mathrm{S\_DE}) - (\mathrm{k\_travel} * \mathrm{S\_FR}) \\
\frac{d\mathrm{R_{FR}}}{d\mathrm{t}} &= +(\mathrm{k\_recover} * \mathrm{I\_FR}) + (\mathrm{k\_travel} * \mathrm{R\_DE}) - (\mathrm{k\_travel} * \mathrm{R\_FR}) \\
\frac{d\mathrm{I_{FR}}}{d\mathrm{t}} &= +(\mathrm{k\_infect}/\mathrm{N} * \mathrm{I\_FR} * \mathrm{S\_FR}) \\
&\quad +(k\_\mathrm{travel} * \mathrm{I\_DE}) \\
&\quad -(k\_\mathrm{recover} * \mathrm{I\_FR}) \\
&\quad -(k\_\mathrm{travel} * \mathrm{I\_FR})
\end{aligned}
$$

3 ADDITIONAL FIGURES AND TABLES

The following subsection titles correspond to the ones in the main paper.

3.1 Pandemic models

P_4SIR model, unfolded is given in Figure S2.

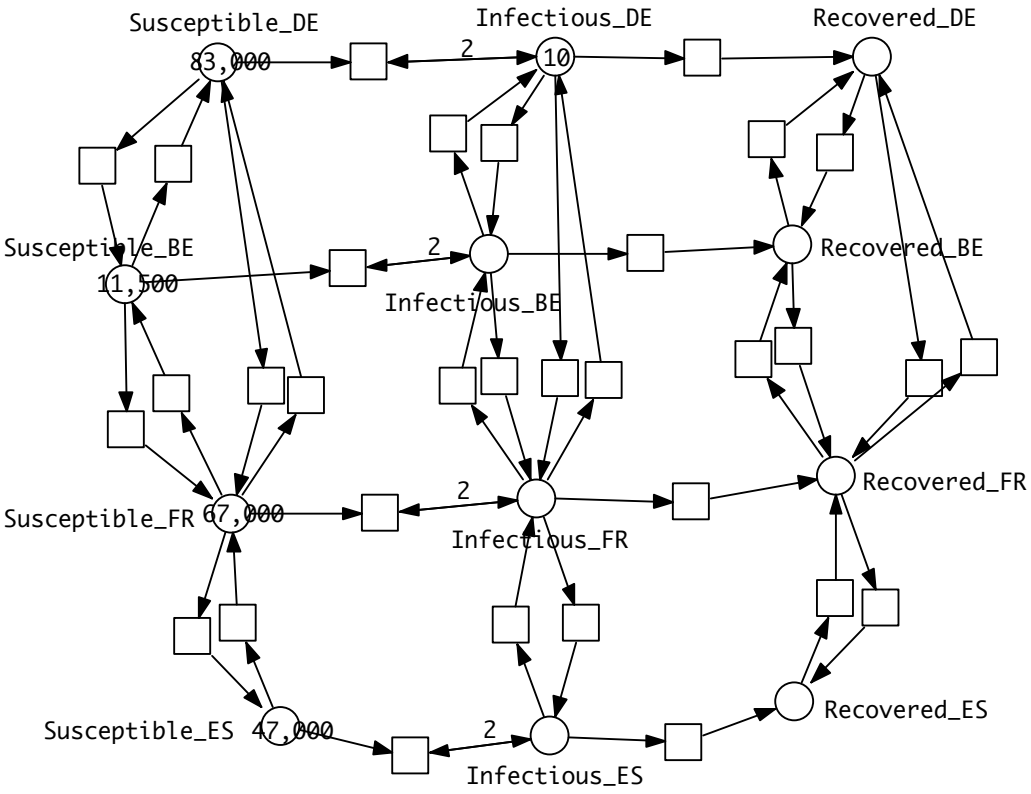

Figure S2. P_4SIR model, unfolded obtained by unfolding P_4SIR, shown in Figure 6 (Middle), layout automatically generated.

3.2 Combined models

P_{10}SIQR model in two versions – see Figure S3.

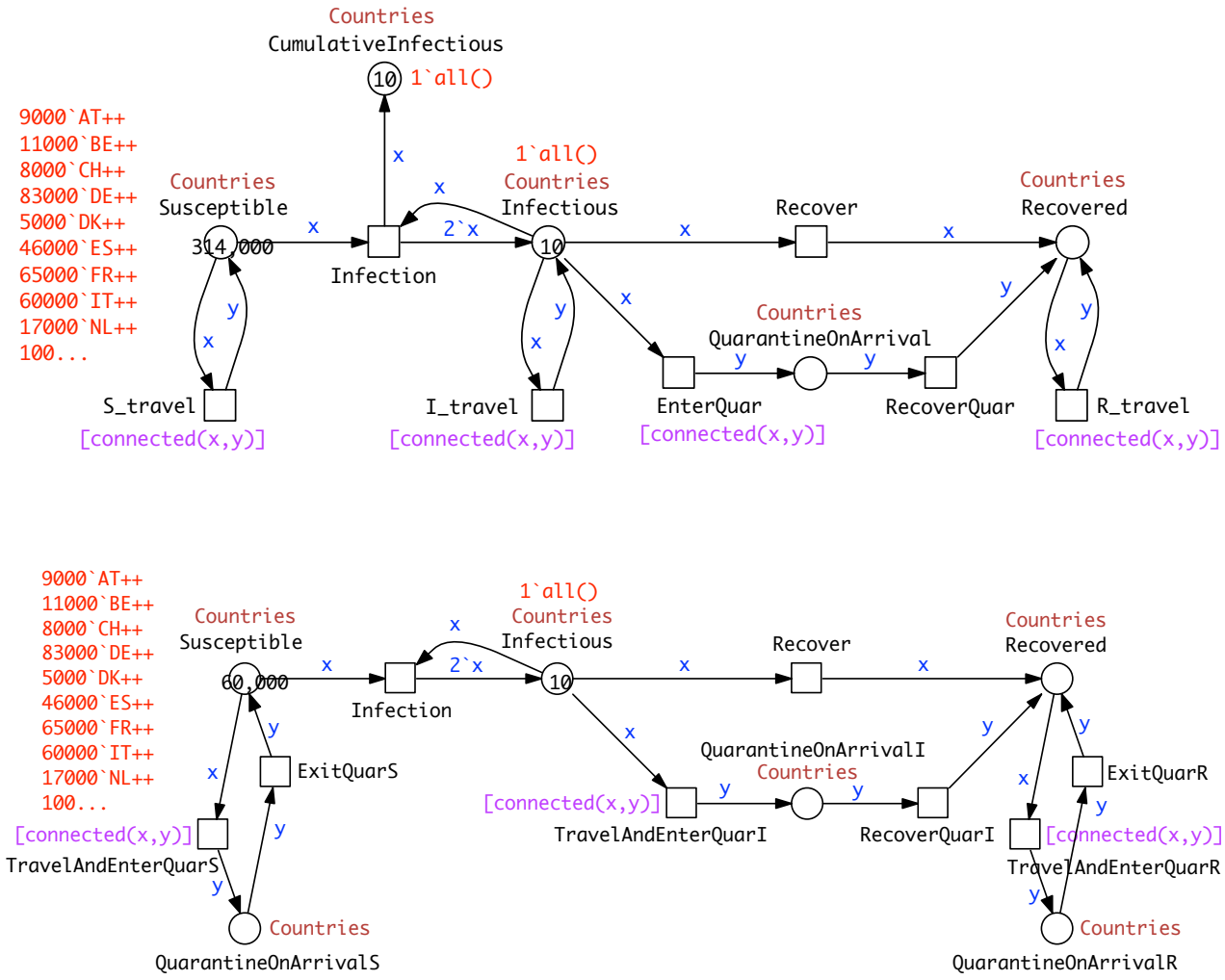

Figure S3. P_{10}QSIR model in two versions for West Europe (10 countries). **(Top)** $P_{10}Q_{I}SIR$. Arrival by *Infectious* may possibly involve quarantine. There is an additional place *CumulativeInfectious* to keep a record of total infections over time. Note: this additional place destroys the conservativeness and induces a second P-invariant, comprising (all uncoloured places of) *Susceptible* and *CumulativeInfectious*. A related analysis is given in Figure S5. **(Bottom)** P_{10}QSIR. All arrivals will involve quarantine.

3.3 Parameter fitting

Parameter scanning

Examples for varying rate parameters, see Figure S4 for SIQR, and Figure S5 for $P_{10}Q_ISIR$.

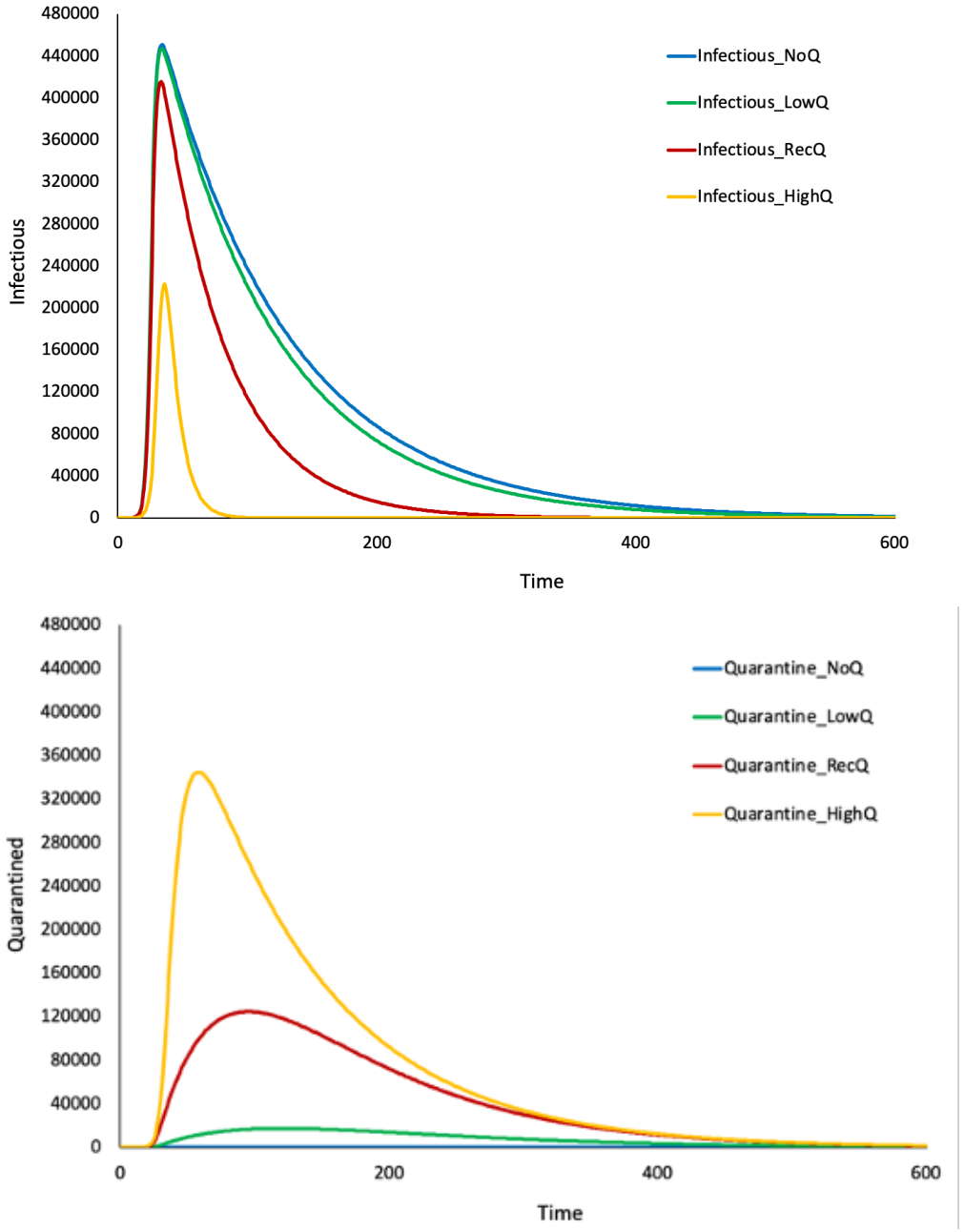

Figure S4. SIQR model with variable quarantine rates. Four simulations of model SIQR (Figure 3 (Top)) with different Quarantine rates. NoQ = 0, LowQ = 10x less than Recover, RecQ = Recover, HighQ = 10x greater than Recover. **(Top)** Infectious compartment, **(Bottom)** Quarantined compartment.

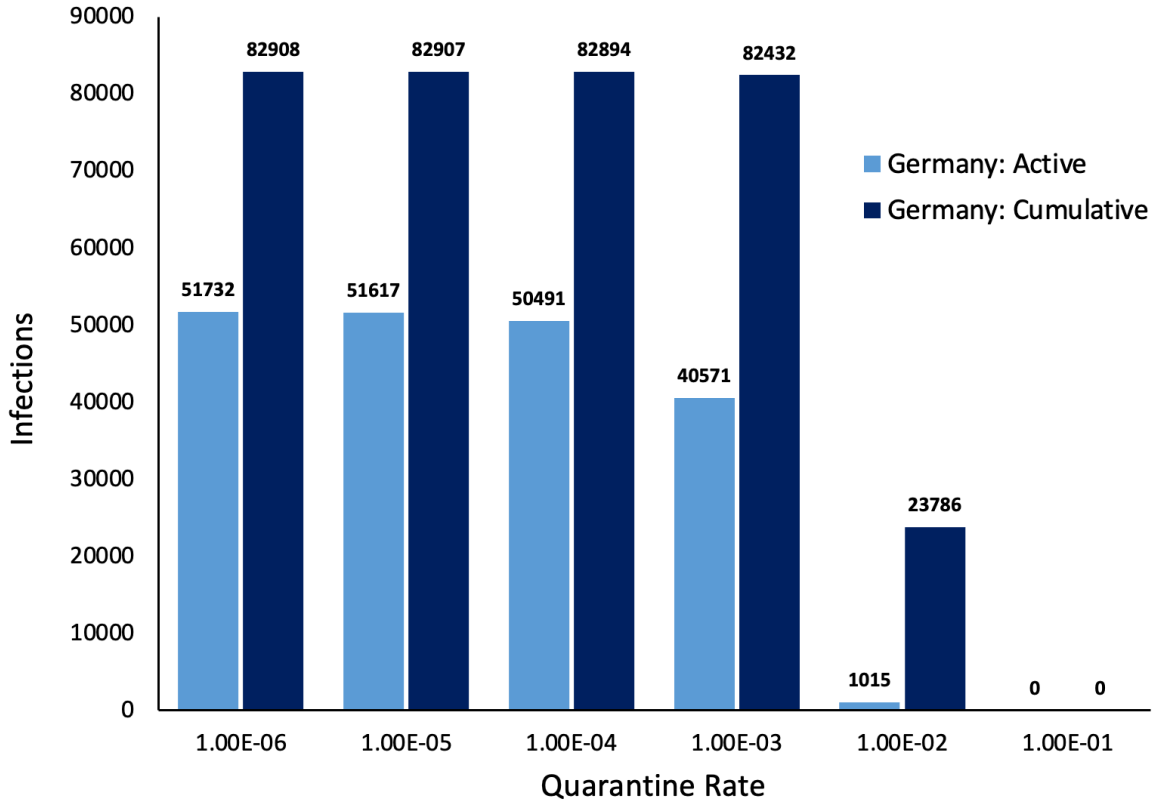

Figure S5. $P_{10}Q_ISIR$ model – effect of modifying quarantine rates. Active and CumulativeInfectious for Germany with varying rates of Quarantine. Output from model in Figure S3 (Top). X axis: rates for how many people go into quarantine. Y axis: Data for the peaks (active infectious at peak, cumulative at peak). Interpretation: up to a certain point quarantine only suppresses the active cases but not cumulative, lengthening out the curve, until quarantine is high.

Parameter optimisation

The model fitted was P_{10}SIR in the unfolded version (see Figure S6, and supplementary files P10SIR.candl, P10SIR_Separate_Rates.andl and P10SIR_Separate_Rates.cpn) employing a Random Restart Hill Climbing Algorithm (RRHCA) using real world data given in Table S1, with results for *Infection* rates given in Table S2 (Objective 1), Table S3 (Objective 2), and Table S4 (Objective 3). The *Connection* rates for Objective 3 are given in Table S5.

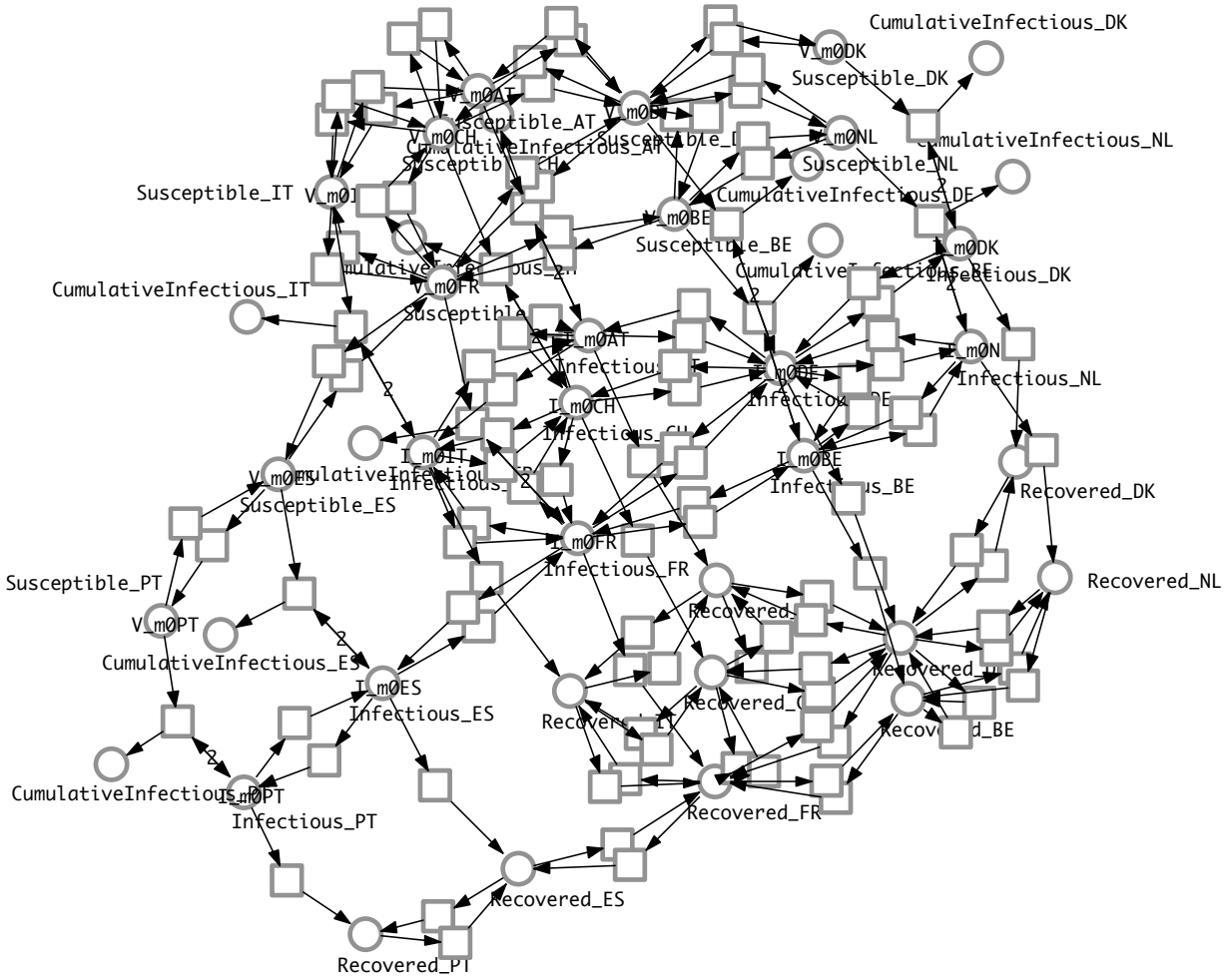

Figure S6. P_{10}SIR model. $\mathcal{CPN}$ for SIR, West Europe, with a *CumulativeInfectious* place to keep a record of total infections over time; obtained by unfolding and automatic layout.

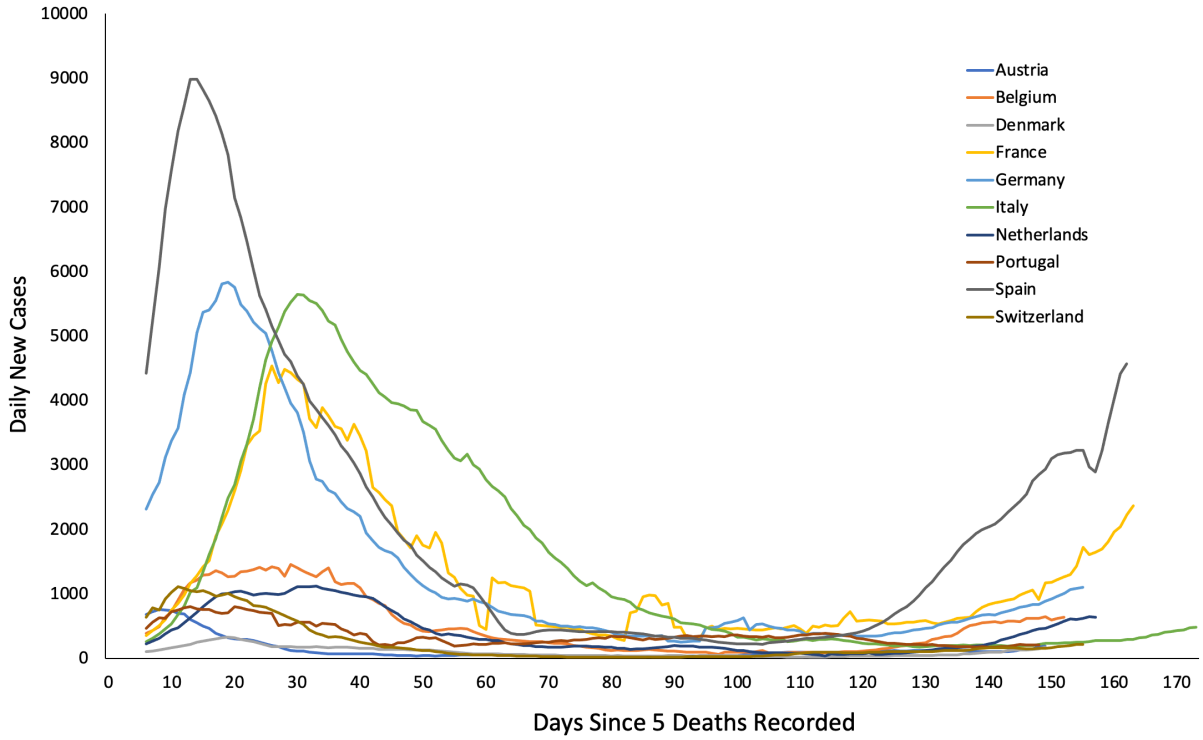

Figure S7. Data used to fit P_{10}SIR . Real world data: daily new cases for 10 Western Europe countries, source data from WorldOmeters (https://www.worldometers.info/coronavirus/) for 19th January to 16th August 2020. Data smoothed by using a 7-day rolling average. Time: day 0 is after 5 or more cumulative deaths. Peak is the raw number. The figure shows clearer peaks for countries which reported a higher number of cases. Data used to fit the model given in Figure S6.

Table S1. Real world data: peak magnitude and timing of daily new cases for 10 Western Europe countries, source data from WorldOmeters (https://www.worldometers.info/coronavirus/) for 19th January to 16th August 2020.

Country	Time	Peak
Austria	8	754
Switzerland	11	1112
Portugal	13	803
Spain	14	8987
Denmark	19	328
Germany	19	5837
France	26	4537
Belgium	29	1453
Italy	30	5646
Netherlands	33	1120

Table S2. Objective 1: Summary statistics for Random Restarts achieving the best Levenshtein Distance of 2 for model given in Figure S6. RR = Random Restart and n refers to the number of solutions producing the summary statistics.

	Infection Rates RR0 (n=202)			Infection Rates RR5 (n=186)			Infection Rates RR9 (n=315)		
Country	**Mean**	**Min**	**Max**	**Mean**	**Min**	**Max**	**Mean**	**Min**	**Max**
Austria	$8.34e^{-8}$	$3.24e^{-8}$	$1.69e^{-7}$	$4.05e^{-8}$	$2.33e^{-8}$	$6.72e^{-8}$	$5.92e^{-7}$	$1.69e^{-7}$	$1.03e^{-6}$
Belgium	$2.82e^{-6}$	$2.07e^{-6}$	$4.72e^{-6}$	$1.98e^{-6}$	$1.24e^{-6}$	$3.41e^{-6}$	$5.40e^{-7}$	$6.68e^{-8}$	$1.06e^{-6}$
Switzerland	$6.72e^{-6}$	$4.09e^{-6}$	$1.20e^{-5}$	$5.16e^{-7}$	$2.64e^{-7}$	$1.10e^{-6}$	$1.41e^{-7}$	$3.86e^{-9}$	$6.89e^{-7}$
Germany	$5.55e^{-7}$	$3.92e^{-7}$	$7.41e^{-7}$	$1.98e^{-5}$	$5.38e^{-6}$	$3.59e^{-5}$	$1.61e^{-6}$	$4.88e^{-7}$	$4.58e^{-6}$
Denmark	$6.19e^{-7}$	$1.45e^{-7}$	$1.54e^{-6}$	$2.93e^{-6}$	$4.90e^{-7}$	$9.79e^{-6}$	$1.05e^{-7}$	$3.51e^{-8}$	$2.46e^{-7}$
Spain	$2.53e^{-6}$	$9.18e^{-7}$	$1.00e^{-5}$	$1.49e^{-6}$	$8.47e^{-7}$	$2.52e^{-6}$	$3.83e^{-6}$	$1.01e^{-6}$	$9.52e^{-6}$
France	$6.11e^{-7}$	$4.80e^{-7}$	$8.72e^{-7}$	$4.83e^{-7}$	$2.46e^{-7}$	$7.26e^{-7}$	$7.47e^{-7}$	$3.44e^{-7}$	$1.49e^{-6}$
Italy	$4.72e^{-7}$	$2.98e^{-7}$	$7.00e^{-7}$	$2.99e^{-7}$	$2.23e^{-7}$	$6.02e^{-7}$	$3.45e^{-7}$	$1.83e^{-7}$	$5.68e^{-7}$
Netherlands	$5.71e^{-7}$	$3.11e^{-7}$	$1.28e^{-6}$	$8.31e^{-7}$	$6.16e^{-7}$	$1.40e^{-6}$	$7.81e^{-7}$	$6.10e^{-7}$	$1.14e^{-6}$
Portugal	$1.51e^{-7}$	$5.13e^{-8}$	$4.50e^{-7}$	$3.35e^{-5}$	$5.24e^{-6}$	$8.36e^{-5}$	$8.82e^{-5}$	$2.63e^{-5}$	$2.21e^{-4}$

Table S3. Objective 2: Infection rates output from the best solution from the RRHCA for model given in Figure S6. (left) Sorted alphabetically, (right) sorted by rate.

Country	**Infection Rate**
Austria	$3.64e^{-5}$
Belgium	$4.31e^{-5}$
Switzerland	$3.46e^{-6}$
Germany	$3.48e^{-6}$
Denmark	$6.94e^{-6}$
Spain	$1.80e^{-5}$
France	$4.44e^{-6}$
Italy	$6.57e^{-6}$
Netherlands	$2.36e^{-6}$
Portugal	$3.22e^{-5}$

Country	**Infection Rate**
Netherlands	$2.36e^{-6}$
Switzerland	$3.46e^{-6}$
Germany	$3.48e^{-6}$
France	$4.44e^{-6}$
Italy	$6.57e^{-6}$
Denmark	$6.94e^{-6}$
Spain	$1.80e^{-5}$
Portugal	$3.22e^{-5}$
Austria	$3.64e^{-5}$
Belgium	$4.31e^{-5}$

Table S4. Objective 3: Infection rates output from the best solution from the RRHCA for model given in Figure S6.(left) Sorted alphabetically, (right) sorted by rate.

Country	**Infection Rate**
Austria	$4.05e^{-5}$
Belgium	$8.03e^{-6}$
Switzerland	$3.49e^{-5}$
Germany	$2.89e^{-6}$
Denmark	$8.69e^{-6}$
Spain	$7.58e^{-6}$
France	$3.75e^{-6}$
Italy	$6.20e^{-6}$
Netherlands	$1.11e^{-6}$
Portugal	$3.49e^{-5}$

Country	**Infection Rate**
Netherlands	$1.11e^{-6}$
Germany	$2.89e^{-6}$
France	$3.75e^{-6}$
Italy	$6.20e^{-6}$
Spain	$7.58e^{-6}$
Belgium	$8.03e^{-6}$
Denmark	$8.69e^{-6}$
Portugal	$3.49e^{-5}$
Switzerland	$3.49e^{-5}$
Austria	$4.05e^{-5}$

Table S5. Objective 3: Travel rates output from the best solution from the RRHCA for model given in Figure S6 (Infection rates in Table S4). Columns: Connection from; Rows: Connection to.

	AT	BE	CH	DE	DK	ES	FR	IT	NL	PT
AT			$3.553e^{-6}$	$1.471e^{-6}$				$1.086e^{-5}$		
BE				$1.115e^{-6}$			$5.525e^{-6}$		$5.408e^{-7}$	
CH	$3.786e^{-6}$			$1.56e^{-6}$			$3.389e^{-7}$	$6.361e^{-6}$		
DE	$3.390e^{-6}$	$7.833e^{-6}$	$3.443e^{-6}$		$2.125e^{-6}$		$4.267e^{-6}$		$3.187e^{-6}$	
DK				$1.960e^{-6}$						
ES							$1.462e^{-6}$			$1.565e^{-6}$
FR		$1.291e^{-6}$	$1.425e^{-6}$	$1.082e^{-5}$		$2.630e^{-6}$		$7.151e^{-6}$		
IT	$2.535e^{-6}$		$2.529e^{-6}$				$1.75e^{-6}$			
NL		$9.549e^{-6}$		$2.342e^{-6}$						
PT						$2.071e^{-5}$				

3.4 Analysis

Correlation analysis

Table S6 records the variables, correlation matrices are given in Figure S8, S9, S10. The correlation matrices for the unfitted models indicate that the percent of population infected at peak is related to the population size for P_{48}SIR, but not for the smaller model P_{10}SIR, perhaps because there are more smaller countries in the smaller model. The number of connections is related to the peak and peak pecentage in the larger model, but only to the peak in the smaller model. Area is a partial proxy for the whole population and is strongly related to peak and time (negatively - i.e. earlier) but not to peak percentage in P_{10}SIR. However in P_{48}SIR area is related to peak and peak percentage but not to time.

However, the correlation over the fitted data indicates that the percentage of the population at the peak is less related to the population size, compared with the unfitted model. Also the size of the peak was less related to length of time from the start of infections in the fitted model compared to the unfitted model. Finally, the time of the peak of infections was not correlated with the number of inter-country connections in the fitted model, as opposed to being negatively correlated in the unfitted model (more connections implying earlier peaks), indicating that the intervention policies that governments have taken to control the epidemic in their countries have overriden the effect of geographical connections (i.e. closing down inter-country travel).

Table S6. Variables used to investigate road connected models further. Categories: (1) real world data external to model, (2) derived measure connecting external (black) to blue, (3) real world data incorporated in model, (4) data generated by model.

Category	Variable	Description
1	Area.Km2	Area in km^2 of a given country.
2	Density	Population density of a given country (population/km2).
3	Connections	Number of connections a country has in land borders.
	initSusc	Susceptible + Infected at time 0.
	Population	The real-world country populations.
	MapPop	The population used in the model (real world population / 1000).
4	peak	The peak of Infected, this represents active cases.
	peakPerc	The percentage of initSusc that were infected at the peak.
	time	The number of steps into the simulation the peak occurred.

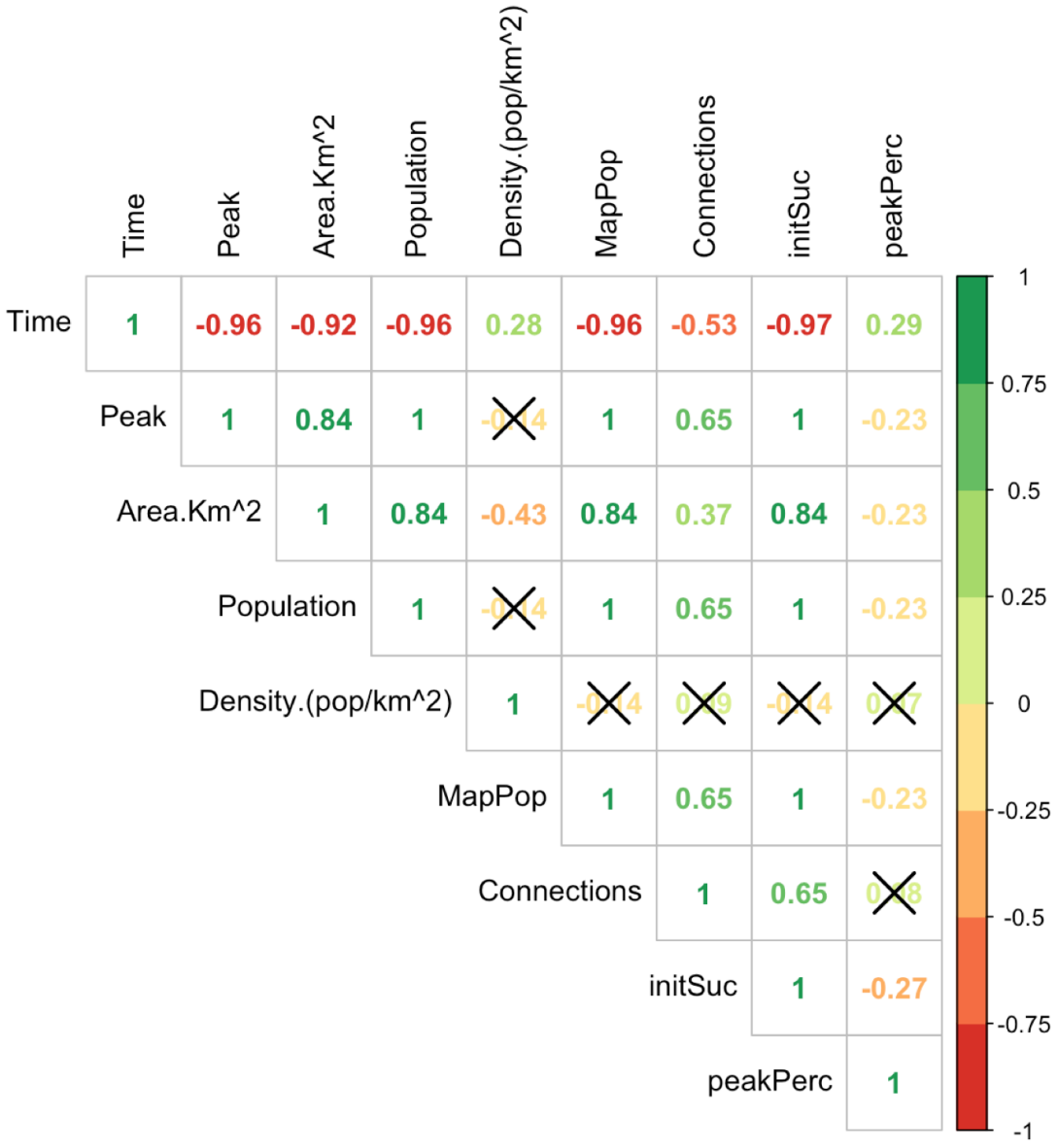

Figure S8. Correlation matrix for P_{10}SIR, not fitted. Green indicates a strong positive correlation and red indicates a strong negative correlation. Black crosses represent a non-significant correlation. Infection rates and travel rates all identical at $1.0e^{-6}$; recovery rates $1.0e^{-2}$.

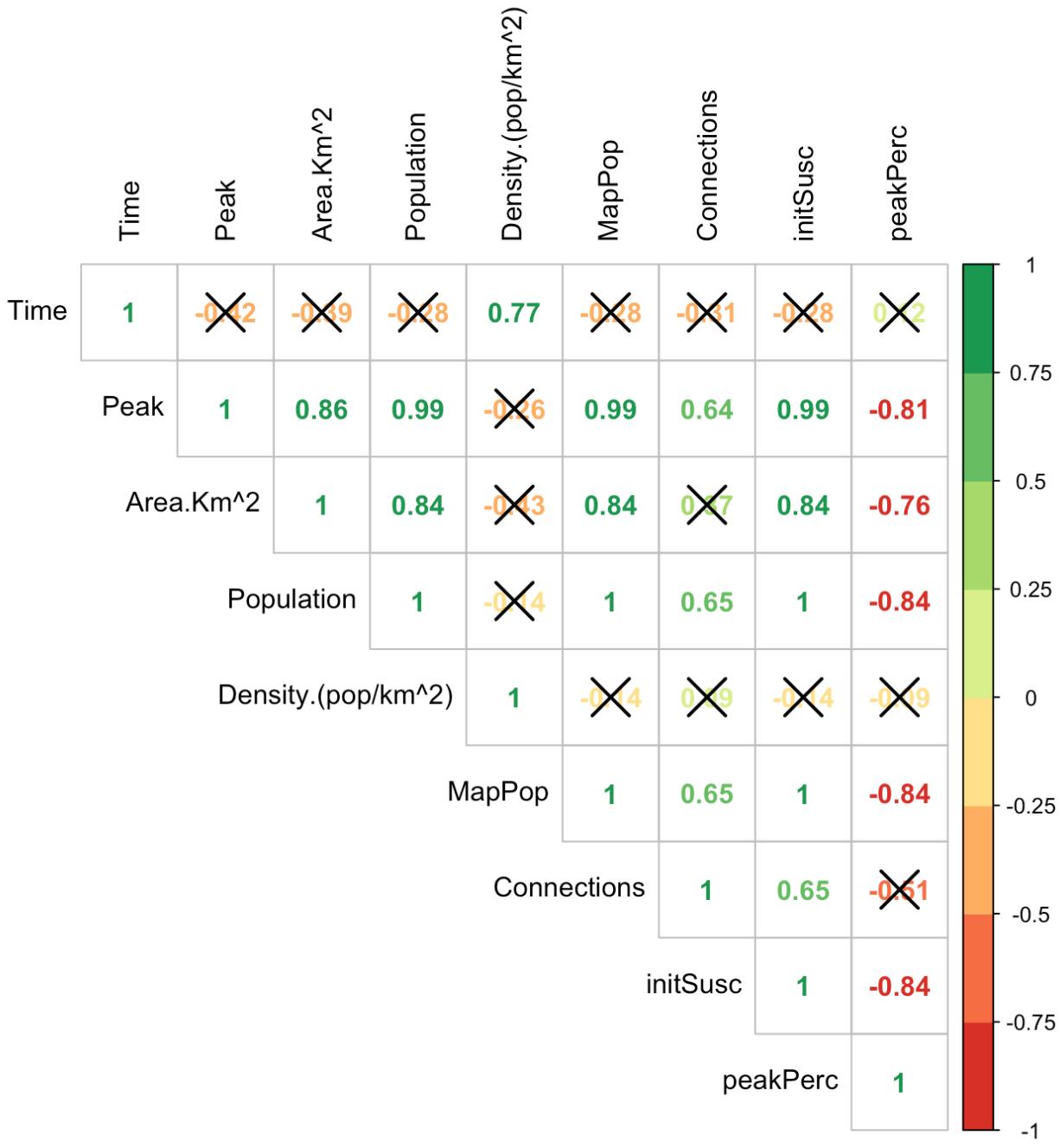

Figure S9. Correlation matrix for P_{10}SIR, fitted. Green indicates a strong positive correlation and red indicates a strong negative correlation. Black crosses represent a non-significant correlation.

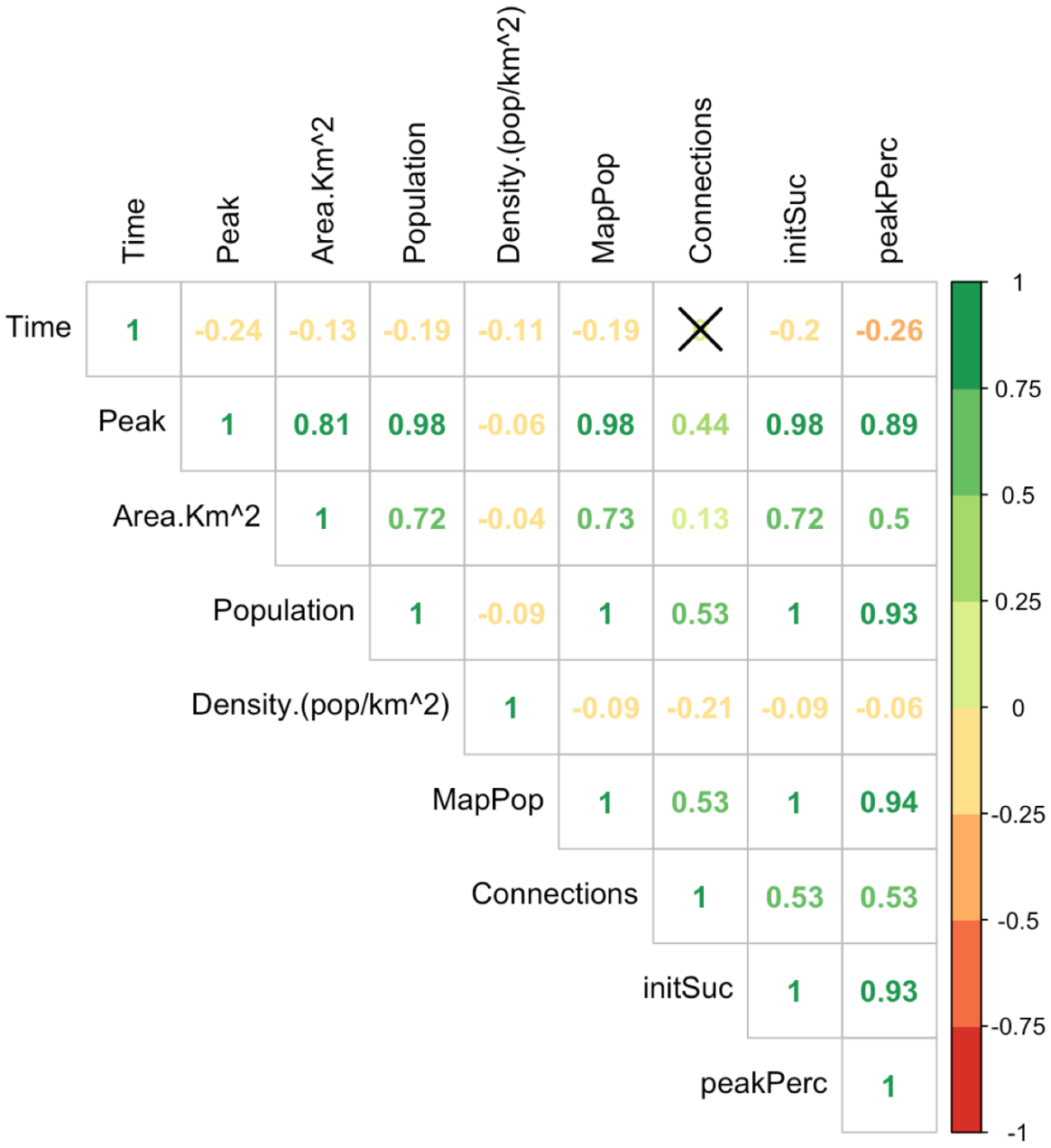

Figure S10. Correlation matrix for P_{48}SIR, not fitted. Green indicates a strong positive correlation and red indicates a strong negative correlation. Black crosses represent a non-significant correlation.

Dendrograms

Examples are given in Figures S11, S12.

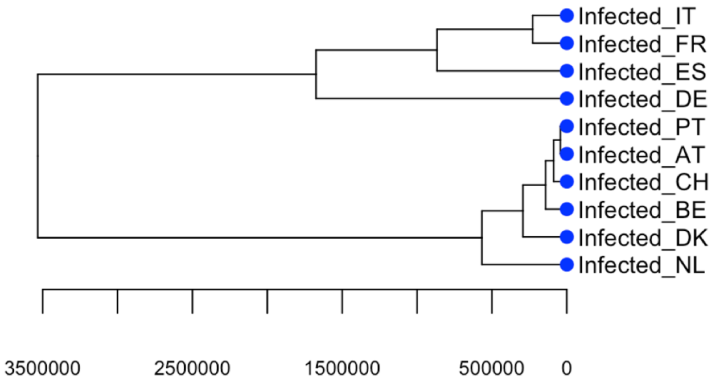

Figure S11. Dendrogram of P_{10}SIR for West Europe, hierarchical clustering. Height is on the x-axis. Data not normalised

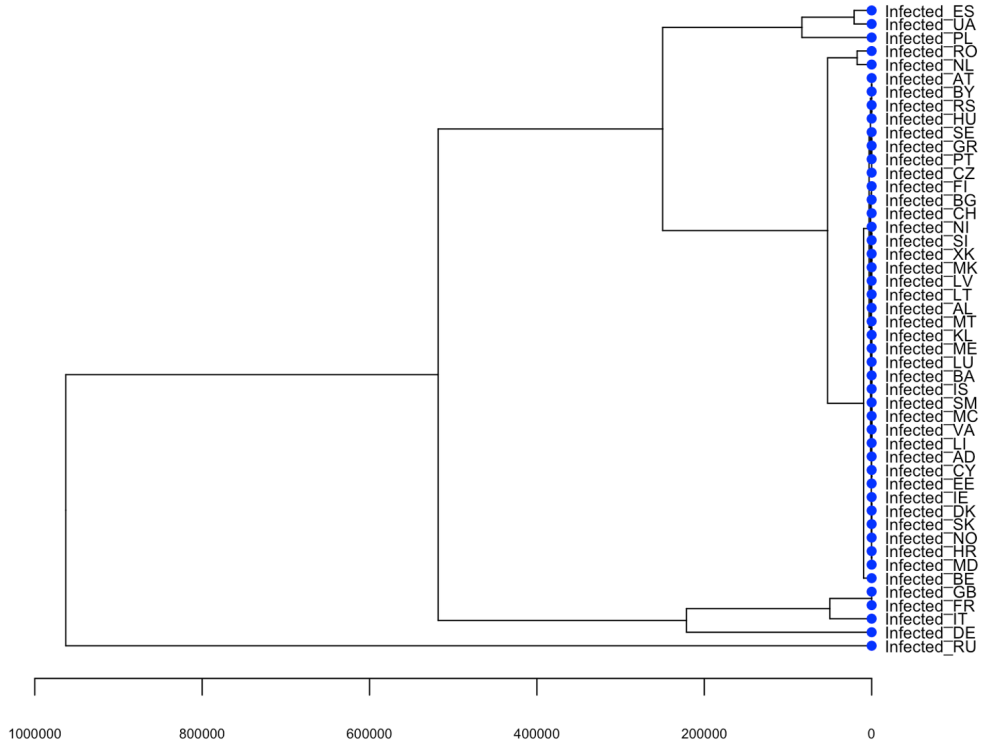

Figure S12. Dendrogram of P_{48}SIR for Europe, hierarchical clustering. Height is on the x-axis. Data not normalised.

4 LIST OF FILES PROVIDED

All models given in figures (provided as Snoopy files):

- **SIR.spn** - standard SIR, see Figure 1
- **SIQR.spn** - SIR extended by quarantine, see Figure 3 (Top)
- **SIAR.spn** - SIR extend by symptomatic/asymptomatic compartments, see Figure 3 (Bottom)
- **SIR-S2.spn** - SIR-S_2^{age}, uncoloured, see Figure 4 (Top)
- **SIR-S2_enum.colspn**, **SIR-S2_int.colspn** - SIR-S_2^{age}, coloured, see Figure 4 (Middle)
- **SIR-S20.spn** - SIR-S_{20}, see Figure 4 (Bottom)
- **P2-SIR.spn** - P_2SIR, uncoloured, see Figure 5 (Top)
- **P2-SIR.colspn** - P_2SIR, coloured, see Figure 5 (Bottom)
- **network-Europe4.spn** - connectivity graph for Europe, 4 countries, uncoloured, see Figure 6 (Top middle)
- **network-Europe4.colspn** - connectivity graph for Europe, 4 countries, coloured, see Figure 6 (Top right)
- **P4-SIR.colspn** - P_4SIR, coloured, see Figure 6 (Middle)
- **P4-SIR.spn** - P_4SIR, uncoloured, see Figure S2
- **P48-SIR-S10.colspn** - P_{48}SIR-S_{10}, see Figure 6 (Bottom)
- **P48-SIR-S10.andl.spn** - P_{48}SIR-S_{10} unfolded, see Figure 8 (Bottom)
- **SIVR.hpn** - SIR with variant virus, see Figure2
- **P10-QI-SIR.colspn** - P_{10}Q_ISIR, see Figure S3 (Top)
- **P10-QSIR.colspn** - P_{10}QSIR, see Figure S3 (Bottom)
- **P10-SIR_Separate_Rates.cpn** - $\mathcal{CPN}$ used for parameter optimisation - unfolded version of P_{10}SIR, given in Figure S6.

Connectivity networks (provided as CANDL files):

- **network-Europe04.candl**
  - unfolding: |P|=4, |T|=8, |A|=16
- **network-Europe05.candl**
  - unfolding: |P|=5, |T|=16, |A|=32
- **network-Europe10.candl**
  - unfolding: |P|=10, |T|=30, |A|=60
- **network-Europe48.candl**
  - unfolding: |P|=47, |T|=170, |A|=340
  - *note:* IS isolated, https://countrycode.org
  - unfolding time: 1.302sec
- **network-China.candl**
  - unfolding: |P|=33, |T|=142, |A|=286
  - *note:* 34 states; TW isolated
  - unfolding time: 0.283sec
- **network-USA.candl**
  - unfolding: |P|=48, |T|=210, |A|=420

- *note:* 50 states; AK, HI isolated
- unfolding time: 1.643sec

P_n SIR models (provided as CANDL files):

- P-Europe02-SIR - Europe fragment, 2 countries
- P-Europe04-SIR - Europe fragment, 4 countries
- P-Europe10-SIR - Western Europe, 10 countries
- P-Europe48-SIR - All Europe, 48 countries
- P-China-SIR - China, 34 provinces
- P-USA-SIR - USA, 50 states

Python programs:

- RRHCA (Random Restart Hill Climbing)
- Python Web Scraper, used to to collect COVID19 Data.

R code:

- Europe.Rmd, Western_Europe.Rmd: Both these files take in csv outputs from *Snoopy*, *Spike* to produce correlation matrices and regression analysis. The Time and *Infectious* traces are extracted from the csv files and joined to real world data (e.g. population, density, country size). Correlation matrices are then produced. Regression analysis is also conducted along with checking the assumptions of regression.
- Clustering.Rmd: This code takes in one csv file (output from *Snoopy*, *Spike*) and performs agglomerative hierarchical clustering using Euclidean distance and complete-linkage. This code selects Time and Infected traces to perform the clustering. The code plots a dendrogram of the clustering results for visualisation.

Spike:

- config file + andl/candl file
- all source files to produce Figure 11.

MC2:

- property library and Unix script to produce Figure 9.

REFERENCES

Assaf, G., Chodak, J., Rohr, C., Schwarick, M., and Heiner, M. (2021). *CANDL Report*. Tech. rep., Brandenburg University of Technology Cottbus, Department of Computer Science. To appear

Assaf, G., Heiner, M., and Liu, F. (2019). Biochemical reaction networks with fuzzy kinetic parameters in Snoopy. In *Proc. CMSB 2019*, eds. L. Bortolussi and G. Sanguinetti (Springer), vol. 11773 of *LNCS/LNBI*, 302–307. doi:10.1007/978-3-030-31304-3_17. First Online: 17 September 2019

Blätke, M.-A., Heiner, M., and Marwan, W. (2015). *BioModel Engineering with Petri Nets* (Elsevier Inc.), chap. 7. 141–193. doi:10.1016/B978-0-12-801213-0.00007-1

Cardelli, L., Tribastone, M., Tschaikowski, M., and Vandin, A. (2017). ERODE: A Tool for the Evaluation and Reduction of Ordinary Differential Equations. In *TACAS 2017* (Springer, LNCS 10206), 310–328

Chodak, J. and Heiner, M. (2019). Spike – reproducible simulation experiments with configuration file branching. In *Proc. CMSB 2019*, eds. L. Bortolussi and G. Sanguinetti (Springer), vol. 11773 of *LNCS/LNBI*, 315–321. doi:10.1007/978-3-030-31304-3_19

Donaldson, R. and Gilbert, D. (2008). A model checking approach to the parameter estimation of biochemical pathways. In *Proc. CMSB* (LNCS 5307, Springer), 269–287

Franzke, A. (2009). *Charlie 2.0 – a multi-threaded Petri net analyzer*. Diploma thesis, Brandenburg University of Technology Cottbus, Department of Computer Science

Heiner, M., Herajy, M., Liu, F., Rohr, C., and Schwarick, M. (2012). Snoopy - a unifying Petri net tool. In *Proc. PETRI NETS 2012* (Springer), vol. 7347 of *LNCS*, 398–407. doi:10.1007/978-3-642-31131-4_22

Heiner, M., Schwarick, M., and Wegener, J. (2015). Charlie - an extensible Petri net analysis tool. In *Proc. PETRI NETS 2015*, eds. R. Devillers and A. Valmari (Springer), vol. 9115 of *LNCS*, 200–211. doi:10.1007/978-3-319-19488-2_10

Herajy, M., Liu, F., Rohr, C., and Heiner, M. (2017). *Coloured Hybrid Petri Nets in Snoopy - User Manual*. Tech. Rep. 01-17, Brandenburg University of Technology Cottbus, Department of Computer Science

Liu, F., Heiner, M., and Rohr, C. (2012). *The Manual for Colored Petri Nets in Snoopy – QPN^C/SPN^C/CPN^C/$GHPN^C$*. Tech. Rep. 02-12, BTU Cottbus, Dep. of CS

Marwan, W., Rohr, C., and Heiner, M. (2012). *Petri nets in Snoopy: A unifying framework for the graphical display, computational modelling, and simulation of bacterial regulatory networks* (Humana Press), vol. 804 of *Methods in Molecular Biology*, chap. 21. 409–437. doi:10.1007/978-1-61779-361-5_21

Schwarick, M., Rohr, C., and Heiner, M. (2016). *MARCIE Manual*. Tech. Rep. 02-16, Brandenburg University of Technology Cottbus, Department of Computer Science

[Dataset] WxWidgets (2021). Wxwidgets website. `http://www.wxwidgets.org`. Accessed: 20/3/2021