Contents

Collection of dynamic systems animations

Nasser M. Abbasi

March 5, 2026 Compiled on March 5, 2026 at 10:10pm

1 Lorenz system

The following are animations of Lorenz system given by

$$
\begin{aligned}
x' &= -a(x+y) \\
y' &= rx - y - xz \\
z' &= -bz + xy
\end{aligned}
$$

More animations will be added later on.

2 Case 1. initial conditions $x(0) = -2, y(0) = -1, z(0) = 1$ and $a = 10, b = \frac{8}{3}, r = 28$

Time duration is 30 seconds. (Can take 10-20 seconds for animations to show up due to size)

3 Case 2. initial conditions $x(0) = -2, y(0) = -1, z(0) = 1$ and $a = 10, b = \frac{8}{3}, r = 15$

Time duration is 10 seconds. (Can take 10-20 seconds for animations to show up due to size)

4 SIRS infection model

The SIRS model is

$$\begin{aligned} \dot{S} &= -\beta S I + \mu R \\ \dot{I} &= \beta S I - \upsilon I \\ \dot{R} &= \upsilon I - \mu R \end{aligned} \tag{1}$$

Where $S = S(t)$ is the population of susceptible individuals, $I = I(t)$, the infected population, and $R = R(t)$ the recovered population. This diagram shows the model where now some of the recovered population can become susceptible again and become infected. The parameter μ indicates how much of the recovered population could become susceptible again.

The units of S, R, I are population measured in *person*. These values can not be negative since they are population amount. The units of μ is $\frac{1}{\text{time}}$ where time can be day or week or any other unit of time. The units of υ is also $\frac{1}{\text{time}}$. The units of β is $\frac{1}{(\text{time})(\text{person})}$

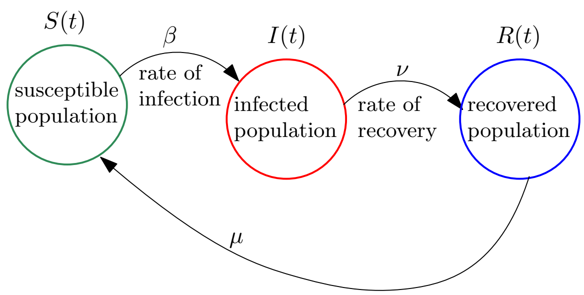

Figure 1: SIRS model

The following program allows one to do analysis on this model and it also displays the critical points.

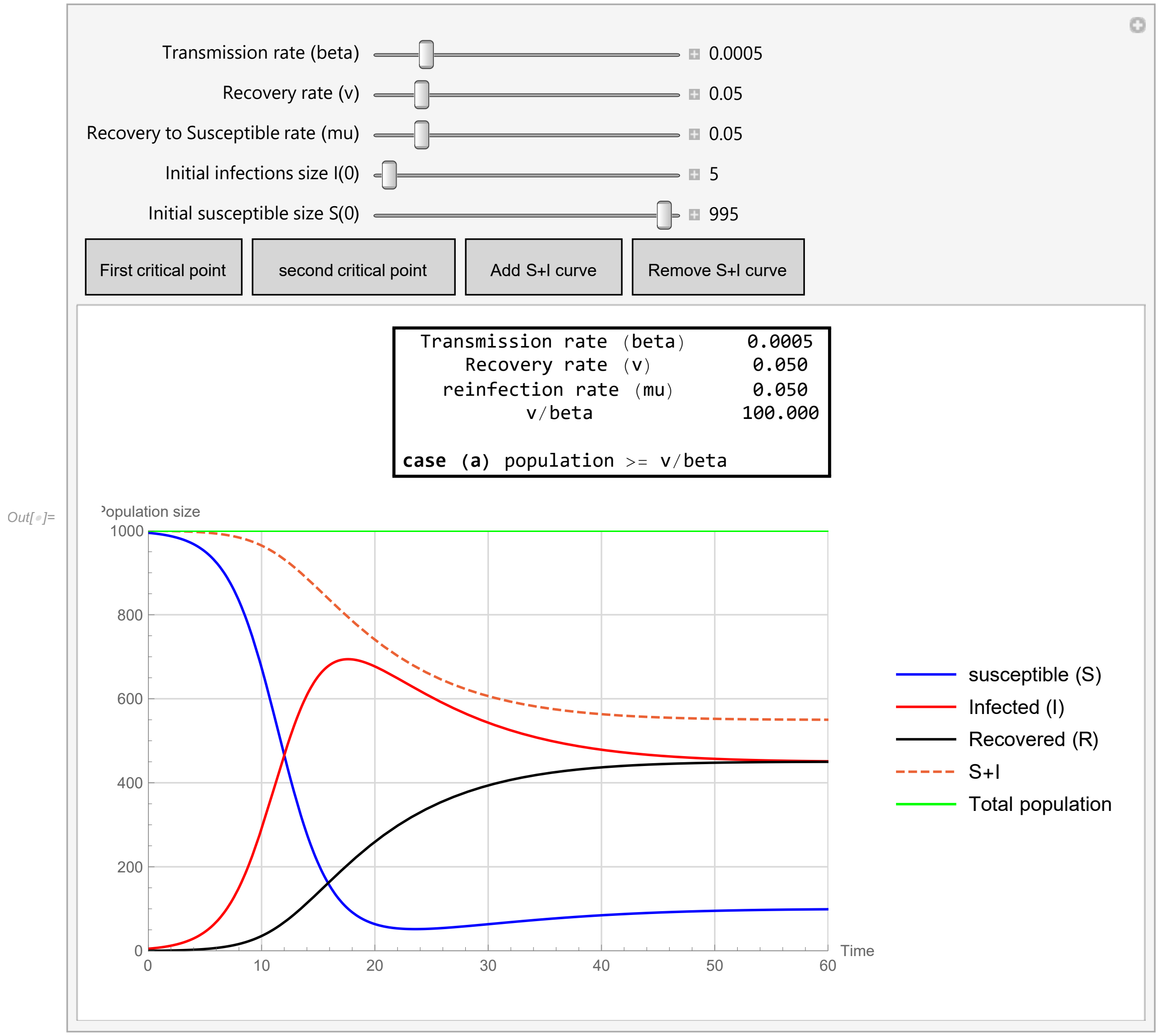

Figure 2: screen shot

The program is written using Mathematica. Here is the notebook

After downloading the notebook and opening it inside Mathematica, you can now use the sliders and analyze the system behaviour for different parameters.

Source code

```
Manipulate[
 tick;

 Module[{ic, eq1, eq2, eq3, S, R, I0, t, initialR, sol},
  If[initialS + initialI > 1000, initialI = 1000 - initialS];
  initialR = 1000 - (initialS + initialI);
```

```
 ic = {S[0] == initialS, I0[0] == initialI, R[0] == initialR};
 eq1 = S'[t] == -beta S[t] *I0[t] + mu*R[t];
 eq2 = I0'[t] == beta*S[t]*I0[t] - v I0[t];
 eq3 = R'[t] == v*I0[t] - mu*R[t];
 sol = NDSolve[{eq1, eq2, eq3, ic}, {S, I0, R}, {t, 0, 60}];
 Grid[{
   {Grid[{
      {"Transmission rate (beta)  ", padIt2[beta, {5, 4}]},
      {"Recovery rate (v) ", padIt2[v, {3, 3}]},
      {"reinfection rate (mu) ", padIt2[mu, {3, 3}]},
      {"v/beta ", padIt2[v/beta, {6, 3}]},
      {},
      {If[1000 >= v/beta,
        Row[{Style["case (a)", Bold], " population >= v/beta"}],
        Row[{Style["case (b)", Bold], " population <= v/beta"}]
        ]}}, Frame -> True
     ]
    },
   {Plot[{
      Evaluate[S[t] /. sol],
      Evaluate[I0[t] /. sol],
      Evaluate[R[t] /. sol],
      If[addSI,
       Evaluate[S[t] /. sol] + Evaluate[I0[t] /. sol], Nothing],
      If[addSI, 1000]

      },
     {t, 0, 60},
     PlotLegends ->
      If[addSI, {"susceptible (S)", "Infected (I)", "Recovered (R)",
        "S+I", "Total population"}, {"susceptible (S)",
        "Infected (I)", "Recovered (R)"}],
     PlotStyle ->
      If[addSI, {Blue, Red, Black, Dashed, Green}, {Blue, Red,
        Black}],
     GridLines -> Automatic, GridLinesStyle -> LightGray,
     AxesLabel -> {"Time (days)", "Population size"},
     ImageSize -> 500,
     ImagePadding -> 30,
     PlotRange -> {{0, 60}, {0, 1000}}
     ]
    }}]
 ],
{{beta, 0.0005, "Transmission rate (beta)"}, 0.0001, 0.003, 0.0001,
```

```
 Appearance -> "Labeled"},
{{v, 0.05, "Recovery rate (v)"}, 0.001, 0.4, 0.001,
 Appearance -> "Labeled"},
{{mu, 0.05, "Recovery to Susceptible rate (mu)"}, 0.001, 0.4, 0.001,
 Appearance -> "Labeled"},
{{initialI, 5, "Initial infections size I(0)"}, 0, 1000, 1,
 Appearance -> "Labeled"},
{{initialS, 995, "Initial susceptible size S(0)"}, 0, 1000, 1,
 Appearance -> "Labeled"},
Grid[{{
   Button[
    Text[Style["First critical point", 11]], {initialI = 0,
     initialS = 1000, tick = Not[tick]}, ImageSize -> {100, 35}],
   Button[Text[Style["second critical point", 11]],
    tick = Not[tick];
    If[1000 >= v/beta,
     {initialI = mu/(v + mu)*(1000 - v/beta), initialS = v/beta}
     , Nothing
     ], ImageSize -> {130, 35}],
   Button[
    Text[Style["Add S+I curve", 11]], {addSI = True,
     tick = Not[tick]}, ImageSize -> {100, 35}],
   Button[
    Text[Style["Remove S+I curve", 11]], {addSI = False,
     tick = Not[tick]}, ImageSize -> {110, 35}]

   }
  }
 ],
{{tick, False}, None},
{{addSI, True}, None},
TrackedSymbols :> {beta, v, mu, initialI, initialS, tick},
SaveDefinitions -> False,
Initialization :> (padIt2[v_, f_List] :=
   AccountingForm[v, f, NumberSigns -> {"", ""},
    NumberPadding -> {"0", "0"}, SignPadding -> True])
]
```

5 SIR infection model

The SIR model is

$$\dot{S} = -\beta S I \tag{1}$$
$$\dot{I} = \beta S I - \upsilon I \tag{2}$$
$$\dot{R} = \upsilon I \tag{3}$$

Where S represent the susceptible population and I the infected population and R the recovered population, and β is the transmission rate from S to I and υ is the recovery rate from I to R.

This animation shows what happens as the transmission rate increases (while the recovery rate is kept fixed). This uses initial conditions of $S(0) = 1000$ and $I(0) = 1$. Which means one person is infected only initially.

The recovery rate is kept at 0.05 while the transmission rate is changed from 0.0003 to 0.03.

This animation shows what happens as the transmission rate is kept fixed at 0.005, while the recovery rate is increased from 0.001 to 0.3

The above program is avaliable as Mathematica notebook below. Here is screen shot

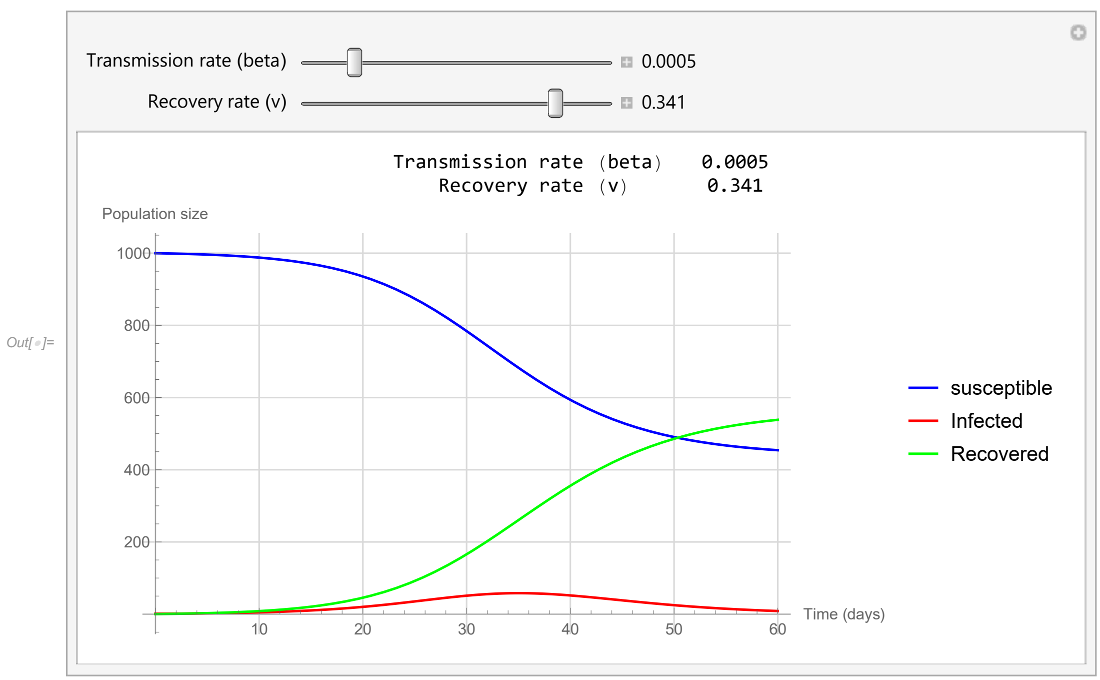

Figure 3: Program written for SIR model analysis

Here is the notebook

Source code

```
Manipulate[
 Module[{ic, eq1, eq2, eq3, S, R, I0, t, sol},
  ic = {S[0] == 1000, I0[0] == 1, R[0] == 0};
  eq1 = S'[t] == -beta S[t]*I0[t];
  eq2 = I0'[t] == beta*S[t] I0[t] - v I0[t];
  eq3 = R'[t] == v*I0[t];
  sol = NDSolve[{eq1, eq2, eq3, ic}, {S, I0, R}, {t, 0, 60}];
  Grid[{{Grid[{{"Transmission rate (beta)  ",
        NumberForm[beta, {4, 4}]}, {"Recovery rate (v) ",
        NumberForm[v, {3, 3}]}}]}},
    {Plot[{
       Evaluate[S[t] /. sol],
       Evaluate[I0[t] /. sol],
       Evaluate[R[t] /. sol]
       },
      {t, 0, 60},
      PlotLegends -> {"susceptible", "Infected", "Recovered"},
      PlotStyle -> {Blue, Red, Green}, GridLines -> Automatic,
      GridLinesStyle -> LightGray,
      AxesLabel -> {"Time (days)", "Population size"},
      ImageSize -> 500]
     }}]
  ],
 {{beta, 0.0005, "Transmission rate (beta)"}, 0.0001, 0.003, 0.0001,
  Appearance -> "Labeled"},
 {{v, 0.05, "Recovery rate (v)"}, 0.001, 0.4, 0.001,
  Appearance -> "Labeled"},
 TrackedSymbols :> {beta, v}
 ]
```

6 Van der Pol system

These are animations of $\ddot{x} + x = \mu(1 - x^2)\dot{x}$ for different μ and different initial conditions showing the limit cycle.

7 Limit cycles

8 Example 1 (Stable limit cycle)

The following system of ode's gives solutions that have limit cycle.

$$
\begin{aligned}
x'(t) &= -by + ax(1 - x^2 - y^2) \\
y'(t) &= ax + by(1 - x^2 - y^2)
\end{aligned}
$$

When a and b is not 1, then the solution shows more iterations until it reaches the limit cycle, which is a circle of radius 1.

The following animations are for different initial conditions (one inside the limit cycle and one outside) and for different values of a, b.

As a becomes closer to zero, it takes many more cycles to appraoch the limit cycle.

9 Example 2. Stable and Unstable limit cycles

These are animations of the system $\dot{r} = r(1 - r^2), \dot{\theta} = 1$ which exhibits a stable limit cycle of radius 1.

And animation of the system $\dot{r} = -r(1 - r^2), \dot{\theta} = 1$ where the limit cycle now becomes unstable.

10 Example 3. Stable and Unstable limit cycles

These are animations of the system $\dot{r} = r(r^2 - 3r + 1), \dot{\theta} = 1$ which exhibits a stable limit cycle of radius 1.

11 $x''(t) + f(x)x'(t) + x(t) = 0$ and periodic solutions

This shows dependence of having perodic solutions for $x''(t) + f(x)x'(t) + x(t) = 0$ where $f(x) = x^2 + x + a$ on the value of a. This simulations shows the solution and the phase space as a changes from -2 to 1. A specific orbit with I.C. $x(0) = 1, x'(0) = 0$ is highlighted in red.

The program is written using Mathematica. Here is the notebook

Source code

```
(*version 2:21 AM, 4/16/2020 *)
Manipulate[
 Module[{f1, f2, x1, x2, sol, x, t, ode},
  f1 = x2;
  f2 = (-x1^2 - x1 - a)*x2 - x1;
  ode = x''[t] + (x[t]^2 + x[t] + a)*x'[t] + x[t] == 0;
  sol = NDSolve[{ode, x[0] == 1, x'[0] == 0}, x, {t, 0, 30}];
  Grid[{
    {Row[{"a = ", NumberForm[a, {4, 2}]}]},
    {
     StreamPlot[{f1, f2}, {x1, -5, 5}, {x2, -7, 7},
      ImageSize -> 300,
      StreamPoints -> {{{{1, 0}, Red}, Automatic}}],
     Plot[Evaluate[x[t] /. sol], {t, 0, 30},
      PlotRange -> {Automatic, {-4, 4}},
      ImageSize -> 300,
      GridLines -> Automatic, GridLinesStyle -> LightGray,
      PlotStyle -> Red,
      AxesLabel -> {"time", "x(t)"},
      BaseStyle -> 12,
      PlotLabel -> "Solution to  x''+f[x] x' + x = 0"
      ]
     }
    }
   ]
  ],
 {{a, -2, "a"}, -2, 1, 0.1, Appearance -> "Labeled"},
 TrackedSymbols :> {a}
 ]
```

12 Fixed point iteration $f(x) = x - x^3$ a sink at $x = 0$

This shows fixed point iteration of $f(x) = x - x^3$ with 2 seeds, one using $x_0 = 0.6$ and one using $x_0 = -0.6$ in the first plot (top left corner plot).

It shows the fixed point interation is stable and converges to the limit $x = 0$ from both sides. Hence $x = 0$ is a sink.

Additional animations shown in the table below, are zoom versions of the same iterations that used the seed $x = 0.6$ in order to obtain better views.

The program is written using Mathematica. Here is the notebook

Source code

```
(*version 2:21 AM, 4/16/2020 *)
Manipulate[
 Module[{f1, f2, x1, x2, sol, x, t, ode},
  f1 = x2;
  f2 = (-x1^2 - x1 - a)*x2 - x1;
  ode = x''[t] + (x[t]^2 + x[t] + a)*x'[t] + x[t] == 0;
  sol = NDSolve[{ode, x[0] == 1, x'[0] == 0}, x, {t, 0, 30}];
  Grid[{
    {Row[{"a = ", NumberForm[a, {4, 2}]}]},
    {
     StreamPlot[{f1, f2}, {x1, -5, 5}, {x2, -7, 7},
      ImageSize -> 300,
      StreamPoints -> {{{{1, 0}, Red}, Automatic}}],
     Plot[Evaluate[x[t] /. sol], {t, 0, 30},
      PlotRange -> {Automatic, {-4, 4}},
      ImageSize -> 300,
      GridLines -> Automatic, GridLinesStyle -> LightGray,
      PlotStyle -> Red,
      AxesLabel -> {"time", "x(t)"},
      BaseStyle -> 12,
      PlotLabel -> "Solution to  x''+f[x] x' + x = 0"
      ]
     }
    }
   ]
  ],
 {{a, -2, "a"}, -2, 1, 0.1, Appearance -> "Labeled"},
 TrackedSymbols :> {a}
 ]
```