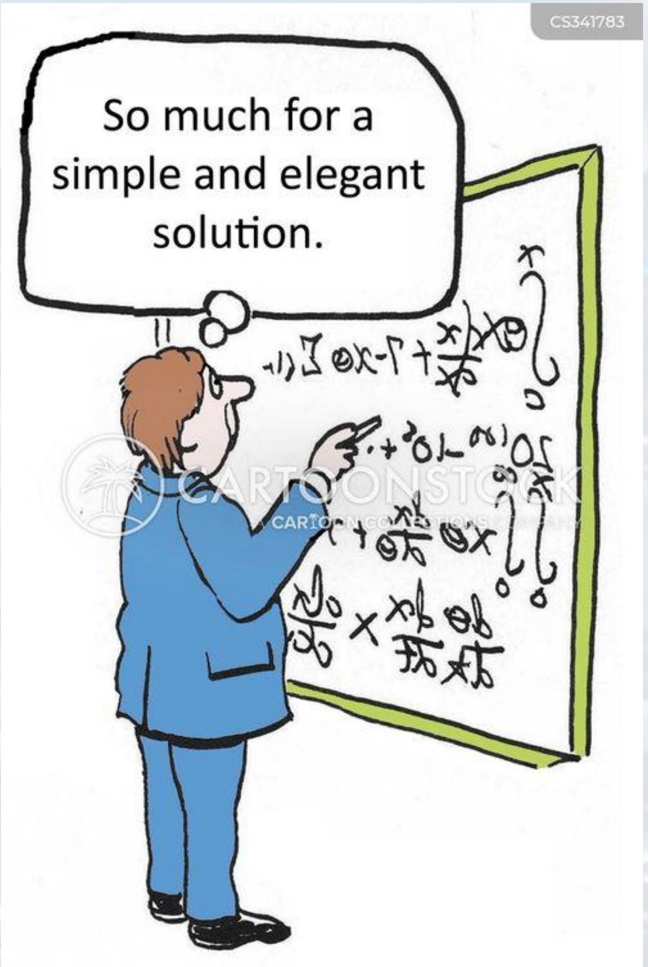
Berkeley
UNIVERSITY OF CALIFORNIA

M. Hohle:

Physics 77: Introduction to Computational Techniques in Physics



Introduction to Computational Techniques in Physics: syllabus

Week	Date	Topic
1	June 12th	Programming Environment & UIs for Python, Programming Fundamentals
2	*June 19th*	Basic Types in Python
3	June 26th	Parsing, Data Processing and File I/O, Visualization
4	July 3rd	Functions, Map & Lambda
5	July 10th	Random Numbers & Probability Distributions, Interpreting Measurements
6	July 17th	Numerical Integration and Differentiation
7	July 24th	Root finding, Interpolation
8	**July 31st**	Systems of Linear Equations, **Ordinary Differential Equations (ODEs)**
9	Aug 7th	Stability of ODEs, Examples
10	Aug 14th	Final Project Presentations

ordinary differential equation:

- **total** derivative → *ordinary* $\frac{d^k f(x)}{dx^k}$ $k \in \mathbb{N}$

- of **n-th** order → *n = max(k)*

- **non-linear** → *power of **any** x is not one*

partial differential equation:

- at least one **partial** derivative

$$\frac{\partial y(x)}{\partial t} = [a\Delta + bg(x)]\, y(x)$$ diffusion

$$\frac{\partial^2 y(x)}{\partial t^2} = [a\Delta + bg(x)]\, y(x)$$ wave

let's start simple:

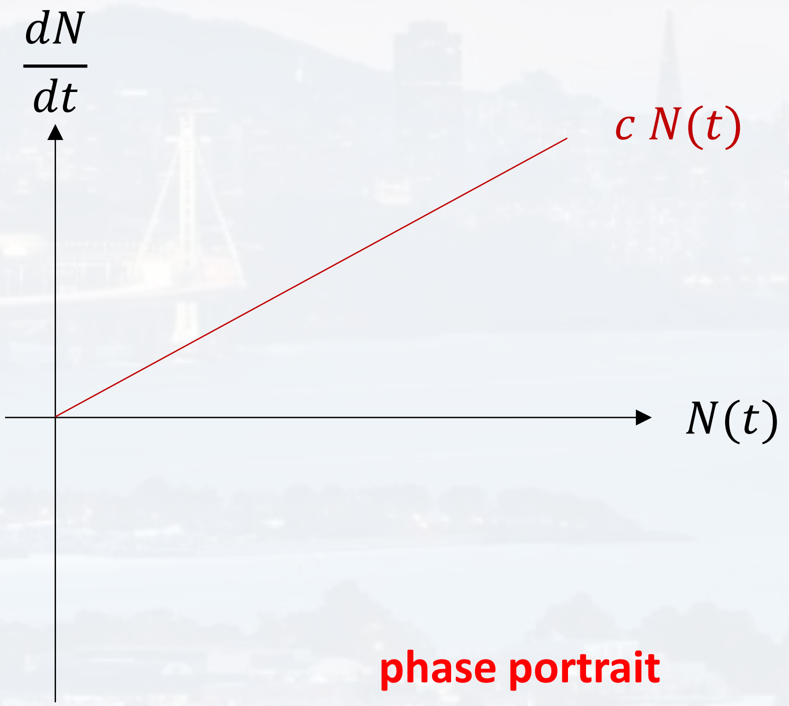
constant relative change per time step

$$\frac{\Delta N}{N}\frac{1}{\Delta t} = c$$

$$\frac{dN}{N} = c\,dt \qquad \frac{dN}{dt} = c\,N$$

$$\int_{N(t=0)}^{N} \frac{1}{N}\,dN = c\int_{0}^{\tau} dt$$

$$N(t) = N(t=0)\,e^{ct}$$



phase portrait

let's start simple:

We want the model to have a limited carrying capacity κ

$$\frac{dN}{N} = c\,dt$$

For $t \rightarrow \infty$ the island can only feed $N_{eq} = \kappa$ cows

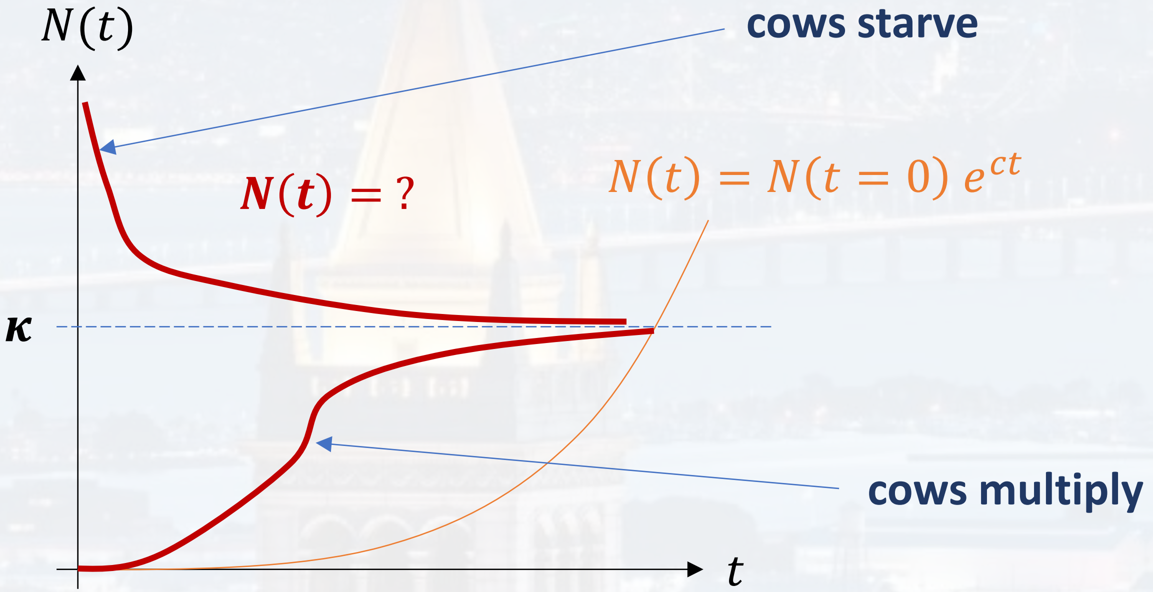
κ: carrying capacity

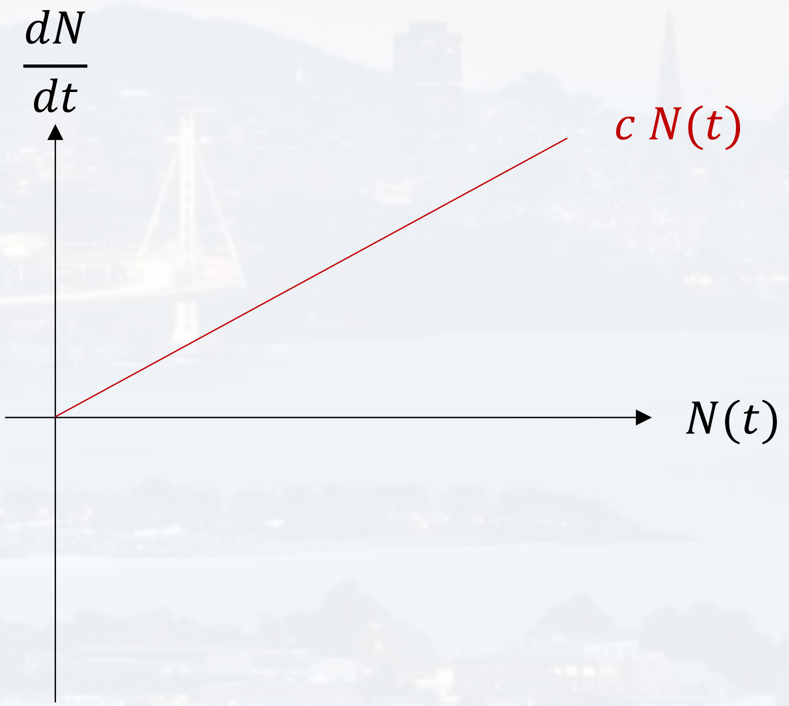
How do we model that?

let's start simple:

We want the model to have a limited **carrying capacity κ**

$$\frac{dN}{N} = c\,dt$$

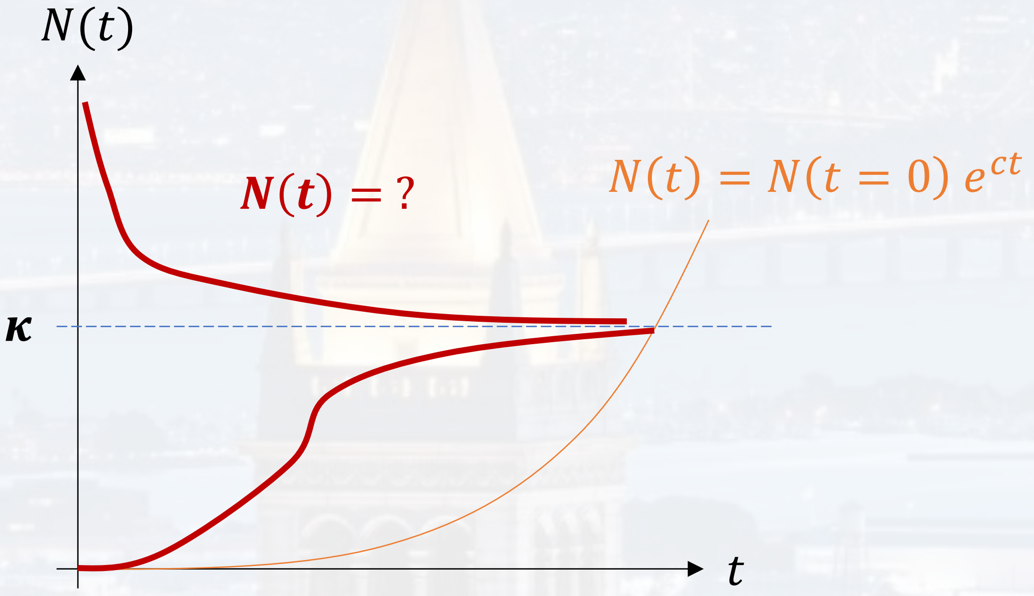


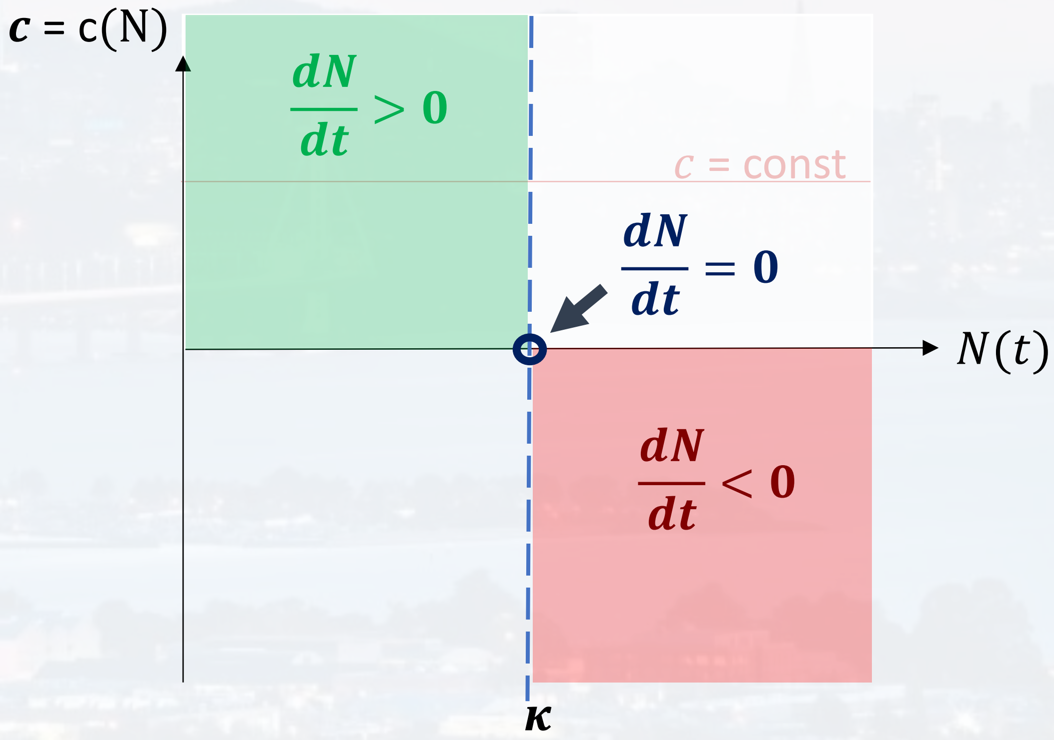


<u>let's start simple:</u>

We want the model to have a limited carrying capacity $\boldsymbol{\kappa}$

$$\frac{dN}{N} = c\,dt$$

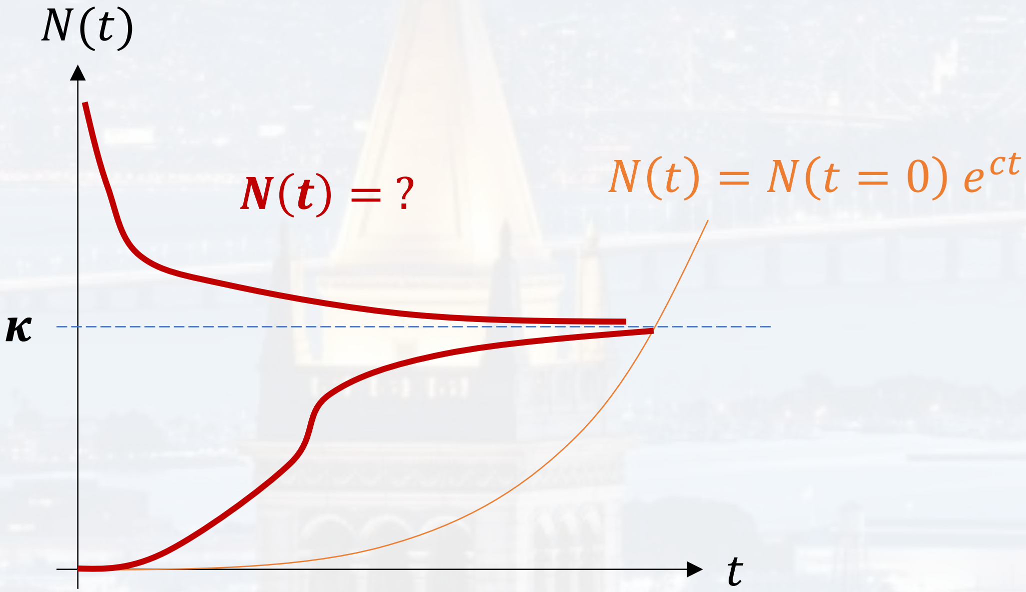


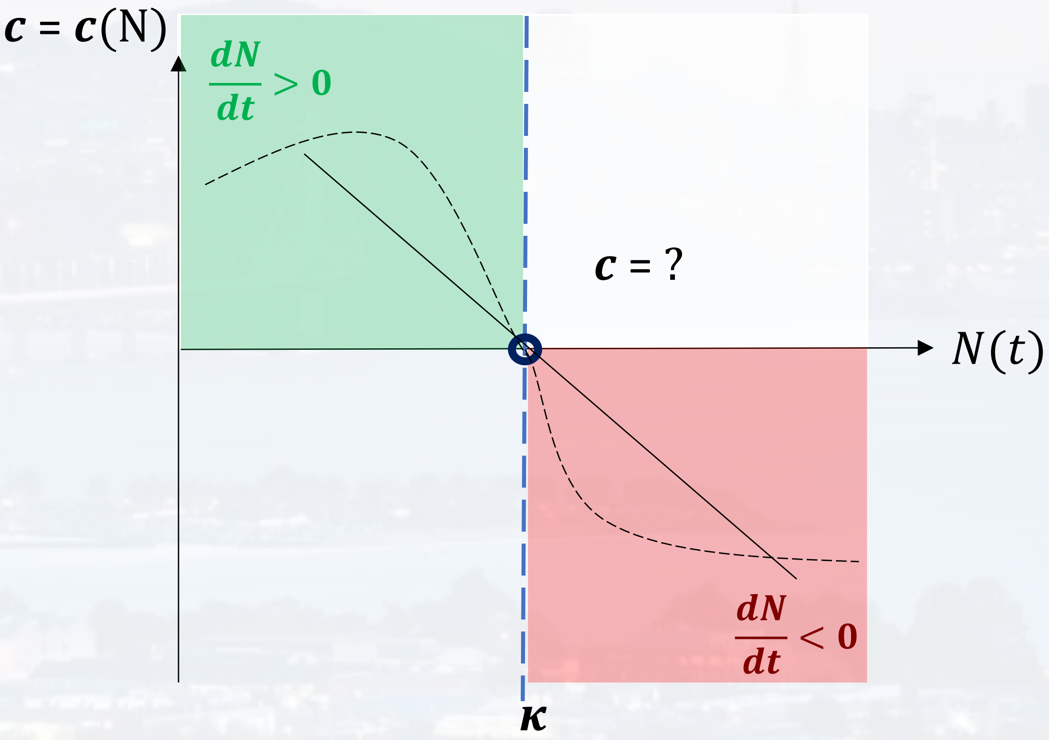


let's start simple:

We want the model to have a limited carrying capacity κ

$$\frac{dN}{N} = c\,dt$$





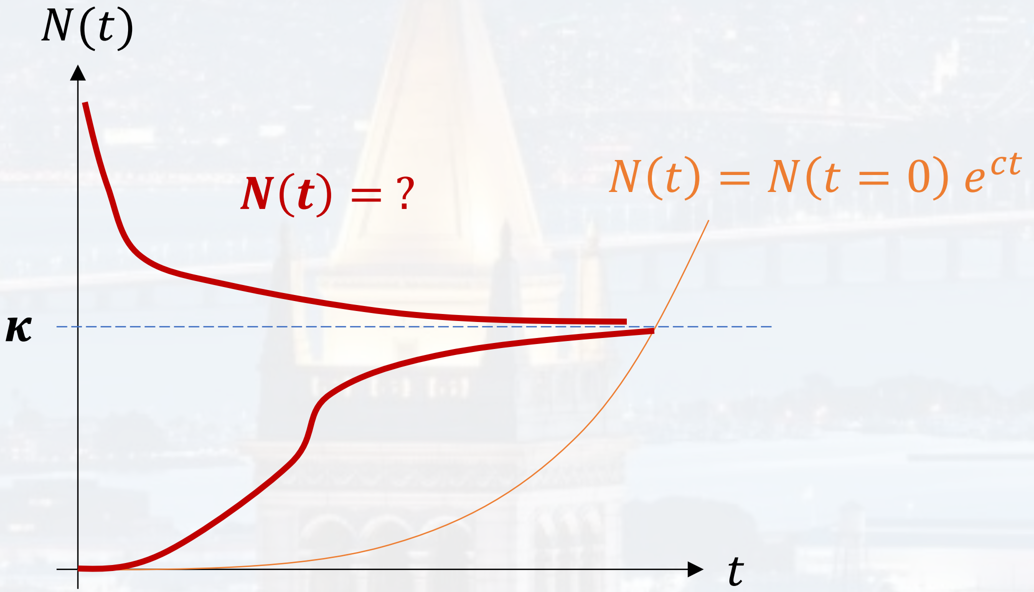
let's start simple:

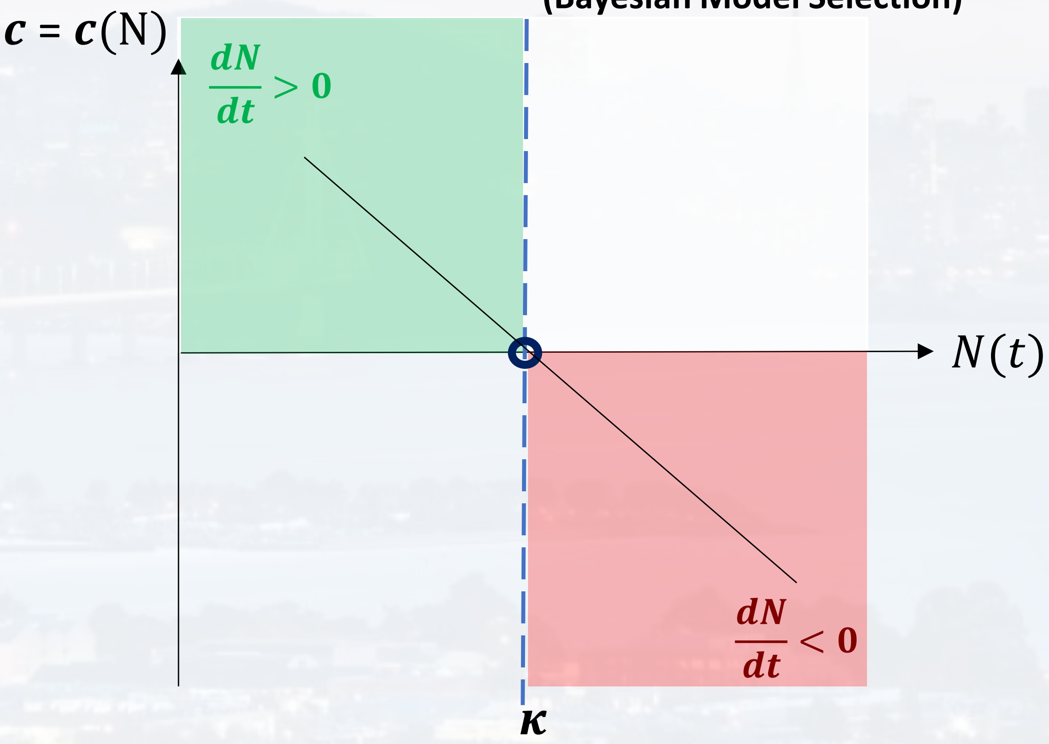
We want the model to have a limited carrying capacity $\boldsymbol{\kappa}$

$$\frac{dN}{N} = c\,dt$$

Occam's razor:

prefer simple model
(Bayesian Model Selection)





let's start simple:

We want the model to have a limited carrying capacity $\boldsymbol{\kappa}$

$$\frac{dN}{N} = c\,dt$$

$$c(N) = c_0 + m\,N$$

$$c(\kappa) = 0 \qquad c(\kappa) = 0 = c_0 + m\kappa \qquad m = -\frac{c_0}{\kappa}$$

$$c(0) = c_0$$

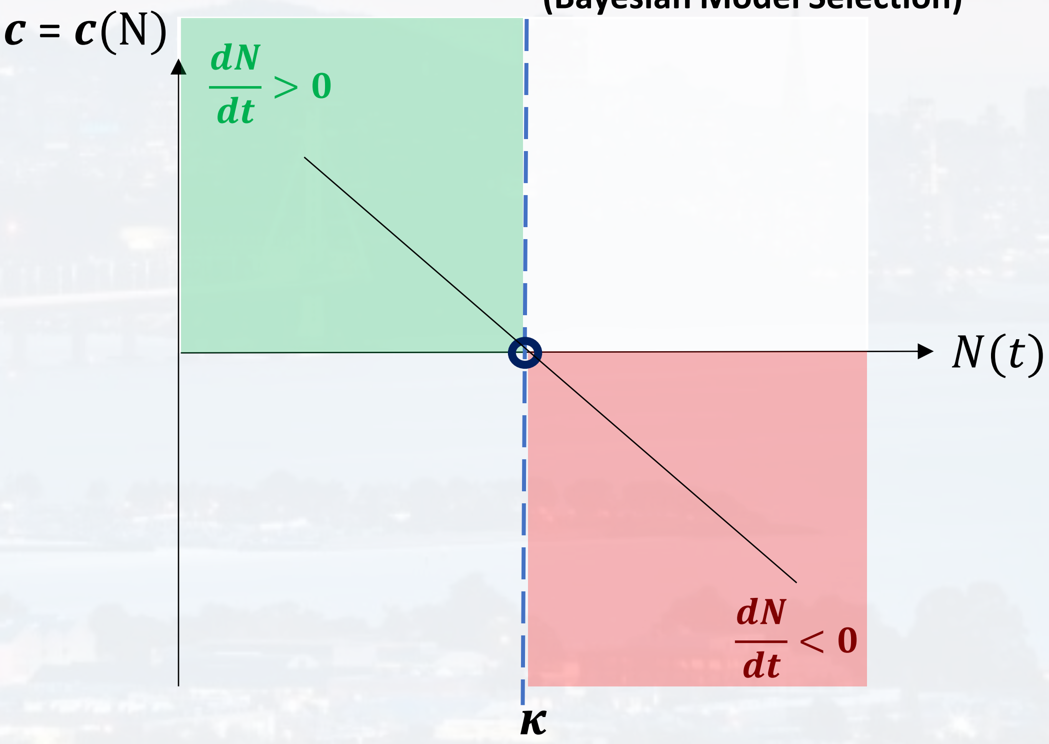
$$c(N) = c_0\left(1 - \frac{1}{\kappa}N\right)$$

$$\boxed{\frac{dN}{N} = c_0\left(1 - \frac{1}{\kappa}N\right)dt}$$

Verhulst Equation

Occam's razor:

prefer simple model
(Bayesian Model Selection)



let's start simple:

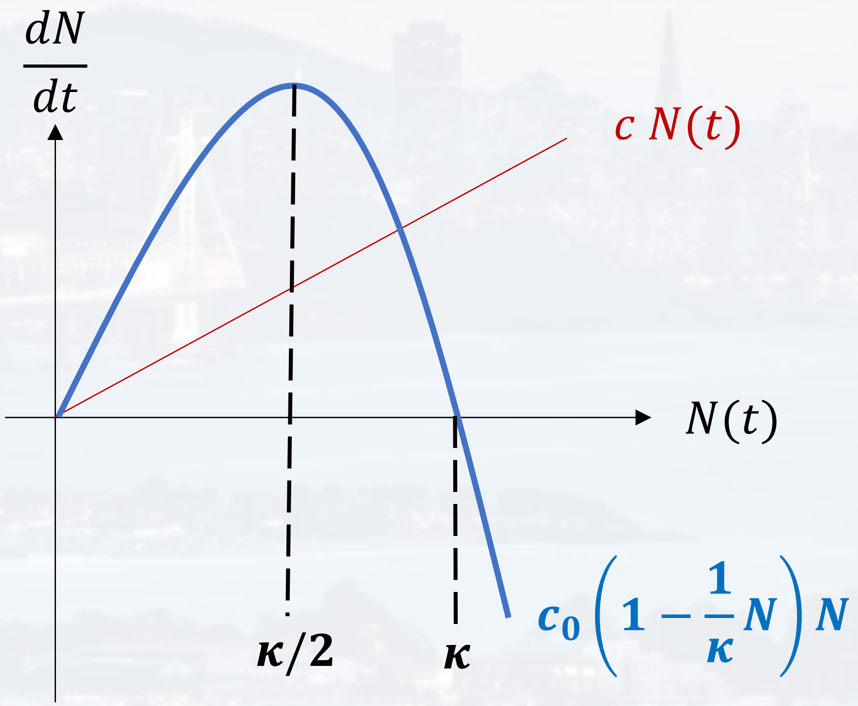
We want the model to have a limited carrying capacity $\boldsymbol{\kappa}$

$$\frac{dN}{N} = c\,dt$$

$$\boxed{\frac{dN}{N} = c_0\left(1 - \frac{1}{\kappa}N\right)dt}$$

Verhulst Equation

$$\frac{dN}{dt} = c_0\left(1 - \frac{1}{\kappa}N\right)N$$

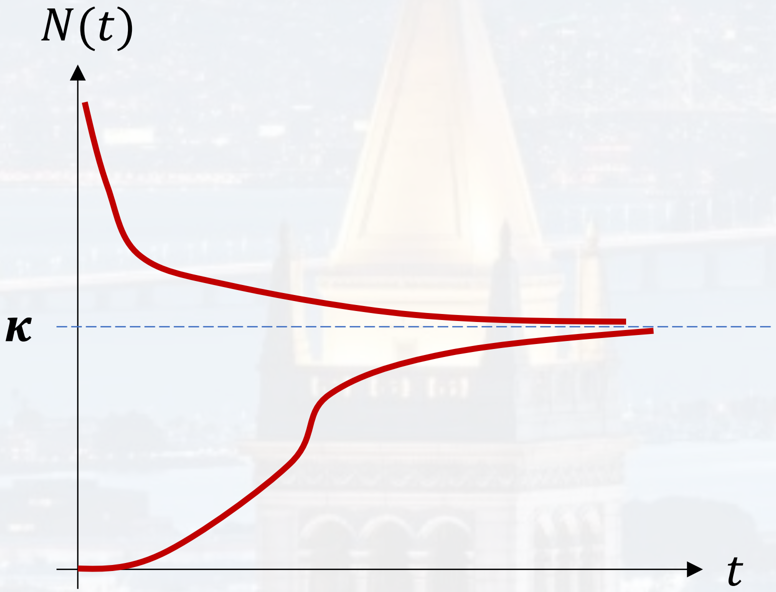


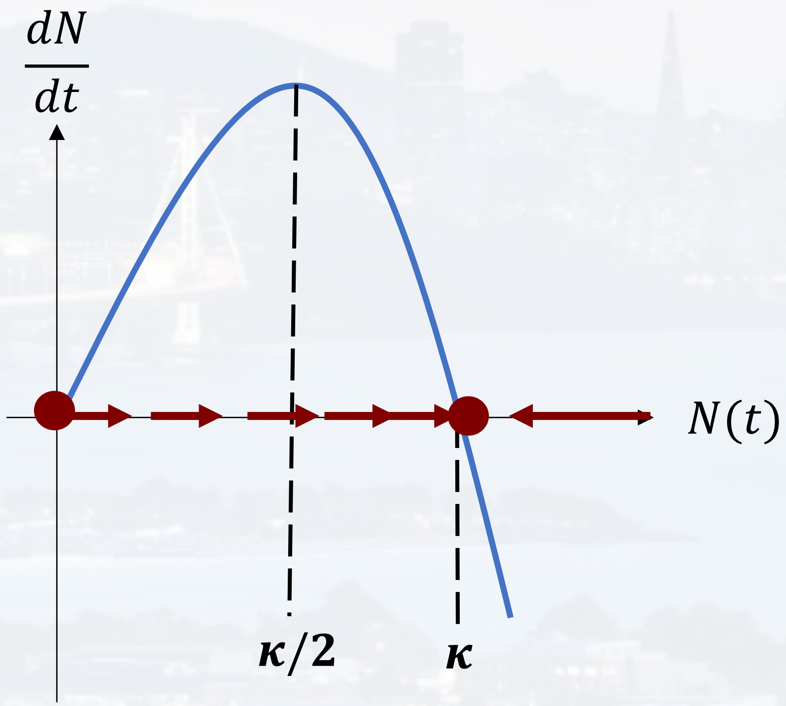
let's start simple:

We want the model to have a limited carrying capacity κ

$$\frac{dN}{dt} = c_0\left(1 - \frac{1}{\kappa}N\right)N$$

Verhulst Equation



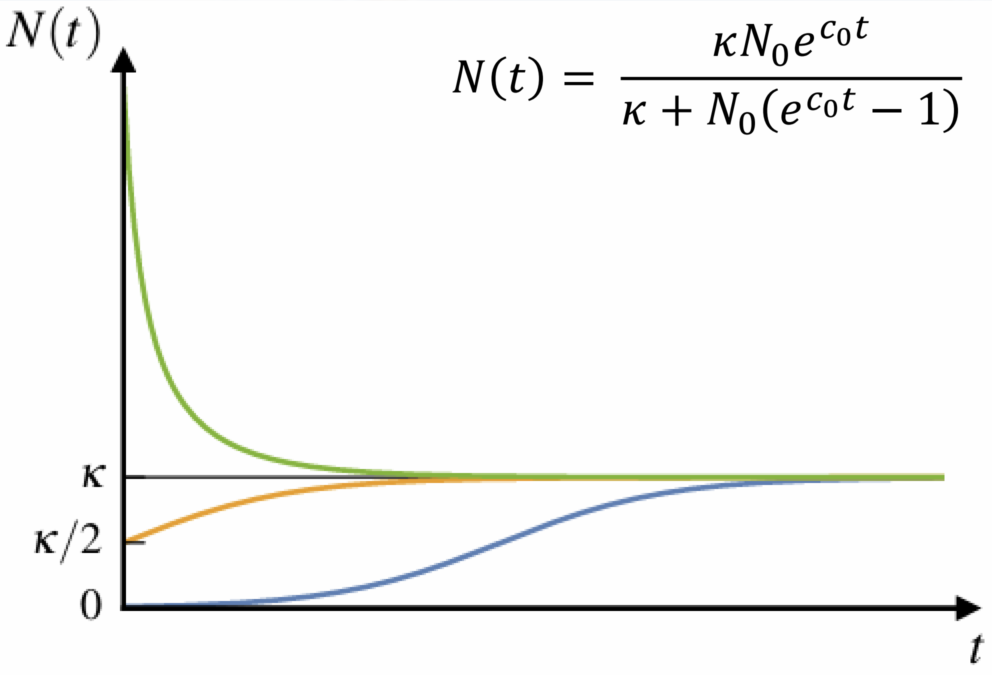


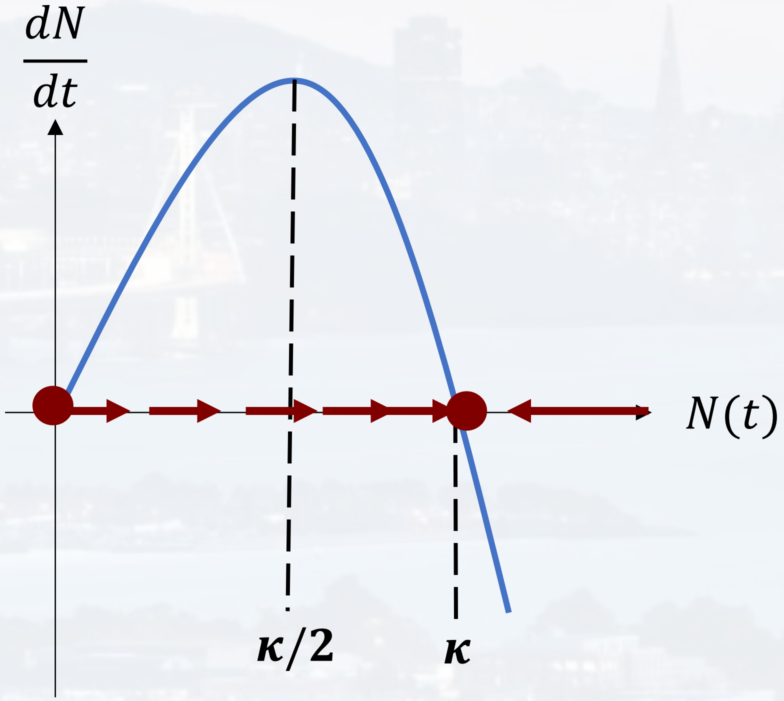
let's start simple:

We want the model to have a limited carrying capacity $\boldsymbol{\kappa}$

$$\frac{dN}{dt} = c_0\left(1 - \frac{1}{\kappa}N\right)N$$

Verhulst Equation





let's start simple:

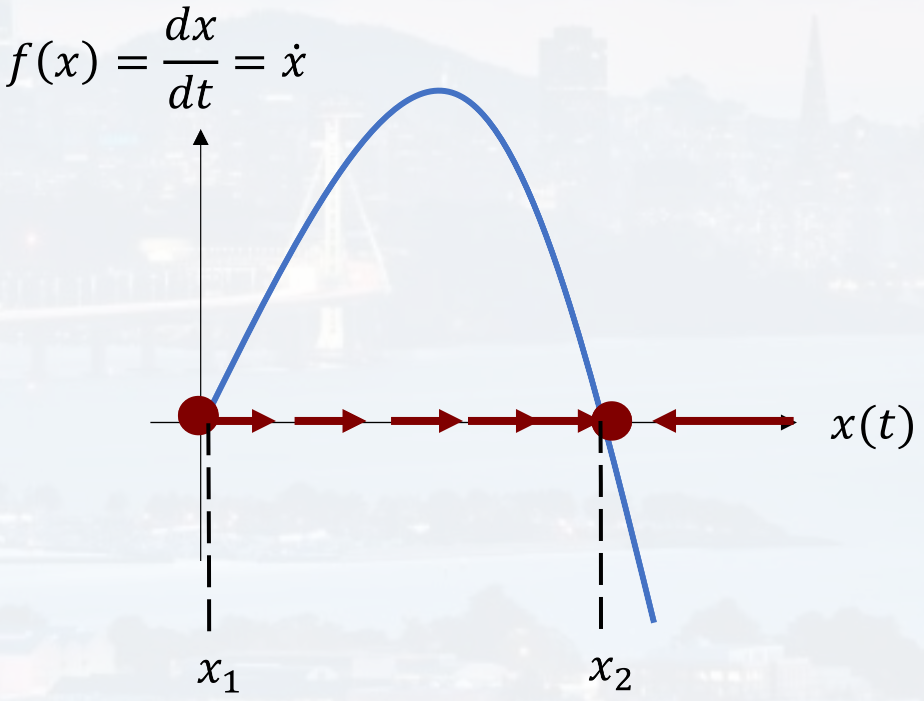
We want the model to have a limited carrying capacity $\boldsymbol{\kappa}$

$$\frac{dN}{dt} = c_0\left(1 - \frac{1}{\kappa}N\right)N$$

Verhulst Equation

x_1, x_2 fixed points

$$f(x) = \frac{dx}{dt} = \dot{x}$$



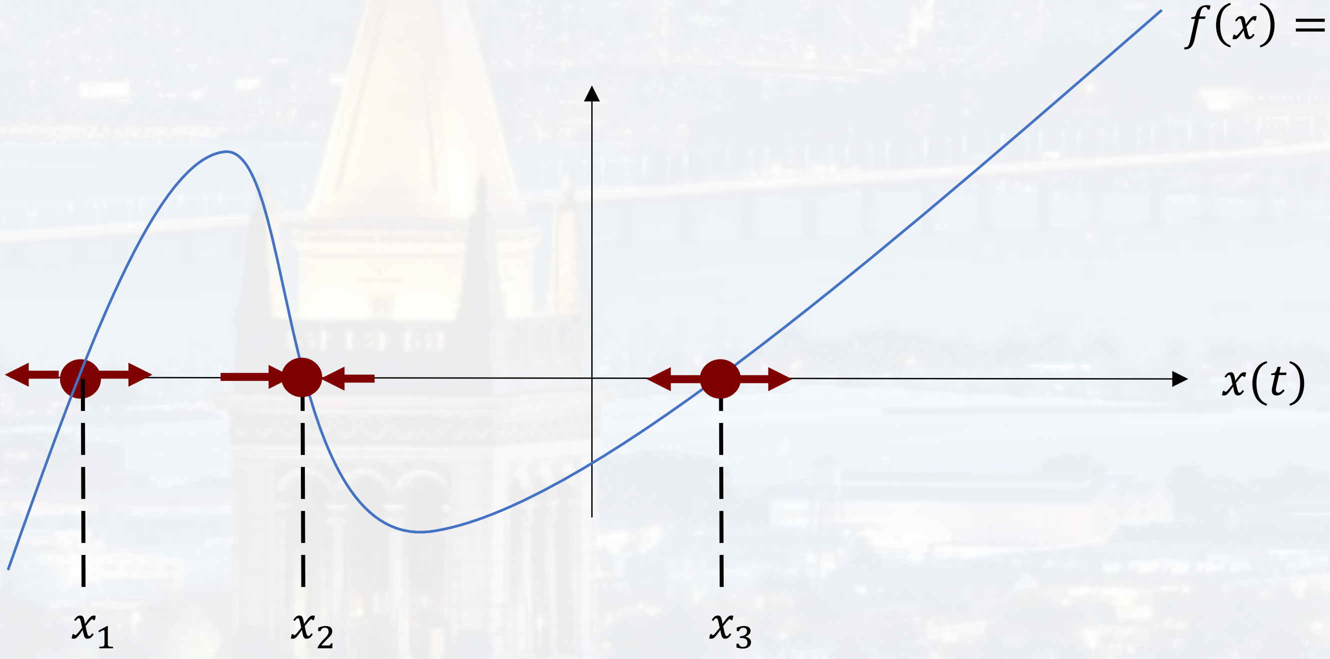
fixed points x^*

x_1: repeller → unstable
x_3: repeller → unstable

$$\frac{df(x)}{dx} = \frac{d}{dx}\dot{x} > 0$$

x_2: attractor → stable

$$\frac{df(x)}{dx} = \frac{d}{dx}\dot{x} < 0$$



fixed points x^*

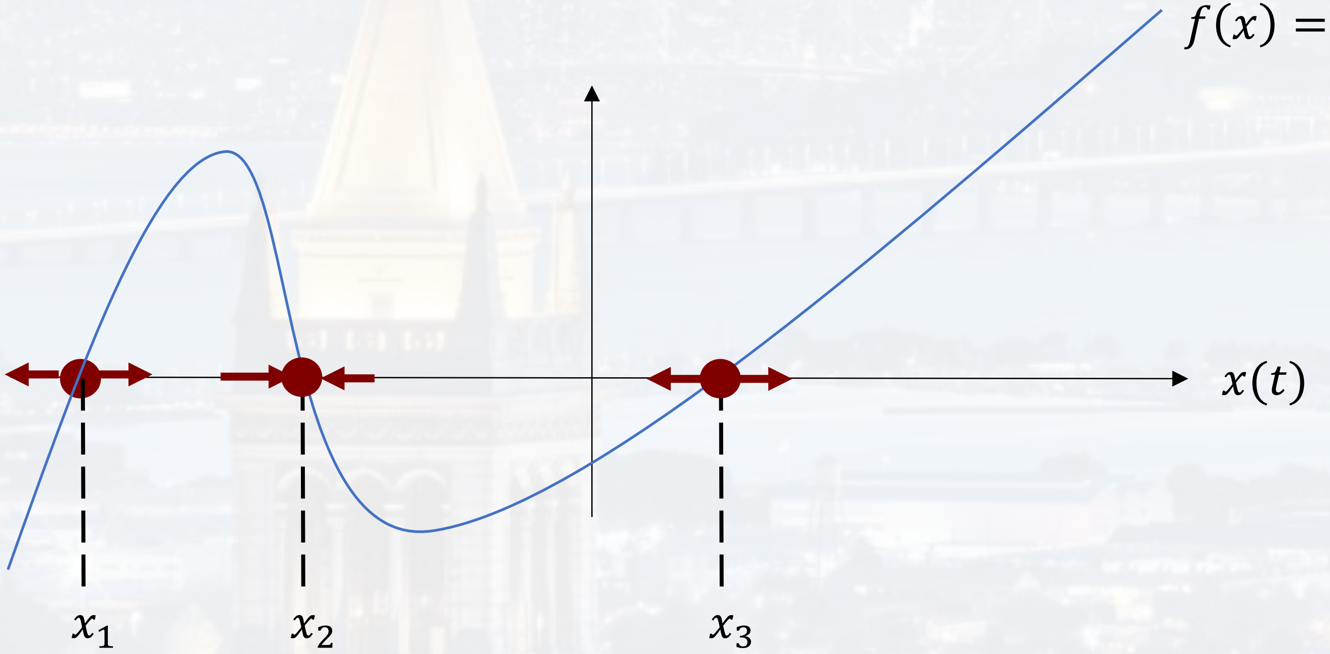
small perturbation $\varepsilon(t)$

$$x(t) = x^* + \varepsilon(t)$$

$$\frac{d\varepsilon(t)}{dt} = \frac{d}{dt}[x(t) - x^*] = f(x) + 0 = f(x^* + \varepsilon(t))$$

$$\boxed{\frac{d\varepsilon(t)}{dt}} = f(x^* + \varepsilon(t)) \approx f(x^*) + \frac{df(x)}{dx}|_{x=x^*}\varepsilon(t) = 0 + \boxed{\frac{df(x)}{dx}|_{x=x^*}\varepsilon(t)}$$

$$\boxed{\varepsilon(t) = \varepsilon_0\, e^{\frac{df(x)}{dx}|_{x=x^*}\, t}}$$



time scale $\tau = \frac{1}{\frac{df(x)}{dx}|_{x=x^*}}$

$\frac{df(x)}{dx} > 0$ unstable

$\frac{df(x)}{dx} < 0$ stable

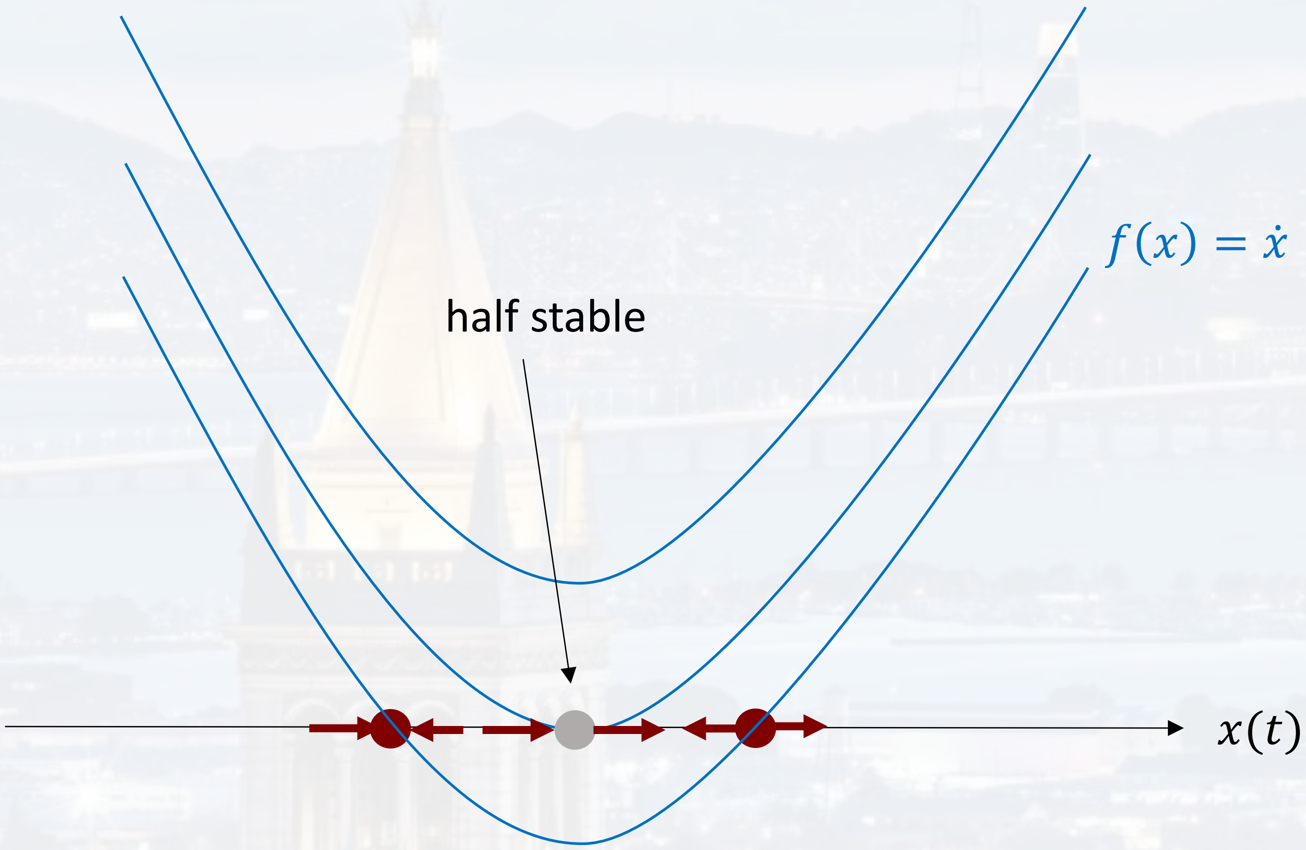
fixed points x^*

small perturbation $\varepsilon(t)$

$$\varepsilon(t) = \varepsilon_0\, e^{\frac{df(x)}{dx}|_{x=x^*}\, t} \qquad \text{time scale } \tau = \frac{1}{\frac{df(x)}{dx}|_{x=x^*}}$$

$$\frac{df(x)}{dx} > 0 \quad \text{unstable}$$

$$\frac{df(x)}{dx} < 0 \quad \text{stable}$$



fixed points x^*

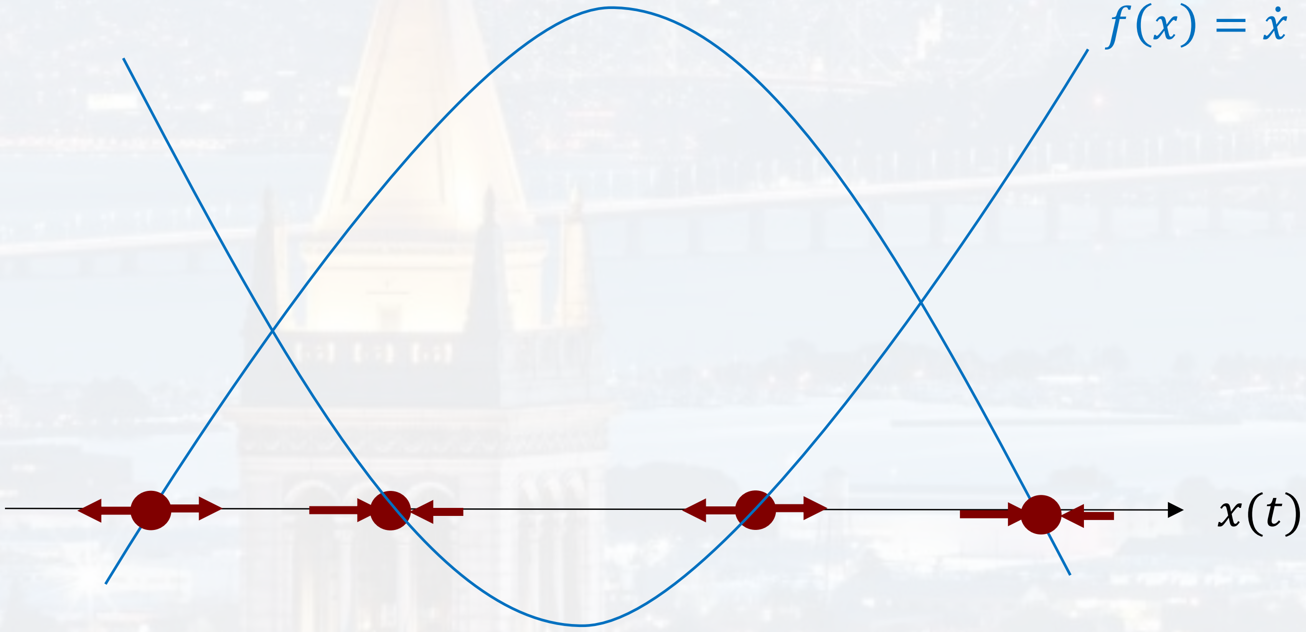
small perturbation $\varepsilon(t)$

$$\varepsilon(t) = \varepsilon_0 \, e^{\frac{df(x)}{dx}|_{x=x^*} t} \qquad \text{time scale } \tau = \frac{1}{\frac{df(x)}{dx}|_{x=x^*}}$$

$$\frac{df(x)}{dx} > 0 \quad \text{unstable}$$

$$f(x) = \boldsymbol{a}x^2 + \boldsymbol{b}x + \boldsymbol{c}$$

$$\frac{df(x)}{dx} < 0 \quad \text{stable}$$



if coupled to another system: → can change dynamics drastically (chem reactions)

2D system

$$f(x,y) = \dot{x} \qquad \dot{x} = -x + a\,y + x^2 y$$

a, b >0

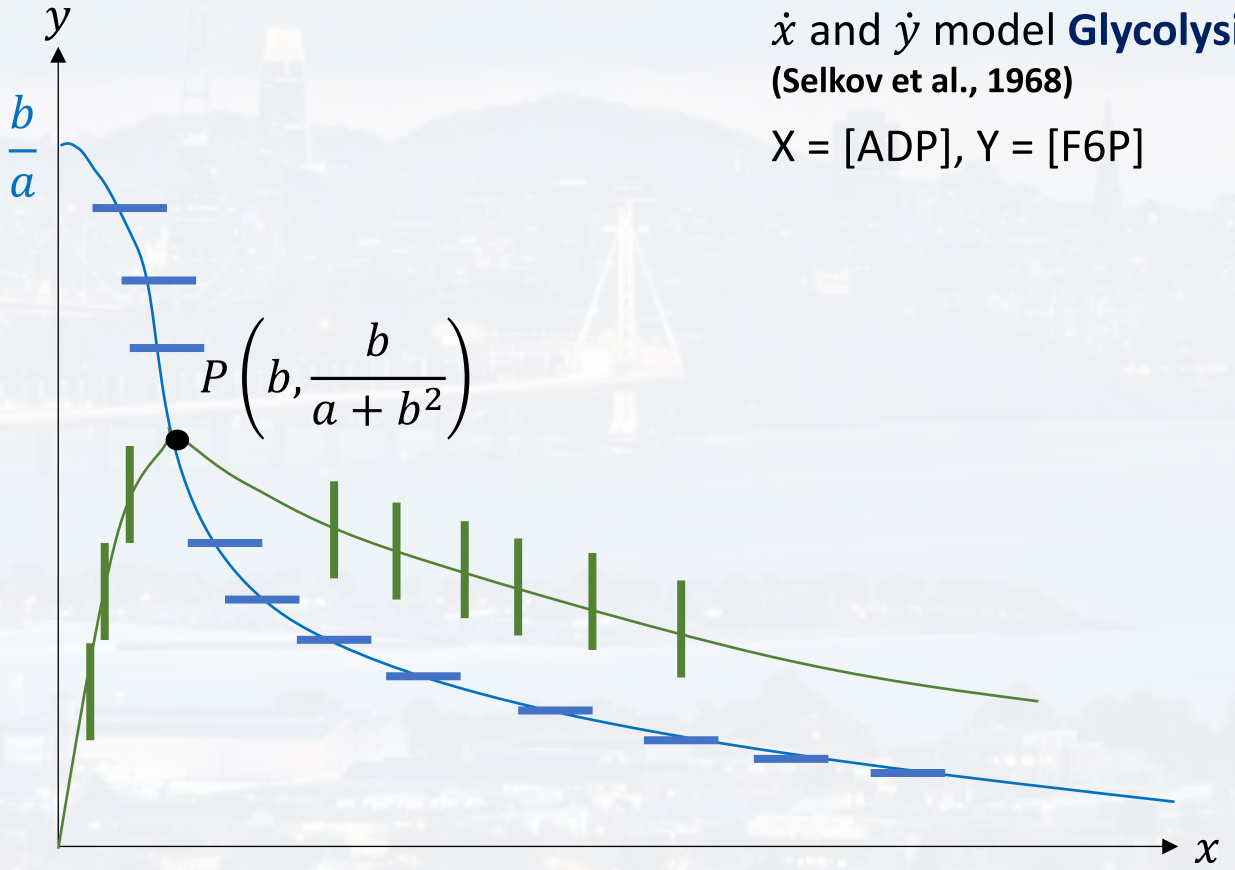
$$g(x,y) = \dot{y} \qquad \dot{y} = b - a\,y - x^2 y$$

non-linear, coupled ODEs

null clines

$\dot{x} = 0 \rightarrow \qquad y_1 = \dfrac{x}{a + x^2}$

$\dot{y} = 0 \rightarrow \qquad y_2 = \dfrac{b}{a + x^2}$



$\dot{x}$ and $\dot{y}$ model **Glycolysis**

(Selkov et al., 1968)

X = [ADP], Y = [F6P]

Find out which way the system moves!

2D system

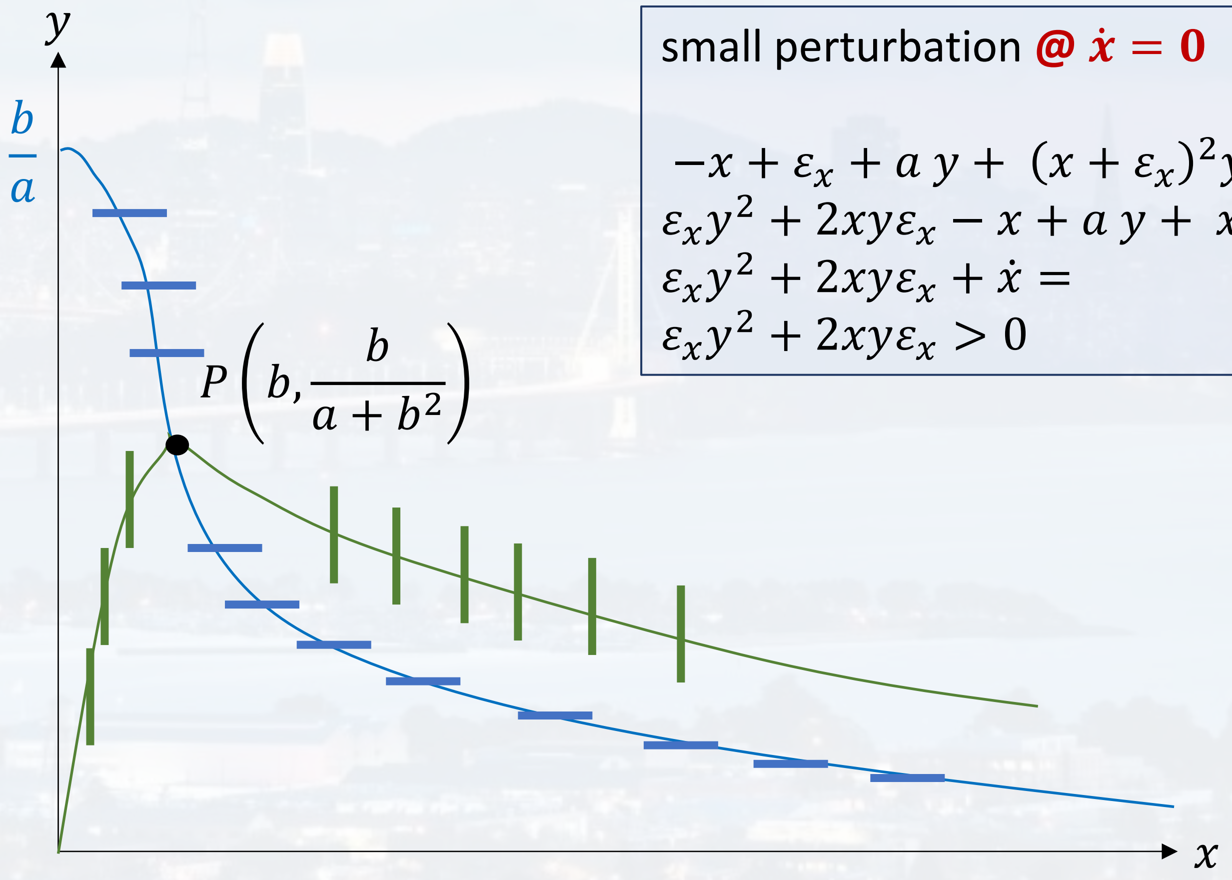
$f(x,y) = \dot{x}$ $\qquad \dot{x} = -x + a\,y + x^2 y$ $\qquad$ a, b >0

$g(x,y) = \dot{y}$ $\qquad \dot{y} = b - a\,y - x^2 y$

null clines

$\dot{x} = 0 \rightarrow \qquad y_1 = \dfrac{x}{a + x^2}$

$\dot{y} = 0 \rightarrow \qquad y_2 = \dfrac{b}{a + x^2}$



small perturbation $@\ \dot{x} = 0$

$$-x + \varepsilon_x + a\,y + (x + \varepsilon_x)^2 y =$$
$$\varepsilon_x y^2 + 2xy\varepsilon_x - x + a\,y + x^2 y =$$
$$\varepsilon_x y^2 + 2xy\varepsilon_x + \dot{x} =$$
$$\varepsilon_x y^2 + 2xy\varepsilon_x > 0$$

Find out which way the system moves!

2D system

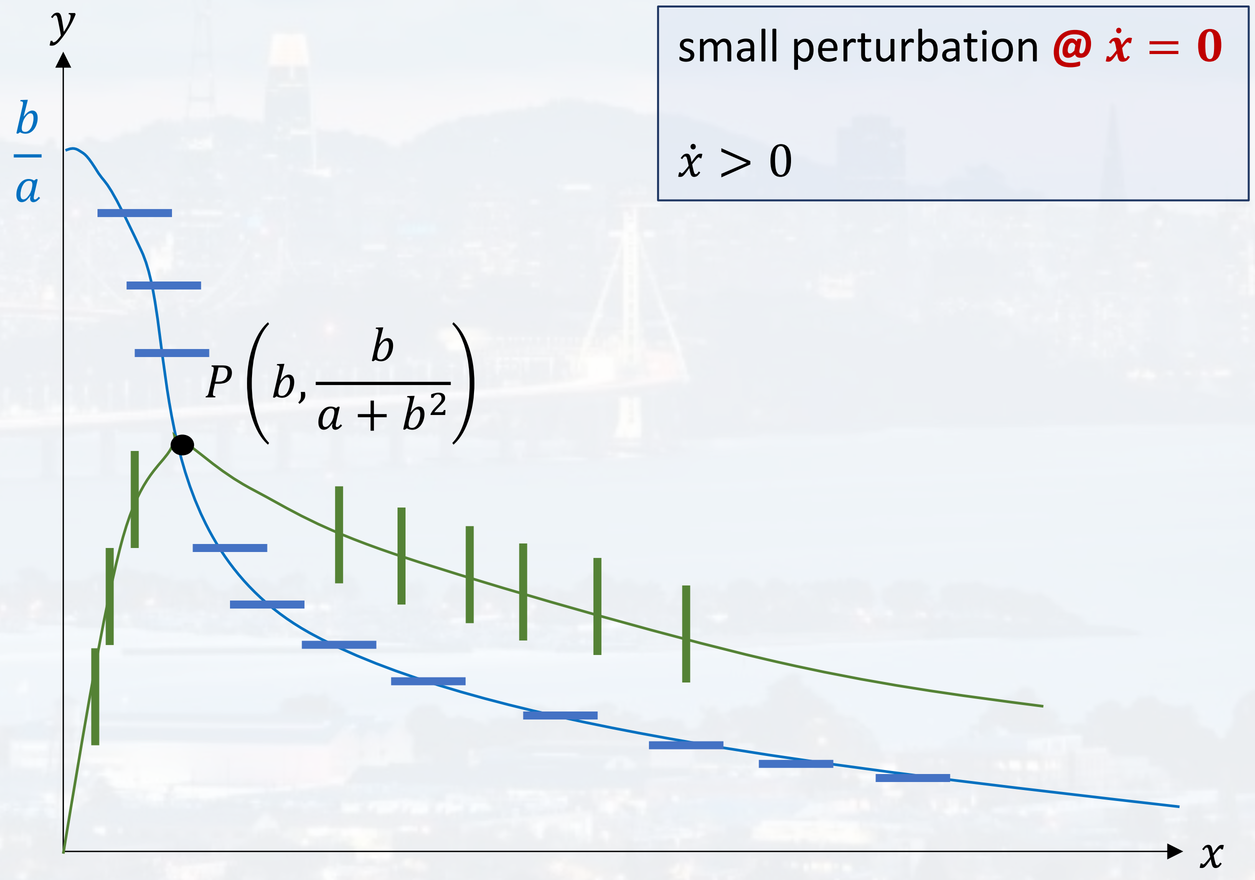
$$f(x,y) = \dot{x} \qquad \dot{x} = -x + a\,y + x^2 y \qquad \text{a, b >0}$$

$$g(x,y) = \dot{y} \qquad \dot{y} = b - a\,y - x^2 y$$

null clines

$\dot{x} = 0 \rightarrow \quad y_1 = \dfrac{x}{a + x^2}$

$\dot{y} = 0 \rightarrow \quad y_2 = \dfrac{b}{a + x^2}$



small perturbation @ $\dot{x} = 0$

$\dot{x} > 0$

Find out which way the system moves!

2D system

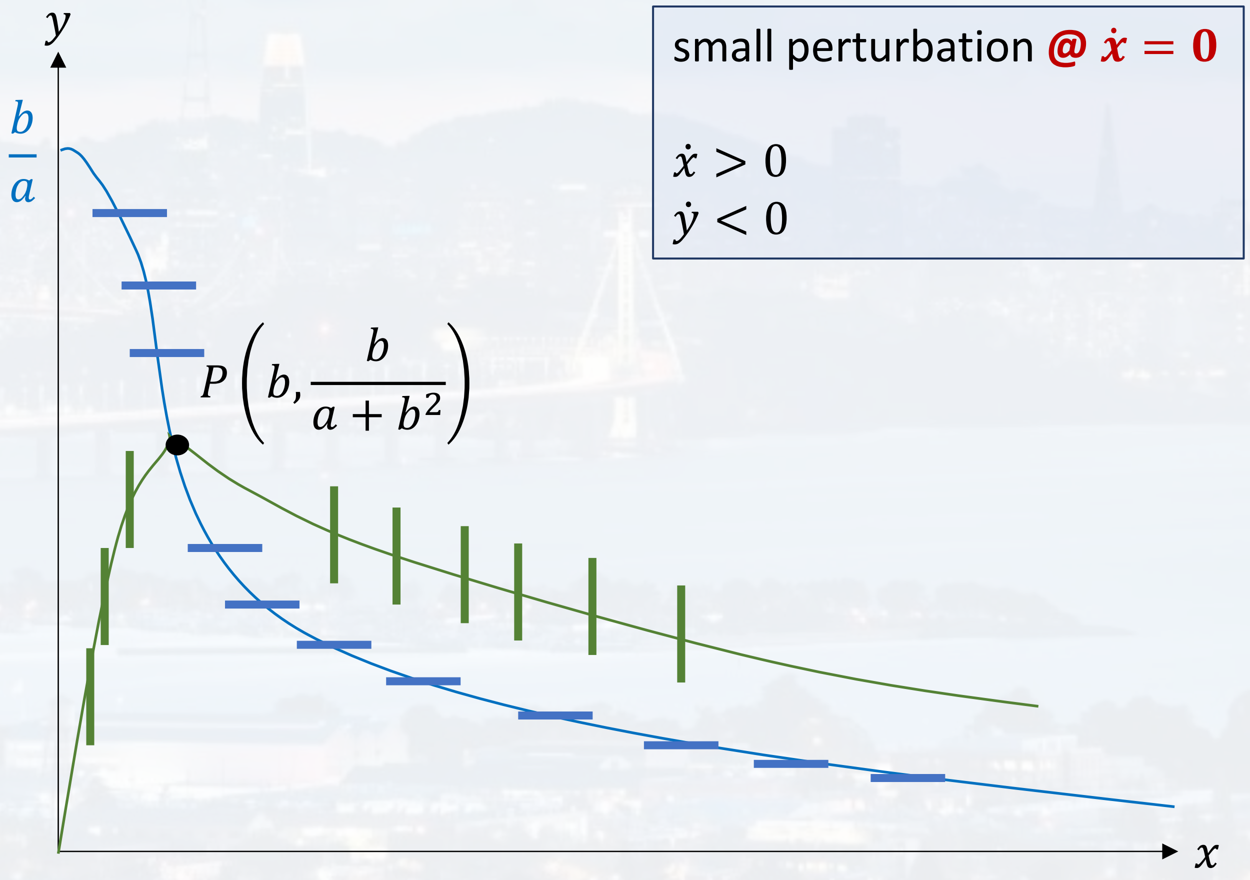
$f(x, y) = \dot{x}$ $\qquad$ $\dot{x} = -x + a\,y + x^2 y$ $\qquad$ a, b >0

$g(x, y) = \dot{y}$ $\qquad$ $\dot{y} = b - a\,y - x^2 y$

null clines

$\dot{x} = 0 \rightarrow \qquad y_1 = \frac{x}{a + x^2}$

$\dot{y} = 0 \rightarrow \qquad y_2 = \frac{b}{a + x^2}$



small perturbation @ $\dot{x} = 0$

$\dot{x} > 0$
$\dot{y} < 0$

Find out which way the system moves!

2D system

$$f(x,y) = \dot{x}$$

$$g(x,y) = \dot{y}$$

$$\dot{x} = -x + a\,y + x^2 y$$

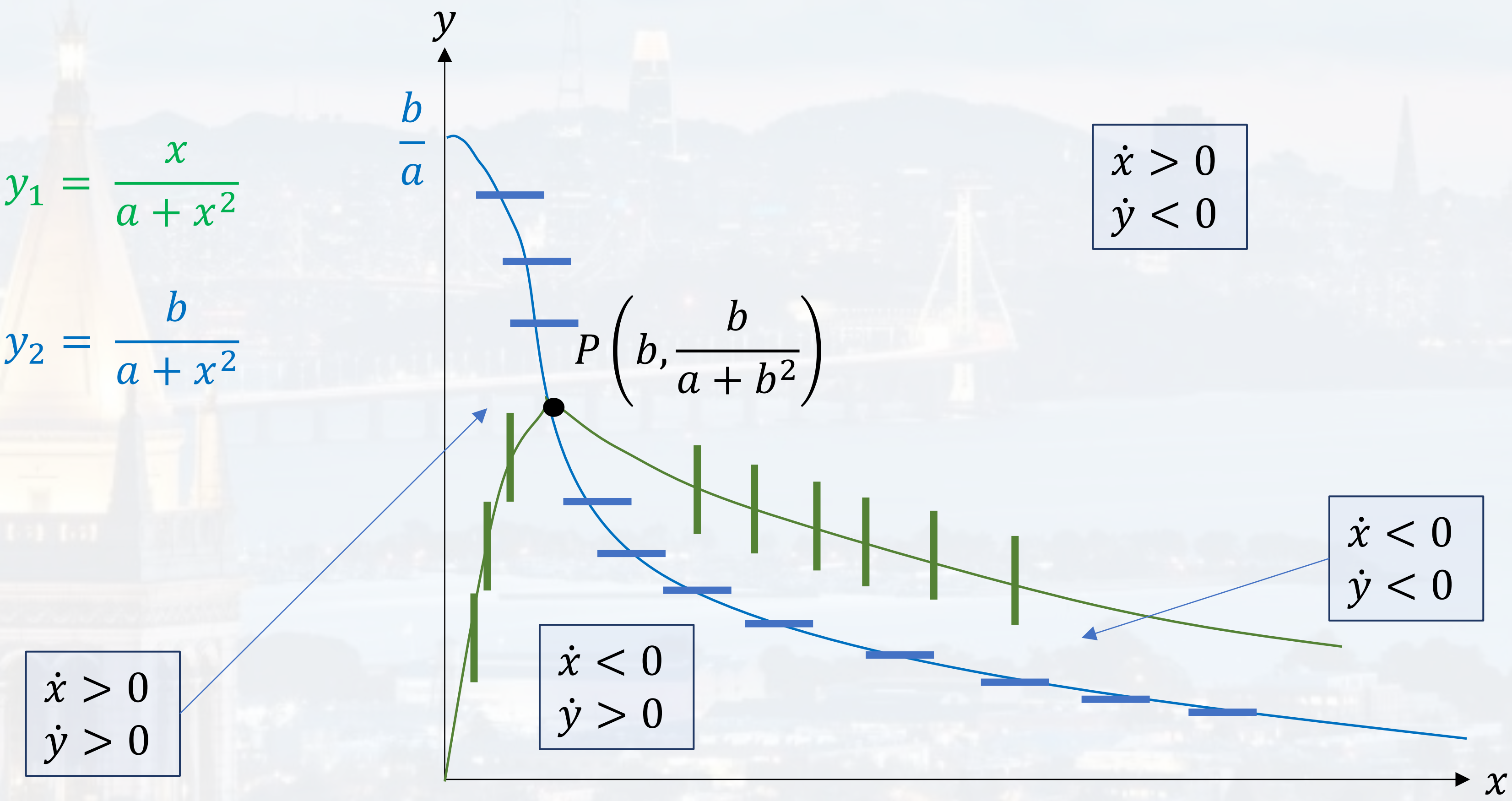
$$\dot{y} = b - a\,y - x^2 y$$

a, b >0

null clines

$$\dot{x} = 0 \rightarrow \quad y_1 = \frac{x}{a + x^2}$$

$$\dot{y} = 0 \rightarrow \quad y_2 = \frac{b}{a + x^2}$$



2D system

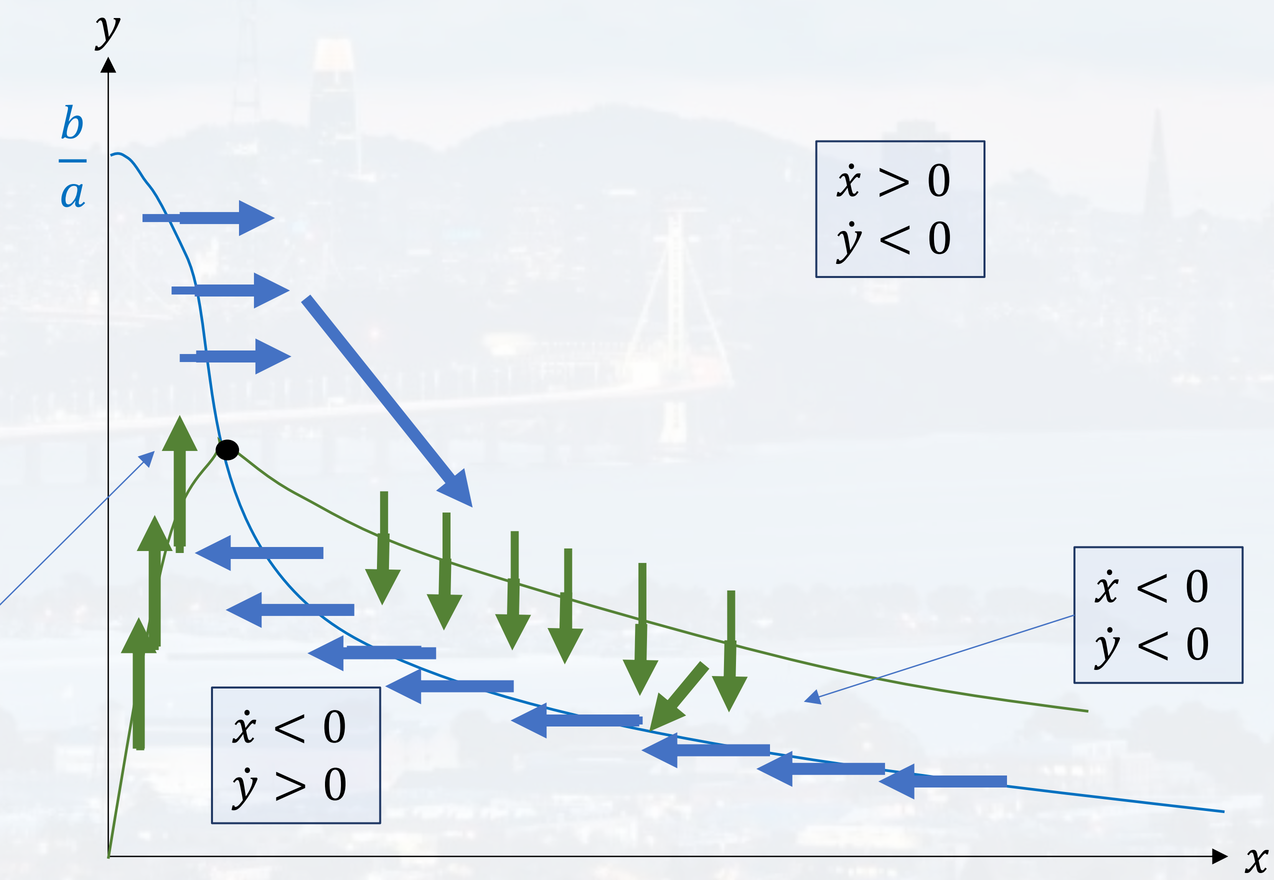
$$f(x,y) = \dot{x} \qquad \dot{x} = -x + a\,y + x^2 y \qquad \text{a, b > 0}$$

$$g(x,y) = \dot{y} \qquad \dot{y} = b - a\,y - x^2 y$$

null clines

$\dot{x} = 0 \rightarrow \qquad y_1 = \dfrac{x}{a + x^2}$

$\dot{y} = 0 \rightarrow \qquad y_2 = \dfrac{b}{a + x^2}$



2D system

$f(x,y) = \dot{x}$

$g(x,y) = \dot{y}$

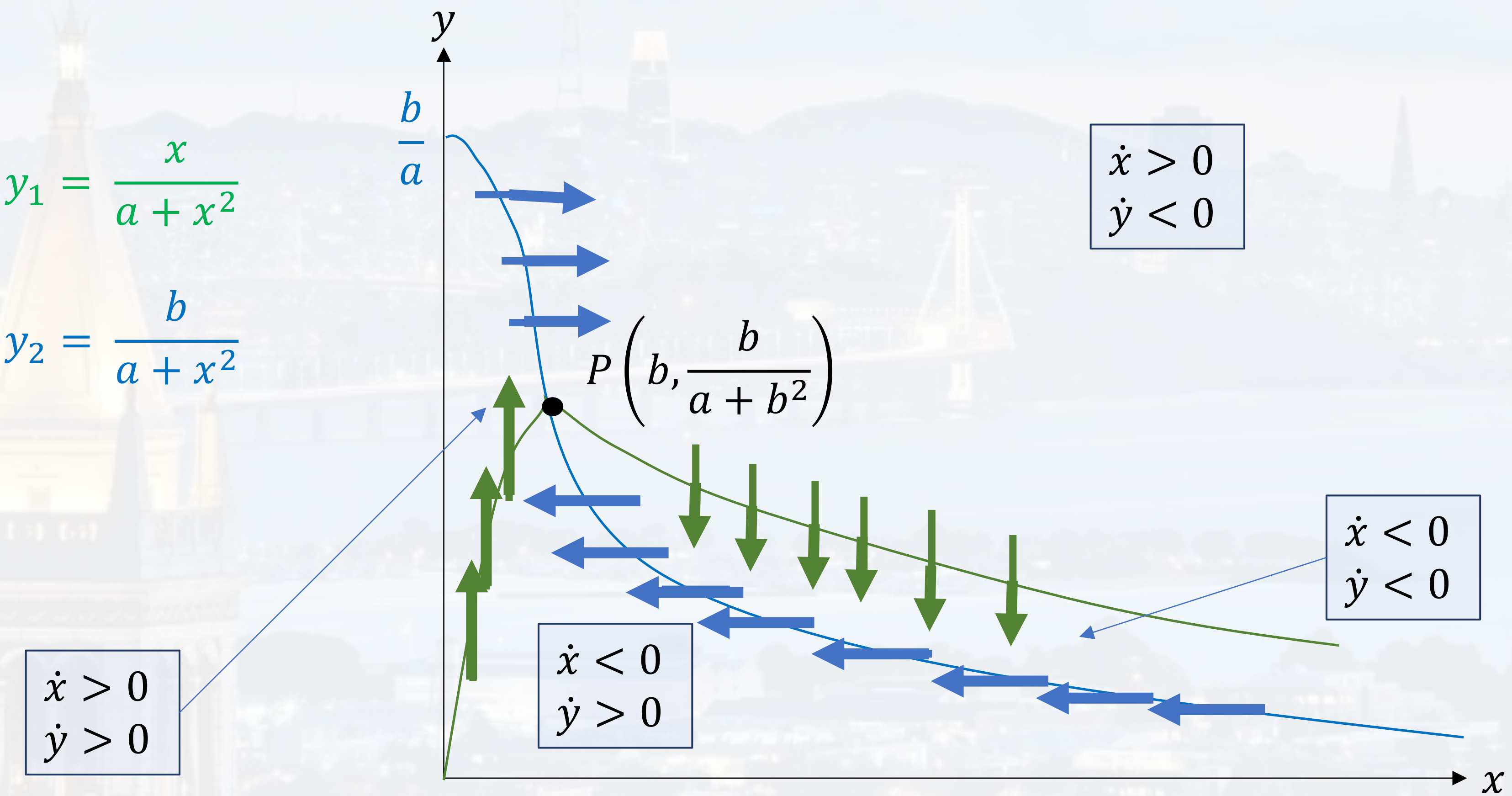
$\dot{x} = -x + a\,y + x^2 y$ a, b >0

$\dot{y} = b - a\,y - x^2 y$

null clines

$\dot{x} = 0 \rightarrow$ $y_1 = \frac{x}{a + x^2}$

$\dot{y} = 0 \rightarrow$ $y_2 = \frac{b}{a + x^2}$

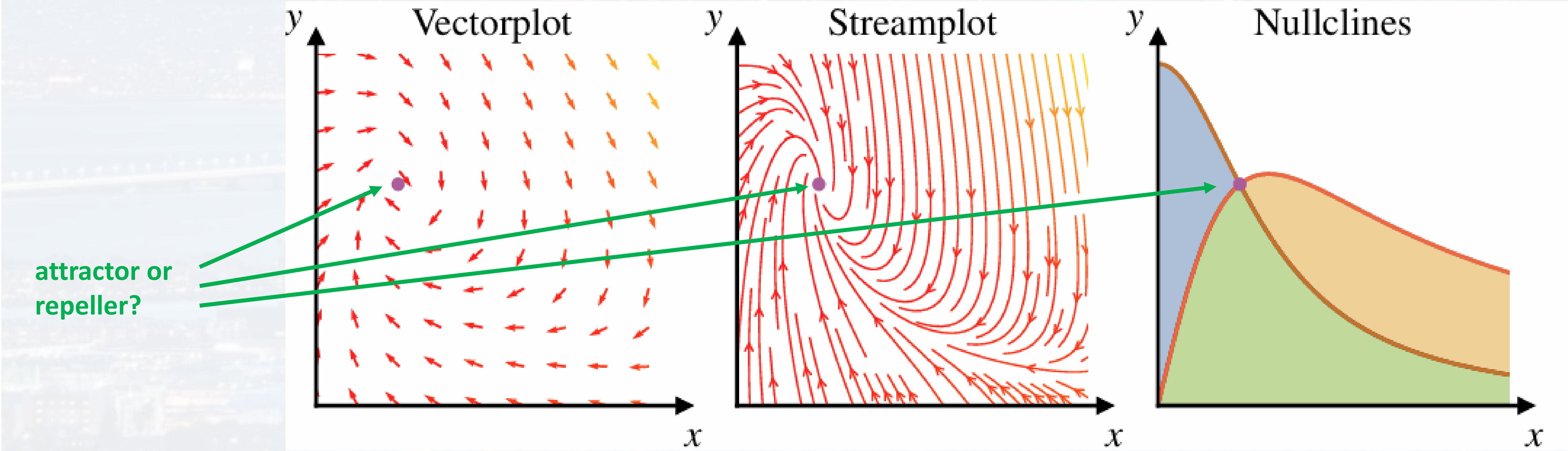


2D system

$$f(x,y) = \dot{x} \qquad \dot{x} = -x + a\,y + x^2 y \qquad \text{a, b >0}$$

$$g(x,y) = \dot{y} \qquad \dot{y} = b - a\,y - x^2 y \qquad \text{non-linear, coupled ODEs}$$

$\dot{x} = 0 \rightarrow \quad y_1 = \frac{x}{a + x^2}$ $\qquad \dot{y} = 0 \rightarrow \quad y_2 = \frac{b}{a + x^2}$

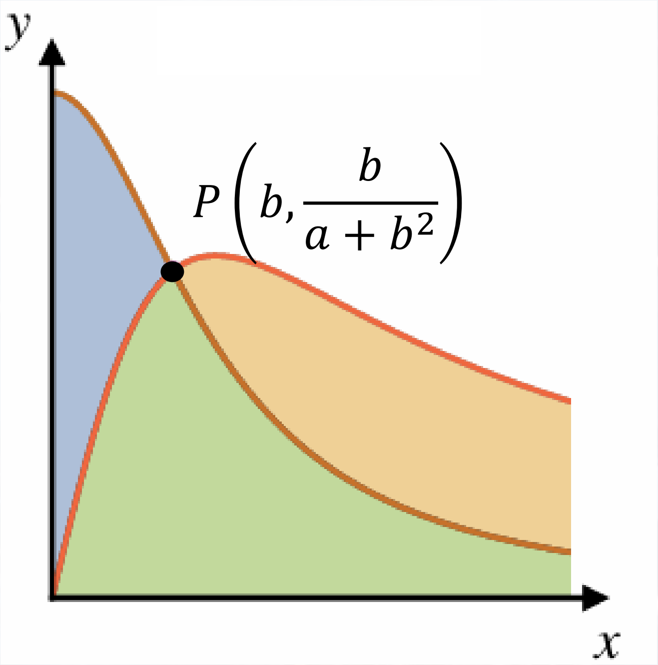


2D system

$f(x,y) = \dot{x}$ $\quad\quad$ $\dot{x} = -x + a\,y + x^2 y$ $\quad\quad$ a, b >0

$g(x,y) = \dot{y}$ $\quad\quad$ $\dot{y} = b - a\,y - x^2 y$ $\quad\quad$ non-linear, coupled ODEs

stability of P:



$$\underbrace{f(x^* + \varepsilon_x, y^* + \varepsilon_y)}_{\frac{d\,\varepsilon_x(t)}{dt}} \approx \underbrace{f(x^*, y^*)}_{=0} + \underbrace{\frac{\partial f(x,y)}{\partial x}|_{x^*,y^*}}_{\alpha} \varepsilon_x + \underbrace{\frac{\partial f(x,y)}{\partial y}|_{x^*,y^*}}_{\beta} \varepsilon_y$$

$$\underbrace{g(x^* + \varepsilon_x, y^* + \varepsilon_y)}_{\frac{d\,\varepsilon_y(t)}{dt}} \approx \underbrace{g(x^*, y^*)}_{=0} + \underbrace{\frac{\partial g(x,y)}{\partial x}|_{x^*,y^*}}_{\gamma} \varepsilon_x + \underbrace{\frac{\partial g(x,y)}{\partial y}|_{x^*,y^*}}_{\delta} \varepsilon_y$$

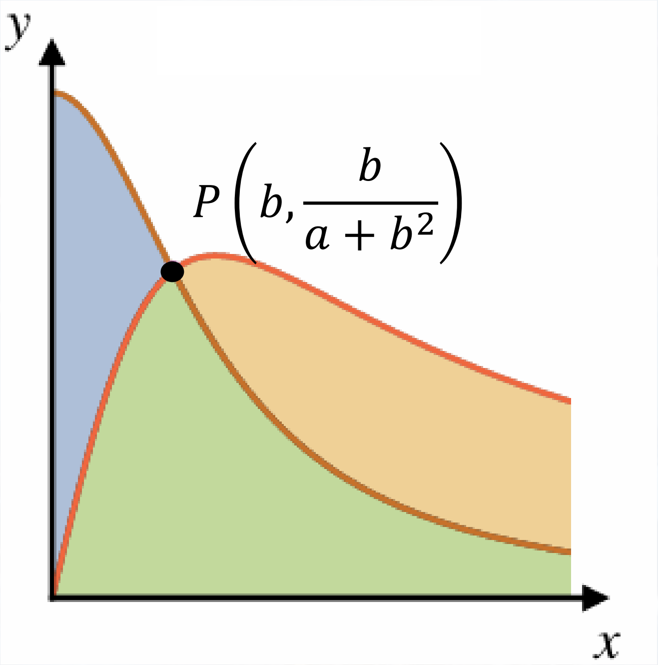
$$\dot{\vec{\varepsilon}} = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \vec{\varepsilon}$$

2D system

$f(x,y) = \dot{x}$ $\qquad \dot{x} = -x + a\,y + x^2 y$ $\qquad$ a, b >0

$g(x,y) = \dot{y}$ $\qquad \dot{y} = b - a\,y - x^2 y$ $\qquad$ non-linear, coupled ODEs

stability of P:



$$\dot{\vec{\varepsilon}} = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \vec{\varepsilon} \qquad A = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \qquad \varepsilon(t) = \varepsilon_0\, e^{\frac{df(x)}{dx}|_{x=x^*}\, t}$$

$$\vec{\varepsilon}(t) = \vec{\varepsilon}(t=0)\, e^{\lambda t}$$ λ: eigenvalue of A

$$0 \overset{!}{=} det \begin{pmatrix} \alpha - \lambda & \beta \\ \gamma & \delta - \lambda \end{pmatrix} = \lambda^2 - (\alpha+\delta)\lambda + (\alpha\delta - \gamma\beta)$$

one can show that $\tau = \dfrac{b^4 + (2a-1)b^2 + a(1+a)}{a + b^2}$

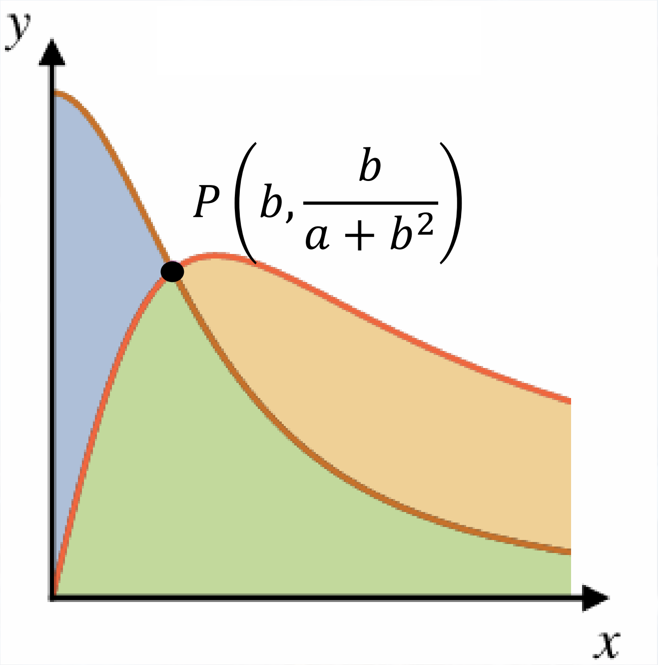
$\tau > 0$ P is a repeller
$\tau < 0$ P is an attractor

2D system

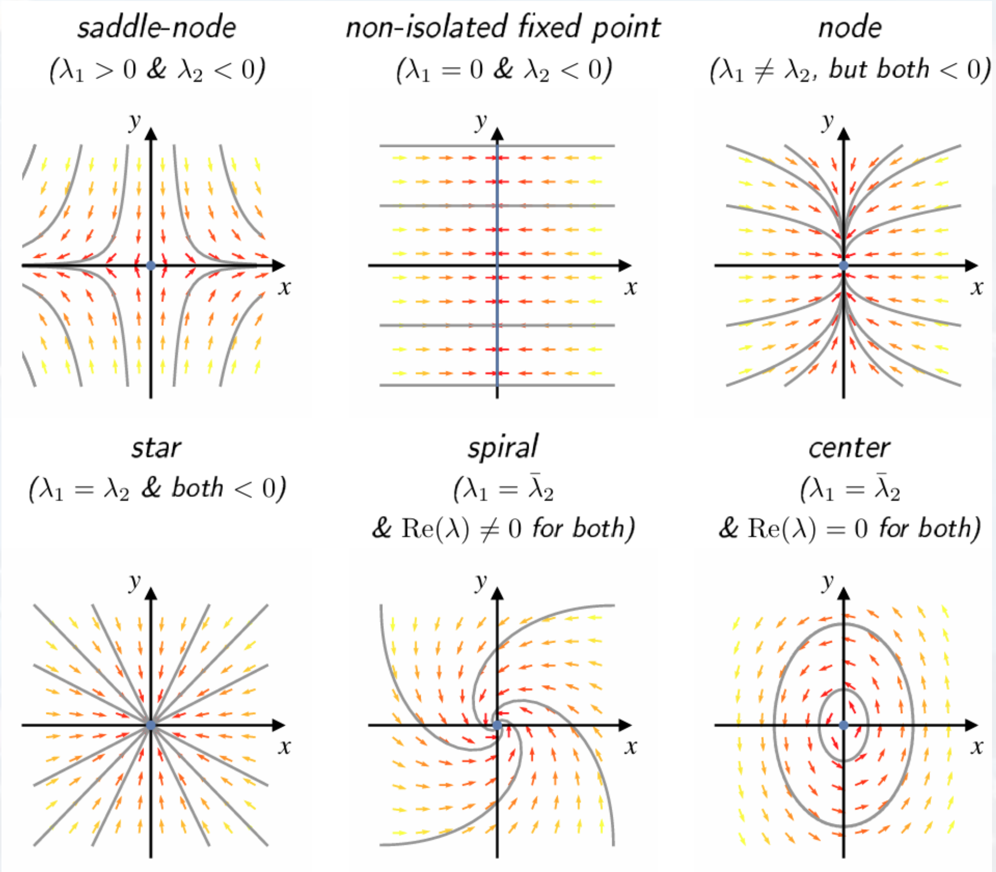
$$f(x,y) = \dot{x} \qquad \dot{x} = -x + a\,y + x^2 y \qquad \text{a, b >0}$$

$$g(x,y) = \dot{y} \qquad \dot{y} = b - a\,y - x^2 y \qquad \text{non-linear, coupled ODEs}$$

stability of P: $\vec{\varepsilon}(t) = \vec{\varepsilon}(t=0)\, e^{\lambda t}$



attractor: real part of all eigenvalues has to be negative!



2D system

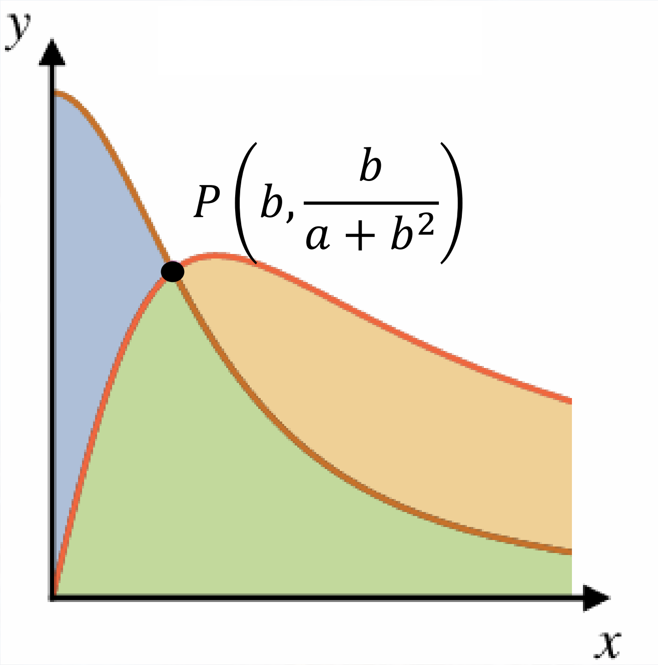
$$f(x,y) = \dot{x} \qquad \dot{x} = -x + a\,y + x^2 y$$

$$g(x,y) = \dot{y} \qquad \dot{y} = b - a\,y - x^2 y$$

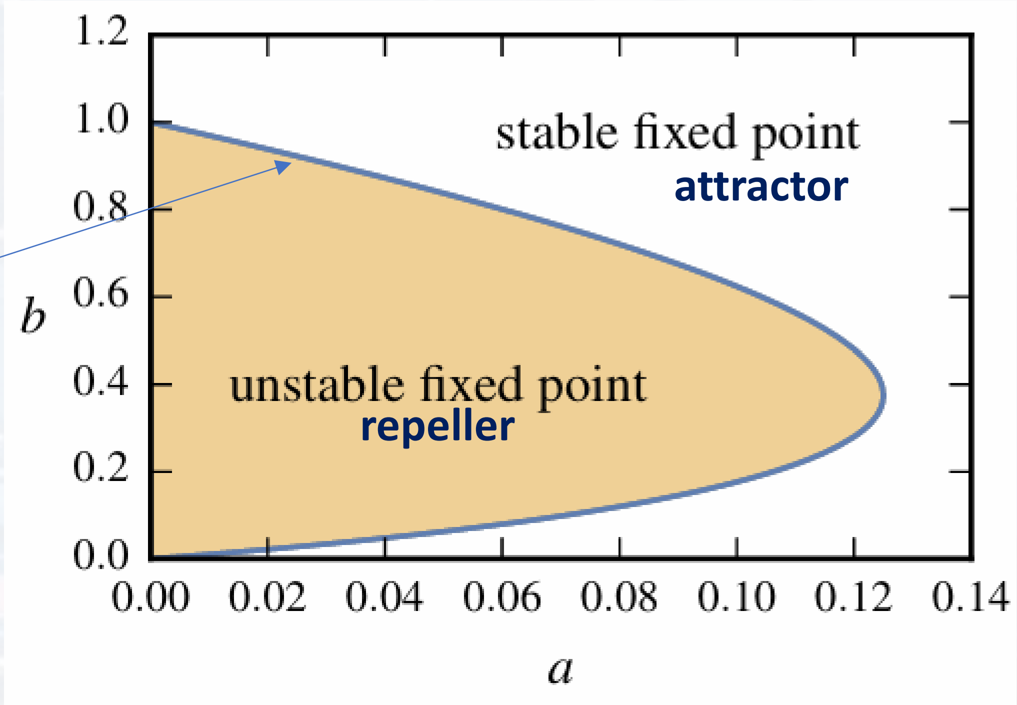
stability of P: $\vec{\varepsilon}(t) = \vec{\varepsilon}(t=0)\, e^{\lambda t}$ $\quad \tau = \dfrac{b^4 + (2a-1)b^2 + a(1+a)}{a + b^2}$

$\tau > 0$ **P is a repeller**

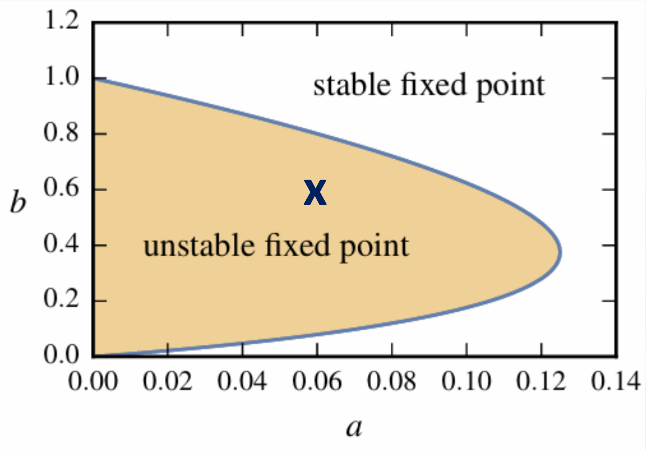
$\tau < 0$ **P is an attractor**



$\tau = 0$



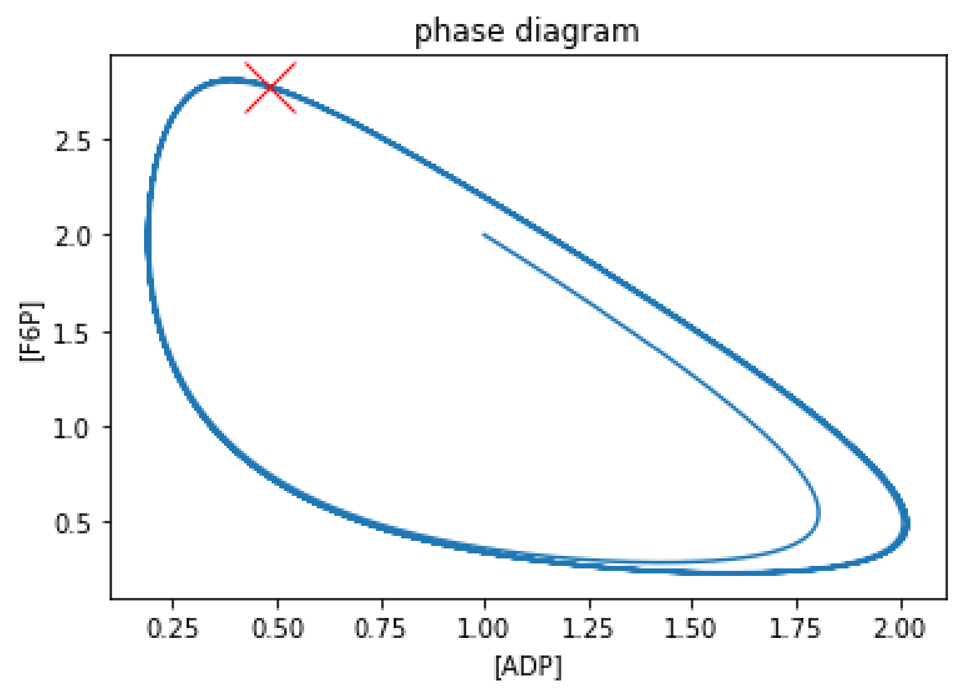
2D system

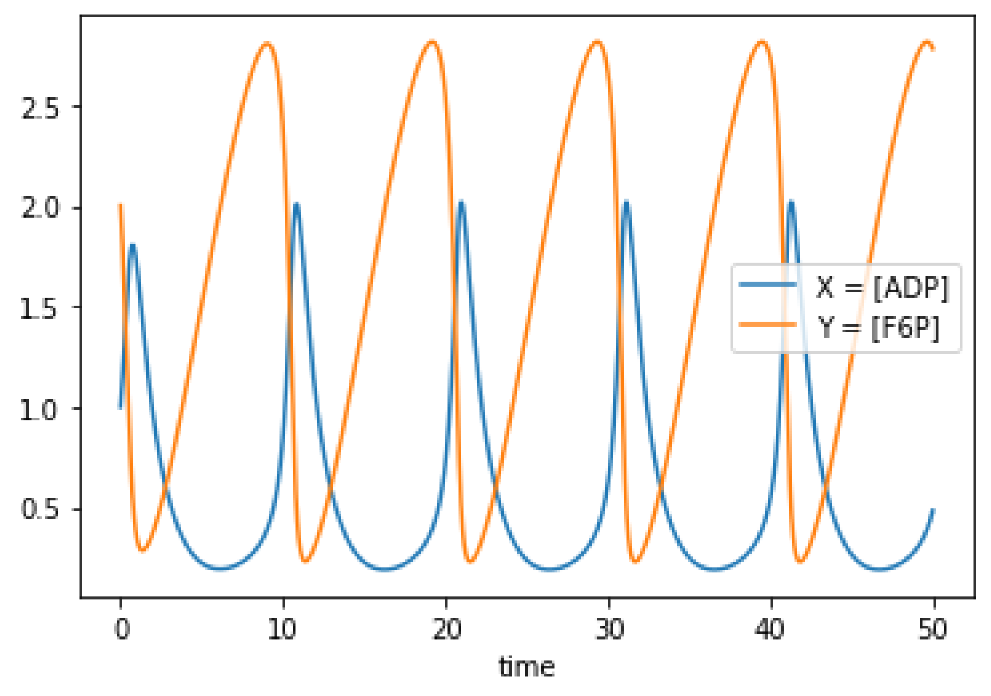


$$P\left(b, \frac{b}{a+b^2}\right) = (0.6, 1.42)$$

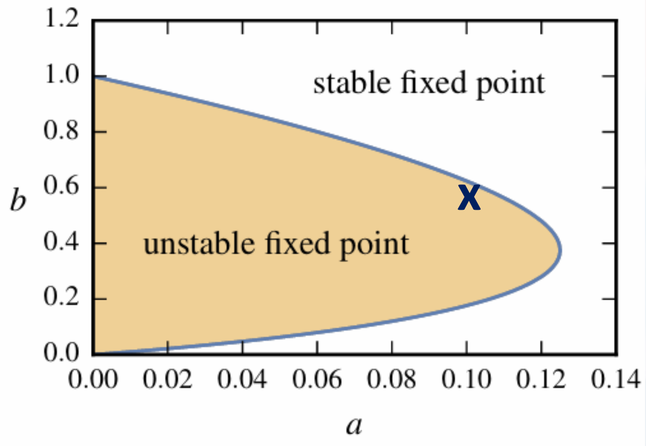
$\dot{x}$ and $\dot{y}$ model **Glycolysis**
(Selkov et al., 1968)

X = [ADP], Y = [F6P]





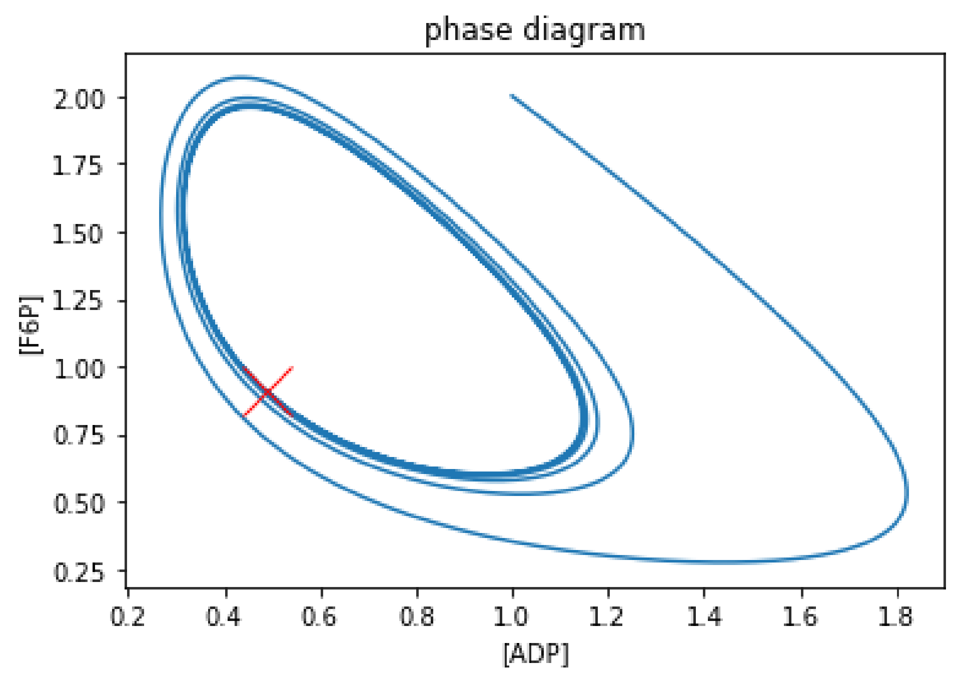
2D system

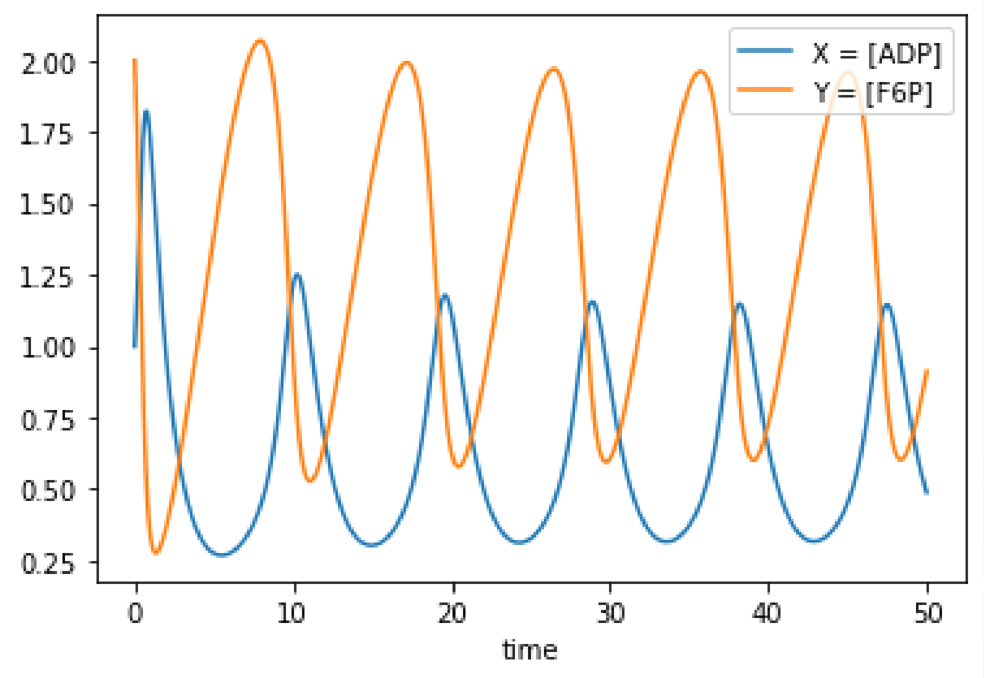


$$P\left(b, \frac{b}{a+b^2}\right) = (0.6, 1.30)$$

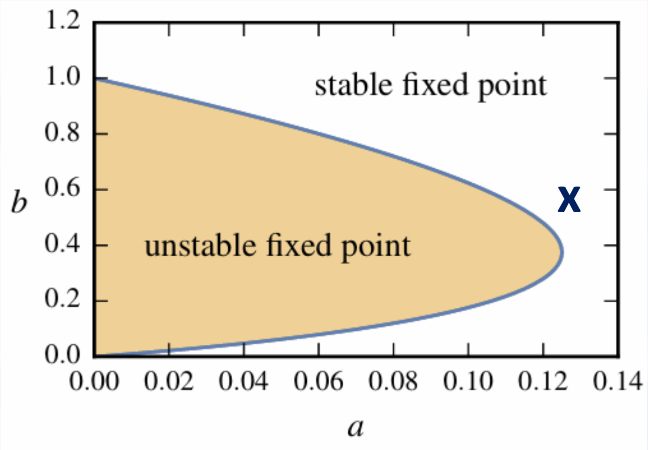
$\dot{x}$ and $\dot{y}$ model **Glycolysis**
(Selkov et al., 1968)

X = [ADP], Y = [F6P]





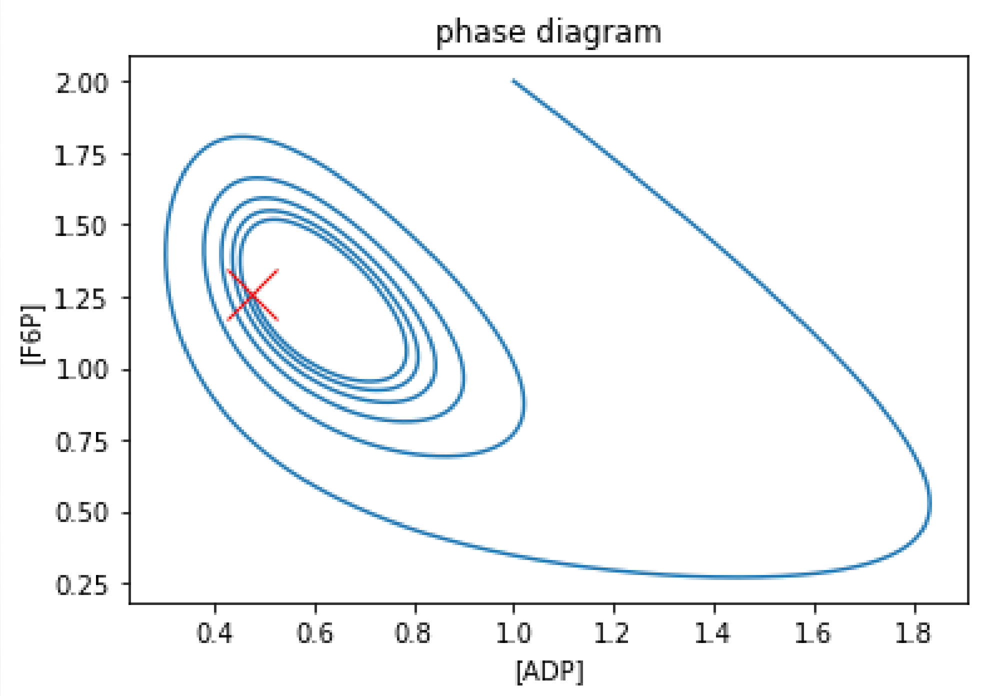
2D system

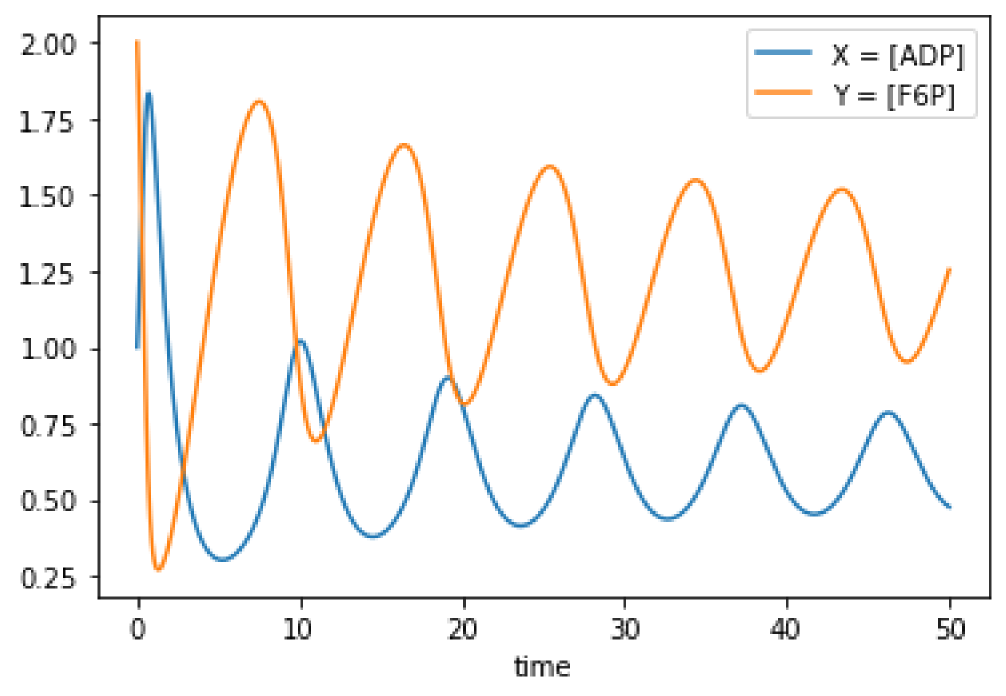


$$P\left(b, \frac{b}{a+b^2}\right) = (0.6, 1.24)$$

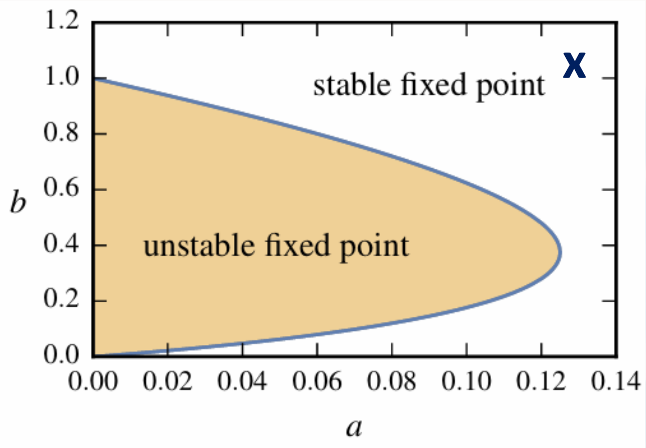
$\dot{x}$ and $\dot{y}$ model **Glycolysis**
(Selkov et al., 1968)

X = [ADP], Y = [F6P]





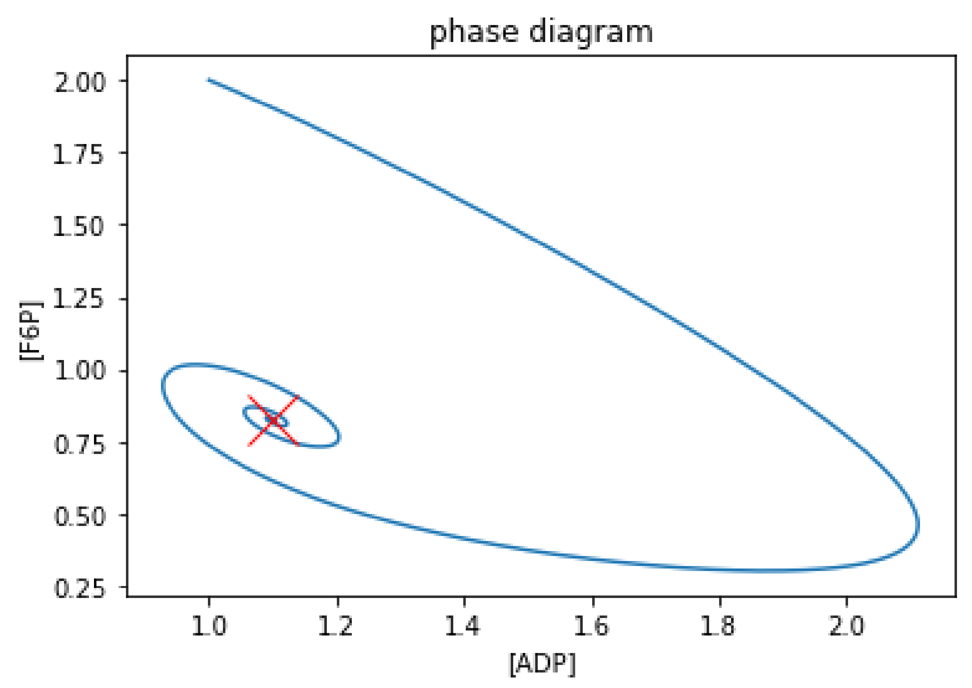
2D system

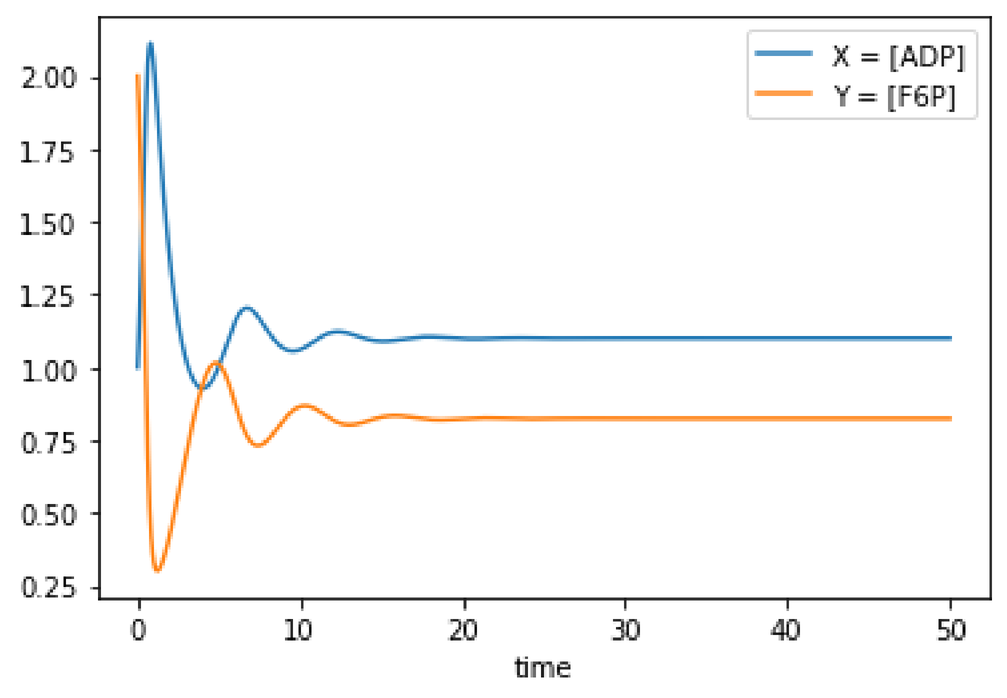


$$P\left(b, \frac{b}{a+b^2}\right) = (1.1, 0.82)$$

$\dot{x}$ and $\dot{y}$ model **Glycolysis**
(Selkov et al., 1968)

X = [ADP], Y = [F6P]





Runge-Kutta-Heun

$\frac{dy}{dt} = f(x(t), t)$ initial condition: $y_0 = y(t_0)$ goal: $y(t)$

$$y(t+dt) = y(t) + \frac{dy}{dt} dt + \frac{1}{2}\frac{d^2y}{dt^2} dt^2 + \cdots$$

$$y(t+dt) = y(t) + f(x,t)\, dt + \frac{1}{2}\frac{d}{dt} f(x,t)\, dt^2 + \cdots$$

$$y(t+dt) = y(t) + f(x,t)\, dt + \frac{1}{2}\left[\frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial t}\right] dt^2 + \cdots$$

note: more general $\frac{dy}{dx} = f(x(t), y(x(t)))$

Runge-Kutta-Heun

$$\frac{dy}{dx_-} = f(x(t), y(x(t)))$$

$$\frac{dy}{dx_+} = f(x(t) + \Delta x, y + \Delta x\, f(x(t)))$$

$$\frac{dy}{dx} = \frac{1}{2}\left[\frac{dy}{dx_-} + \frac{dy}{dx_+}\right]$$

updating:

$$x(new) = x(t) + \Delta x$$

$$y(new) = y(t) + \Delta x \frac{1}{2}\left[\frac{dy}{dx_-} + \frac{dy}{dx_+}\right] = y(t) + \Delta x \frac{1}{2}[f(x(t), y(x(t))) + f(x(t) + \Delta x, y + \Delta x\, f(x(t)))]$$

Runge-Kutta-Heun

$$x(new) = x(t) + \Delta x$$

$$y(new) = y(t) + \Delta x \frac{1}{2}\left[\frac{dy}{dx_-} + \frac{dy}{dx_+}\right] = y(t) + \Delta x \frac{1}{2}[f(x(t), y(x(t))) + f(x(t) + \Delta x, y + \Delta x\, f(x(t)))]$$

more precise (here shown for for 1D *y = y(t)):*

$$k_1 = f(t_n, y_n)$$

$$k_2 = f(t_n + \frac{\Delta t}{2}, y_n + \frac{\Delta t k_1}{2})$$

$$k_3 = f(t_n + \frac{\Delta t}{2}, y_n + \frac{\Delta t k_2}{2})$$

$$k_4 = f(t_n + \Delta t, y_n + \Delta t k_3)$$

$$t(new) = t_{n+1} = t_n + \Delta t$$

$$\mathrm{y}(new) = y_{n+1} = y_n + \frac{\Delta t}{6}\underbrace{[k_1 + 2k_2 + 2k_3 + k_4]}_{\text{recall: Simpson \& Simpson 3/8}}$$

Runge-Kutta-Heun

$$x(new) = x(t) + \Delta x$$

$$y(new) = y(t) + \Delta x \frac{1}{2}\left[\frac{dy}{dx_-} + \frac{dy}{dx_+}\right] = y(t) + \Delta x \frac{1}{2}[f(x(t), y(x(t))) + f(x(t) + \Delta x, y + \Delta x\, f(x(t)))]$$

```
from scipy.integrate import solve_ivp

method = 'RK45'
```

4: referring to the number of subintervals for integration (4 is equivalent to the Simpson rule)
5: referring to the order of the Taylor approximation for the derivatives

$$\dot{x} = -x + a\,y + x^2 y$$

$$\dot{y} = b - a\,y - x^2 y$$

$\dot{x}$ and $\dot{y}$ model **Glycolysis**
(Selkov et al., 1968)

initial values: x(t=0) and y(t=0)

[t_start, t_end]

parameter of the function

contains the actual ODEs

```
def SolveGlycolysis(Init, t_span, a, b):

    XY      = ode_solver(Glycolysis, Init, t_span, method = 'RK45',\
                         a = a, b = b)

    t = XY.t
    X = XY.y[0,:]
    Y = XY.y[1,:]

    #######################plotting result############################

    #...
```

$$\dot{x} = -x + a\,y + x^2 y$$

$$\dot{y} = b - a\,y - x^2 y$$

```
def SolveGlycolysis(Init, t_span, a, b):

    XY      = ode_solver(Glycolysis, Init, t_span, method = 'RK45',\
                        a = a, b = b)


def Glycolysis(Init, t, a, b):

        x = Init[0]
        y = Init[1]

        dx = -x + a*y + (x**2)*y
        dy = b  - a*y - (x**2)*y

        D = [dx, dy]

        return D
```

note: t is an input variable, even though it is not being used explicitly
→ integration over t

$$\dot{x} = -x + a\,y + x^2 y$$

$$\dot{y} = b - a\,y - x^2 y$$

the actual solver

```
def SolveGlycolysis(Init, t_span, a, b):

    XY      = ode_solver(Glycolysis, Init, t_span, method = 'RK45',\
                         a = a, b = b)

    t = XY.t
    X = XY.y[0,:]
    Y = XY.y[1,:]

    #######################plotting result################################

    #...
```

$$\dot{x} = -x + a\,y + x^2 y$$

$$\dot{y} = b - a\,y - x^2 y$$

```python
def SolveGlycolysis(Init, t_span, a, b):

    XY      = ode_solver(Glycolysis, Init, t_span, method = 'RK45',\
                         a = a, b = b)



from scipy.integrate import solve_ivp

def ode_solver(ode_func, Init, t_span, method = 'RK45', **params):


        result = solve_ivp(fun = lambda t, y: ode_func(y, t, **params),\
                           t_span = t_span, y0 = Init, method = method,\
                           rtol = 1e-9, atol = 1e-9, max_step = 0.01)


        return result
```

$$\dot{x} = -x + a\,y + x^2 y$$

$$\dot{y} = b - a\,y - x^2 y$$

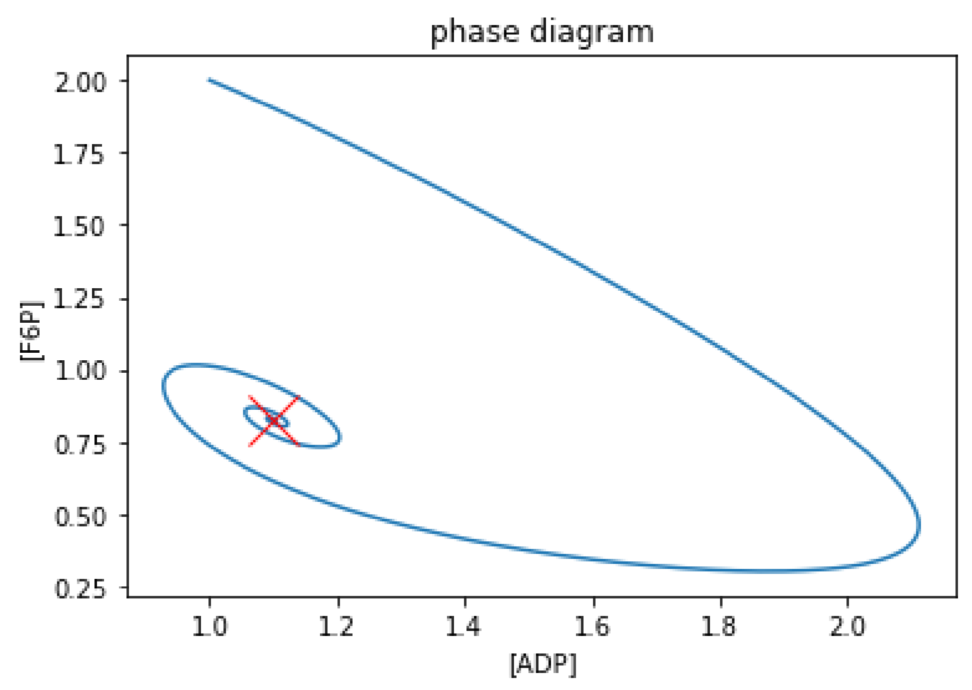
```
def SolveGlycolysis(Init, t_span, a, b):

    XY      = ode_solver(Glycolysis, Init, t_span, method = 'RK45',\
                         a = a, b = b)


from scipy.integrate import solve_ivp

def ode_solver(ode_func, Init, t_span, method = 'RK45', **params):


        result = solve_ivp(fun = lambda t, y: ode_func(y, t, **params),\
                           t_span = t_span, y0 = Init, method = method,\
                           rtol = 1e-9, atol = 1e-9, max_step = 0.01)


        return result
```

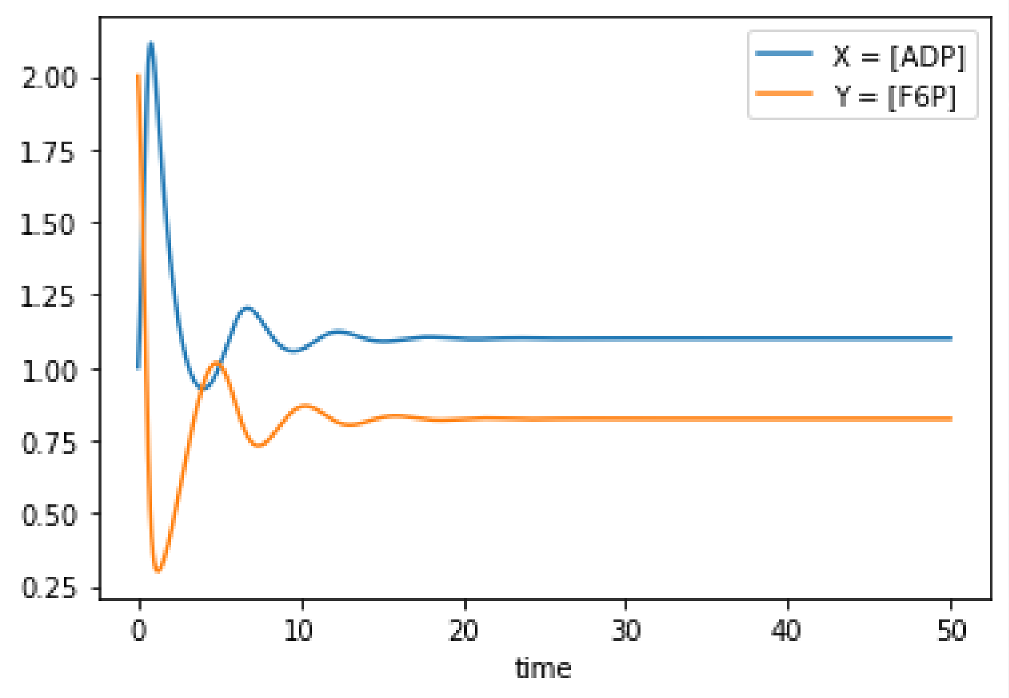
$$\dot{x} = -x + a\,y + x^2 y$$

$$\dot{y} = b - a\,y - x^2 y$$

run:

```
a = 0.125
b = 1.1
SolveGlycolysis([1, 2], [0, 50], a, b)
```





run:

```
SolvePhageTherapy(Init, t_span, rates)
```

Synergistic elimination of bacteria by phage and the immune system

Chung Yin (Joey) Leung* and Joshua S. Weitz†
School of Biology, Georgia Institute of Technology, Atlanta, Georgia 30332, USA and
School of Physics, Georgia Institute of Technology, Atlanta, Georgia 30332, USA

$$\dot{B} = \boxed{\overbrace{rB\left(1-\frac{B}{K_C}\right)}^{\text{Growth}}} - \overbrace{\phi BP}^{\text{Lysis}} - \overbrace{\frac{\epsilon IB}{1+B/K_D}}^{\text{Immune killing}},$$

$$\dot{P} = \overbrace{\beta\phi BP}^{\text{Replication}} - \overbrace{\omega P}^{\text{Decay}},$$

$$\dot{I} = \overbrace{\boxed{\alpha I\left(1-\frac{I}{K_I}\right)}\frac{B}{B+K_N}}^{\text{Immune stimulation}}.$$

B:	bacteria
P:	phages
I:	immune cells

$$\frac{dN}{dt} = c_0\left(1-\frac{1}{\kappa}N\right)N$$

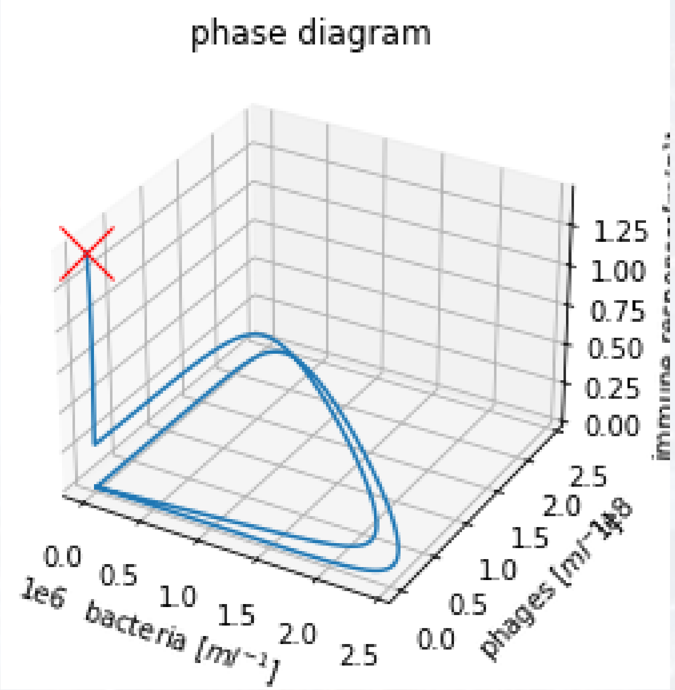
recall:
Verhulst Equation

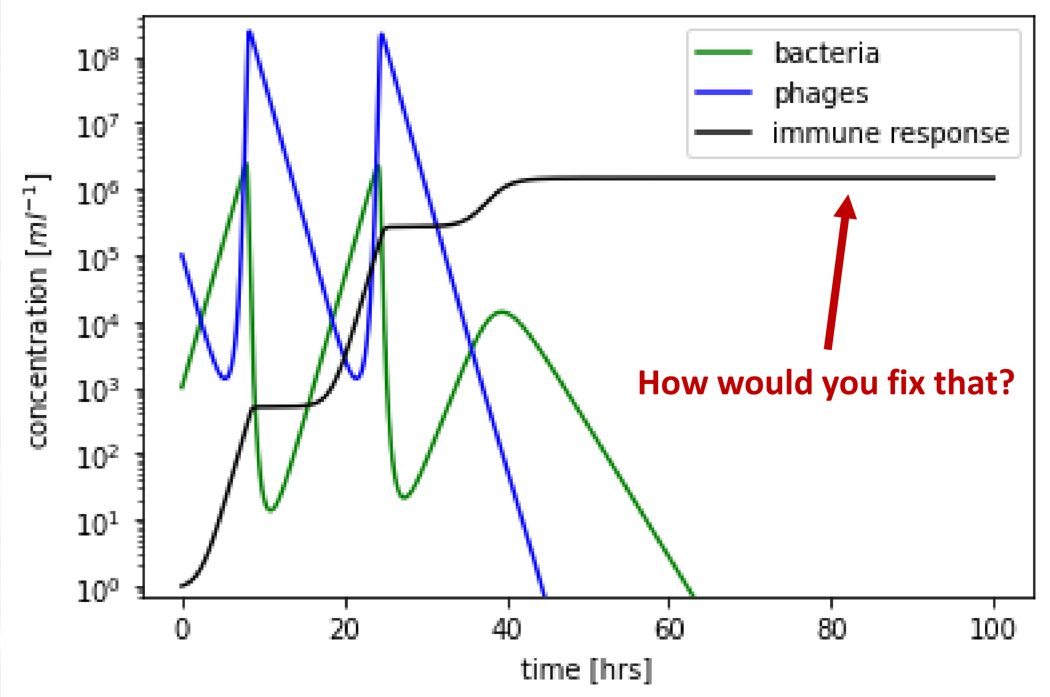
<u>run:</u>

```
SolvePhageTherapy(Init, t_span, rates)
```

Synergistic elimination of bacteria by phage and the immune system

Chung Yin (Joey) Leung* and Joshua S. Weitz†
School of Biology, Georgia Institute of Technology, Atlanta, Georgia 30332, USA and
School of Physics, Georgia Institute of Technology, Atlanta, Georgia 30332, USA





Berkeley
UNIVERSITY OF CALIFORNIA

Thank you for your attention!