growth_single

November 15, 2022

1 Growth of a single species

In this notebook we will study the dynamics of a minimal model of the growth of a single species in a chemostat.

1.1 Constructing the model

We consider the dynamics of microbial growth of a single species in stirred tank reactors (chemostat) assuming the medium is homogeneous.

Let's consider: - $s(t)$ is the substrate concentration $(g \cdot L^{-1})$ at time t - $x(t)$ is the biomass concentration $(g \cdot L^{-1})$ at time t - $v(t)$ is the volume of the culture medium (L) at time t - S_{in} is the substrate intake concentration $(g \cdot L^{-1})$ - F_{in} is the influent flow rate (Lh^{-1}) - F_{out} is the effluent flow rate (Lh^{-1}) - $\mu(\cdot)$ is a the specific growth velocity function, assumed to be positive and of class $\mathcal{C}^1$, its values are measured in h^{-1}. - Y is the yield of substrate consumption

The model can be constructed as follows:

$$\begin{cases} \frac{\mathrm{d}(vx)}{\mathrm{d}t} &= \underbrace{\mu(\cdot)vx}_{\text{biomass growth}} - \underbrace{F_{\text{out}}x}_{\text{biomass loss}} \\ \frac{\mathrm{d}(vs)}{\mathrm{d}t} &= -\underbrace{\frac{\mu(\cdot)}{Y}vx}_{\text{substrate consumption}} + \underbrace{F_{\text{in}}S_{\text{in}}}_{\text{substrate mass intake}} - \underbrace{F_{\text{out}}s}_{\text{substrate mass loss}} \\ \frac{\mathrm{d}v}{\mathrm{d}t} &= F_{\text{in}} - F_{\text{out}} \end{cases}$$

The equations describing $\frac{\mathrm{d}(vx)}{\mathrm{d}t}$ and $\frac{\mathrm{d}(vs)}{\mathrm{d}t}$ can be simplified by exploiting the derivative of a product

$$\frac{\mathrm{d}(vx)}{\mathrm{d}t} = \mu(\cdot)vx - F_{\text{out}}x \tag{1}$$
$$\frac{\mathrm{d}v}{\mathrm{d}t}x + v\frac{\mathrm{d}x}{\mathrm{d}t} = \mu(\cdot)vx - F_{\text{out}}x \tag{2}$$
$$(F_{\text{in}} - F_{\text{out}})x + v\frac{\mathrm{d}x}{\mathrm{d}t} = \mu(\cdot)vx - F_{\text{out}}x \tag{3}$$
$$\frac{\mathrm{d}x}{\mathrm{d}t} = \mu(\cdot)x - \frac{F_{\text{in}}}{v}x \tag{4}$$

similarly

$$\frac{\mathrm{d}(vs)}{\mathrm{d}t} = -\frac{\mu(\cdot)}{Y}vx + F_{\text{in}}S_{\text{in}} - F_{\text{out}}s \tag{5}$$
$$\frac{\mathrm{d}v}{\mathrm{d}t}s + v\frac{\mathrm{d}s}{\mathrm{d}t} = -\frac{\mu(\cdot)}{Y}vx + F_{\text{in}}S_{\text{in}} - F_{\text{out}}s \tag{6}$$
$$(F_{\text{in}} - F_{\text{out}})s + v\frac{\mathrm{d}s}{\mathrm{d}t} = -\frac{\mu(\cdot)}{Y}vx + F_{\text{in}}S_{\text{in}} - F_{\text{out}}s \tag{7}$$
$$\frac{\mathrm{d}s}{\mathrm{d}t} = -\frac{\mu(\cdot)}{Y}x + \frac{F_{\text{in}}}{v}(S_{\text{in}} - s) \tag{8}$$

By defining the dilution rate $D = \frac{F_{\text{in}}}{v}$ we can rewrite the system in terms of concentrations

$$\begin{cases} \frac{\mathrm{d}s}{\mathrm{d}t} = -\frac{\mu(\cdot)}{Y}x + D(S_{\text{in}} - s) \\ \frac{\mathrm{d}x}{\mathrm{d}t} = (\mu - D)x \\ \frac{\mathrm{d}v}{\mathrm{d}t} = F_{\text{in}} - F_{\text{out}} \end{cases}$$

where - $s(t)$ is the substrate concentration $(g \cdot L^{-1})$ at time t - $x(t)$ is the biomass concentration $(g \cdot L^{-1})$ at time t - $v(t)$ is the volume of the culture medium (L) at time t - S_{in} is the substrate intake concentration $(g \cdot L^{-1})$ - D is the dilution rate (h^{-1}) - $\mu(\cdot)$ is a the specific growth velocity function, assumed to be positive and of class $\mathcal{C}^1$, its values are measured in h^{-1}. - Y is the yield of substrate consumption

All of the values and parameters are positive.

In this study, we will only consider the case of a continuous stirred tank reactor where the rate of outflow is equal to the rate of inflow which means that $\frac{\mathrm{d}v}{\mathrm{d}t} = 0$. We denote $F = F_{\text{in}} = F_{\text{out}}$ and $D = \frac{F}{v}$

1.2 Specific growth velocity μ

For the minimal model, we consider μ a function of the substrate concentration s. Moreover, we suppose that μ is of the *Monod type*, such that

- $\mu(0) = 0$
- $\mu'(s) > 0, \forall s \geq 0$

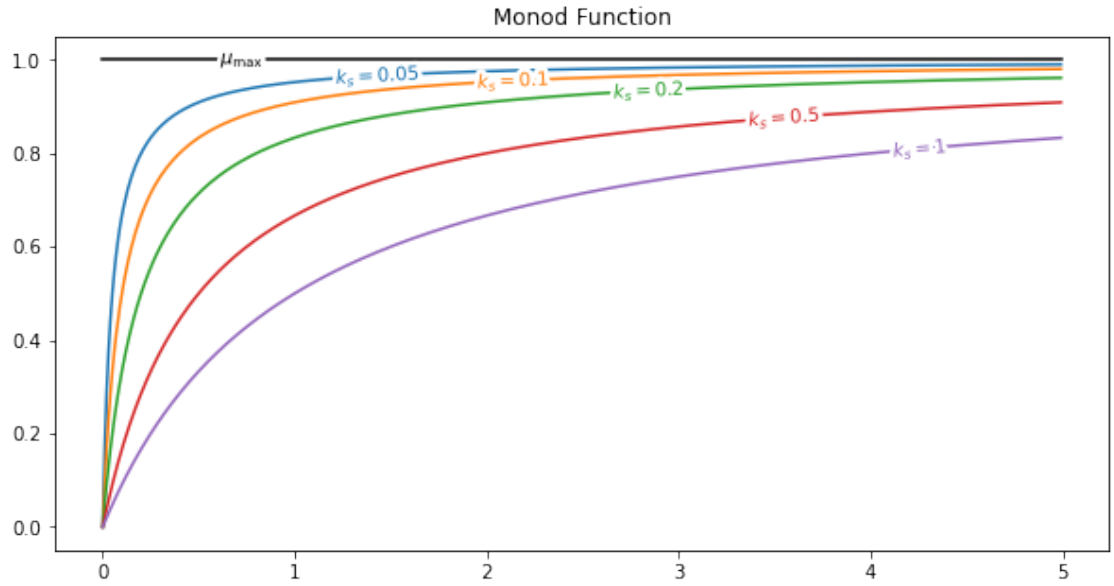
- $\mu([0, \infty[) = [0, \mu_{\max}[$

The **Monod function**, verifying these conditions is defined as

$$\mu(s) = \frac{\mu_{\max} s}{k_s + s}$$

where k_s is called the *semi-saturation* constant since $\mu(k_s) = \frac{\mu_{\max}}{2}$.

[1]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from labellines import labelLines

def mu(s,  max=1.0, ks=0.2):
    return ( max*s) / (ks + s)

plt.figure(figsize=(10,5))
plt.title("Monod Function")

s = np.arange(0, 5, 0.01)
plt.plot(s, np.ones(len(s)), color='black', label='$\\mu_\\max$')
for ks in [0.05, 0.1, 0.2, 0.5, 1]:
    plt.plot(s, mu(s, ks=ks), label='$k_s={}$'.format(ks))

labelLines(plt.gca().get_lines(), zorder=2.5)
plt.show()
```



1.3 The minimal model definition

We consider the yield Y to be a non-zero constant, with a simple variable change $\tilde{x} = \frac{x}{Y}$ we get

$$\begin{cases} \frac{\mathrm{d}s}{\mathrm{d}t} = -\mu(s)\tilde{x} + D(S_{\text{in}} - s) \\ \frac{\mathrm{d}\tilde{x}}{\mathrm{d}t} = \frac{1}{Y}\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{(\mu(s)-D)x}{Y} = (\mu(s) - D)\tilde{x} \end{cases}$$

since Y does not intervene in either equations, it merely is a scaling factor for $x(t)$ and has no effect on the dynamics of the studied model, we can simply set $Y = 1$, hence obtaining the *minimal model*:

$$\boxed{\begin{cases} \frac{\mathrm{d}s}{\mathrm{d}t} = -\mu(s)x + D(S_{\text{in}} - s) \\ \frac{\mathrm{d}x}{\mathrm{d}t} = (\mu(s) - D)x \end{cases}}$$

Now that we have defined the model, so without further ado, we can study its dynamics.

1.4 Model dynamics

1.4.1 Nullclines

Nullclines (zero-growth isoclines).

Biomass nullcline $\frac{\mathrm{d}x}{\mathrm{d}t} = 0$ We seek to find the states of the system (s, x) at which the biomass is invariant. Let'es denote,

$$I_x = \left\{ (s, x) \in \mathbb{R}_+^2 \,\middle|\, \frac{\mathrm{d}x}{\mathrm{d}t} = 0 \right\}$$

Expanding the expression for $\frac{\mathrm{d}x}{\mathrm{d}t}$

$$\begin{aligned} I_x &= \left\{ (s, x) \in \mathbb{R}_+^2 \,|(\mu(s) - D)x = 0 \right\} && (9) \\ &= \left\{ (s, x) \in \mathbb{R}_+^2 \,|x = 0 \right\} \cup \left\{ (s, x) \in \mathbb{R}_+^2 \,|\mu(s) - D = 0, D < \mu_{\max} \right\} && (10) \\ &= \left\{ (s, 0) \in \mathbb{R}_+^2 \right\} \cup \left\{ (s, x) \in \mathbb{R}_+^2 \,|\mu(s) = D < \mu_{\max} \right\} && (11) \\ &= \left\{ (s, 0) \in \mathbb{R}_+^2 \right\} \cup \left\{ (s, x) \in \mathbb{R}_+^2 \,\middle|\, s = \mu^{-1}(D), D < \mu_{\max} \right\} && (12) \\ &= \left\{ (s, 0) \in \mathbb{R}_+^2 \right\} \cup \left\{ (\mu^{-1}(D), x) \in \mathbb{R}^2 \,|D < \mu_{\max} \right\} && (13) \end{aligned}$$

The set $\{(s, 0)\}$ is the horizontal axis, which corresponds to no biomass ($x = 0$), this axis in logically invariant.

The set $\left\{(\mu^{-1}(D), x)\right\}$ corresponds to the vertical line that corresponds to the substrate concentration $s = \mu^{-1}(D)$ which is a well-defined value if $D < \mu_{\max}$ for any Monod-type function μ since a Monod-type function is monotonic (specifically: increasing) and positive $\forall s \geq 0$.

For the Monod function $\mu(s) = \frac{\mu_{\max} s}{k_s + s}$, we can easily find the exression for μ^{-1} by setting $\mu(s) = r \in [0, \mu_{\max}[$.

$$\mu(r) = \frac{r k_s}{\mu_{\max} - r}$$

The biomass nullcline is therefore the union of these two lines (if the vertical line is well-defined).

Substrate nullcline $\frac{\mathrm{d}s}{\mathrm{d}t} = 0$ Similarly as for the biomass, we aim to find the states (s, x) that leave the substrate concentration invariant.

$$\begin{aligned}
I_s &= \left\{ (s,x) \in \mathbb{R}^2_+ \,\middle|\, \frac{\mathrm{d}s}{\mathrm{d}t} = 0 \right\} && (14)\\
&= \left\{ (s,x) \in \mathbb{R}^2_+ \,|{-\mu(s)x + D(S_{\text{in}} - s) = 0} \right\} && (15)\\
&= \left\{ (s,x) \in \mathbb{R}^2_+ \,\middle|\, x = \frac{D(S_{\text{in}} - s)}{\mu(s)} \right\} && (16)\\
&= \left\{ \left(s, \frac{D(S_{\text{in}} - s)}{\mu(s)} \right) \in \mathbb{R}^2_+ \right\} && (17)
\end{aligned}$$

1.4.2 Equilibria

We search for the system's equilibria; points at which $\frac{\mathrm{d}s}{\mathrm{d}t} = 0$ and $\frac{\mathrm{d}x}{\mathrm{d}t} = 0$. The set of equilibrium points can be defined as $E = I_s \cap I_x$. Which corresponds geometrically to the intersection of the I_x nullclines with the I_s nullcline.

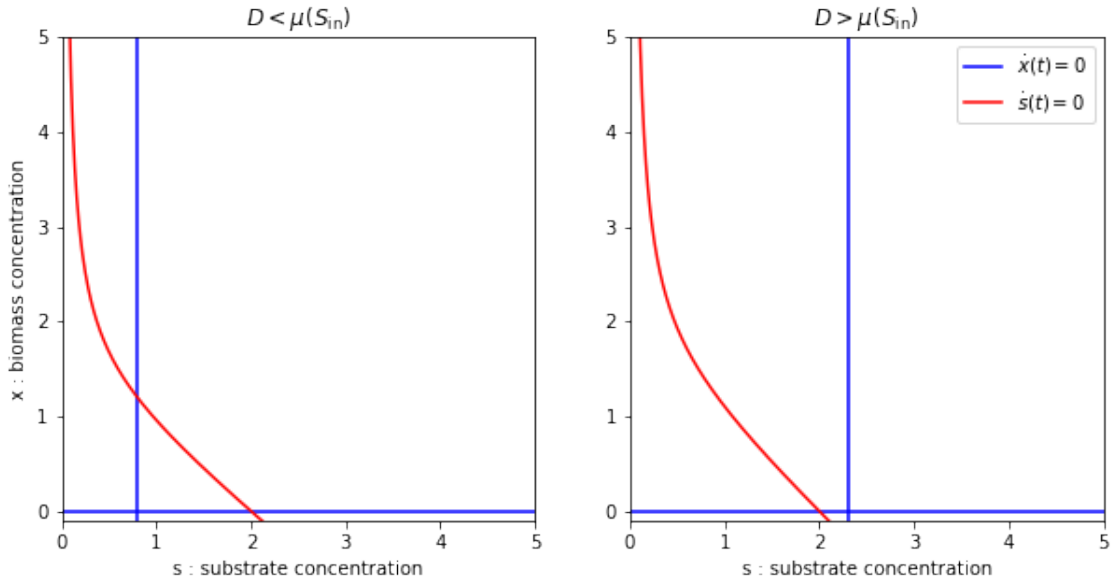
$$\begin{aligned}
E &= I_s \cap I_x && (18)\\
&= \left\{ \left(s, \frac{D(S_{\text{in}} - s)}{\mu(s)} \right) \in \mathbb{R}^2_+ \right\} \cap \left(\left\{ (s,0) \in \mathbb{R}^2_+ \right\} \cup \left\{ (\mu^{-1}(D), x) \in \mathbb{R}^2_+ \,|D < \mu_{\max} \right\} \right) && (19)\\
&= \left(\left\{ \left(s, \frac{D(S_{\text{in}} - s)}{\mu(s)} \right) \in \mathbb{R}^2_+ \right\} \cap \left\{ (s,0) \in \mathbb{R}^2_+ \right\} \right) \cup \left(\left\{ \left(s, \frac{D(S_{\text{in}} - s)}{\mu(s)} \right) \in \mathbb{R}^2_+ \right\} \cap \left\{ (\mu^{-1}(D), x) \in \mathbb{R}^2_+ \right\} \right) && (20)\\
&= \left\{ (s,0) \in \mathbb{R}^2_+ \,\middle|\, \frac{D(S_{\text{in}} - s)}{\mu(s)} = 0 \right\} \cup \left\{ \left(s, \frac{D(S_{\text{in}} - s)}{\mu(s)} \right) \in \mathbb{R}^2_+ \,\middle|\, s = \mu^{-1}(D) \right\} && (21)\\
&= \left\{ (s,0) \in \mathbb{R}^2_+ \,|{S_{\text{in}} - s = 0} \right\} \cup \left\{ \left(\mu^{-1}(D), \frac{D(S_{\text{in}} - \mu^{-1}(D))}{\mu(\mu^{-1}(D))} \right) \in \mathbb{R}^2_+ \right\} && (22)\\
&= \{ (S_{\text{in}}, 0) \} \cup \left\{ \left(\mu^{-1}(D), S_{\text{in}} - \mu^{-1}(D) \right) \middle| S_{\text{in}} - \mu^{-1}(D) > 0 \right\} && (23)\\
&= \underbrace{\{ (S_{\text{in}}, 0) \}}_{E_0} \cup \underbrace{\left\{ \left(\mu^{-1}(D), S_{\text{in}} - \mu^{-1}(D) \right) |\mu(S_{\text{in}}) > D \right\}}_{E_1} && (24)
\end{aligned}$$

While E_0 exists, and lays on the $x = 0$ nullcline, E_1 is empty if $\mu(S_{\text{in}}) < D$ as the equilibrium falls in the fourth quadrant ($x < 0$), which does not make sense as we only deal with positive values of biomass. In the case $\mu(S_{\text{in}}) = D$ we have $E_0 = E_1$.

Moreover, $\mu^{-1}(D)$ is only defined for $D < \mu_{\max}$. Hence, E_1 is non-empty for $D \in]\mu(S_{\text{in}}), \mu_{\max}[$.

[2]:
```
%run ../model.py
%run ../api.py
plt.rcParams["figure.figsize"]=10,10

fig, (ax1,ax2) = plt.subplots(1, 2)
for ax in [ax1, ax2]:
    ax.axis([-0, 5, -0.1, 5])
    ax.axes.set_aspect('equal')
    ax.set_xlabel('s : substrate concentration')

params.update(D=0.8)
plot_isocline_dx(ax1, **params)
plot_isocline_ds(ax1, **params)
ax1.set_title('$D<\\mu(S_\\mathrm{{in}})$')
ax1.set_ylabel('x : biomass concentration')

params.update(D=0.92)
plot_isocline_dx(ax2, **params)
plot_isocline_ds(ax2, **params)
ax2.set_title('$D>\\mu(S_\\mathrm{{in}})$')

plt.legend()
plt.show()
```

```
<Figure size 432x288 with 0 Axes>
```



1.5 Local stability around equilibria

Identifying equilibria is the first step in studying a non-linear system, as the study of the system around its equilibria allows a better understanding of its global behaviour.

In order to study its local behaviour around equilibria, we linearize the system with the help of its first order Taylor expansion.

Considering a system, defined by it's state equation $\frac{\mathrm{d}q}{\mathrm{d}t} = f(q,t)$ where $q(t)$ is the system's state vector at time t. Formally, let $U \subset \mathbb{R}^n$, we have

$$\begin{array}{rccc} f: & U \times \mathbb{R} & \to & \mathbb{R}^n \\ & (q,t) & \mapsto & \begin{pmatrix} f_1(q,t) \\ \vdots \\ f_n(q,t) \end{pmatrix} \end{array}$$

The linearized system around an equilibrium q_0 is given by the Taylor formula of f

$$f(q,t) = f(q_0,t) + \nabla_q f(q_0,t)(q-q_0) + o(\|q-q_0\|)$$

where $\nabla_q f$ is the Jacobian of f with respect to q, that is

$$\nabla_q f = \begin{pmatrix} \frac{\partial f_1}{\partial q_1} & \cdots & \frac{\partial f_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial q_1} & \cdots & \frac{\partial f_n}{\partial q_n} \end{pmatrix}$$

We can therefore describe the system $\forall q \in B_r(q_0)$ for a sufficiently small $r > 0$ where $B_r(q_0) = \{q \in \mathbb{R}^n \,|\, \|q-q_0\| < r\}$ is an open ball.

$$\frac{\mathrm{d}q}{\mathrm{d}t} \approx f(q_0,t) + J_f(q_0)(q(t)-q_0)$$

Since q_0 is an equilibrium, by definition $f(q_0,t) = 0, \forall t$, we can translate the system in order to take q_0 to the origin, let $\tilde{q}(t) = q(t) - q_0$, we have a linear system

$$\frac{\mathrm{d}\tilde{q}}{\mathrm{d}t} \approx J_f(q_0)\tilde{q}(t)$$

1.5.1 System Jacobian

In the case of the chemostat model

$$\begin{cases} \frac{\mathrm{d}s}{\mathrm{d}t} = f_1(s,x) = -\mu(s)x + D(S_{\text{in}} - s) \\ \frac{\mathrm{d}x}{\mathrm{d}t} = f_2(s,x) = (\mu(s) - D)x \end{cases}$$

$$J(s,x) = \begin{pmatrix} \frac{\partial f_1}{\partial s} & \frac{\partial f_1}{\partial x} \\ \frac{\partial f_2}{\partial s} & \frac{\partial f_2}{\partial x} \end{pmatrix} = \begin{pmatrix} -\mu'(s)x - D & -\mu(s) \\ \mu'(s)x & \mu(s) - D \end{pmatrix}$$

System around E_0

$$A = J(S_{\text{in}}, 0) = \begin{pmatrix} -D & -\mu(S_{\text{in}}) \\ 0 & \mu(S_{\text{in}}) - D \end{pmatrix}$$

We consider the system in a small open ball around $E_0 = (S_{\text{in}}, 0)$, let $\begin{pmatrix} \tilde{s} \\ \tilde{x} \end{pmatrix} = \begin{pmatrix} s \\ x \end{pmatrix} - \begin{pmatrix} S_{\text{in}} \\ 0 \end{pmatrix}$.

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{pmatrix} \tilde{s} \\ \tilde{x} \end{pmatrix} = A \begin{pmatrix} \tilde{s} \\ \tilde{x} \end{pmatrix}$$

The eigenvalues and eigenvectors of A allow studying the stability of the system around E_0. Moreover, the solution of the linear system is

$$\begin{pmatrix} \tilde{s}(t) \\ \tilde{x}(t) \end{pmatrix} = \exp(At) \begin{pmatrix} \tilde{s}(0) \\ \tilde{x}(0) \end{pmatrix}$$

and the exponential of a square matrix is easily obtained from the diagonalized matrix which has the eigenvalues on its diagonal.

For now, we will only study the eigenvalues, eigenvectors, and eigenspaces.

Eigenvalues and Eigenvectors Let $\lambda \in \mathbb{C}, v \in \mathbb{R}^n$, we say that λ is an eigenvalue of A and $v \in \mathbb{R}^n$ is an eigenvector of A associated to λ if and only if

$$Av = \lambda v$$

Or equivalently, $(A - \lambda Id)v = 0_{\mathbb{R}^n}$.

The eigenvalues are the roots of the characteristic polynomial $\chi_\lambda = \det(A - \lambda Id)$ and the associated eigenspace $S = \ker(A - \lambda Id)$.

Here,

$$\chi_\lambda = \det(A - \lambda Id) = \begin{vmatrix} -\mu'(s)x - D - \lambda & -\mu(s) \\ \mu'(s)x & \mu(s) - D - \lambda \end{vmatrix} = (-D - \lambda)(\mu(S_{\text{in}}) - D - \lambda) \implies \begin{cases} \lambda_1 = -D \\ \lambda_2 = \mu(S_{\text{in}}) - D \end{cases}$$

- $S_{A,1} = \ker(A - \lambda_1 Id) = \ker(A + DId) = \ker \begin{pmatrix} 0 & -\mu(S_{\text{in}}) \\ 0 & \mu(S_{\text{in}}) \end{pmatrix} = \ker \begin{pmatrix} 0 & -1 \\ 0 & 1 \end{pmatrix} = \text{span} \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\}$
- $S_{A,2} = \ker(A - \lambda_2 Id) = \ker(A - (\mu(S_{\text{in}}) - D)Id) = \ker \begin{pmatrix} -\mu(S_{\text{in}}) & -\mu(S_{\text{in}}) \\ 0 & 0 \end{pmatrix} = \ker \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} = \text{span} \left\{ \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right\}$

Stability of a linear system In the case of an n-dimensional linear system $\frac{\mathrm{d}q}{\mathrm{d}t} = f(q, t) = Aq(t)$, the origin is its sole equilibrium $f(q, t) = 0 \iff q(t) = 0_{\mathbb{R}^n}$.

- **Stable** if the system's state does not get further from the equilibrium with respect to its initial state $\forall t$.
- **Asymptotically stable** or **attractor** if the system's state converges to the equilibrium.
- **Unstable** if the system's state gets further from the equilibrium.

Naturally, if a linear system's state is at the origin, it stays at the origin, since $f(0_{\mathbb{R}^n}, t) = 0_{\mathbb{R}^n}, \forall t$. The remaining question is, whether for $q(0) \neq 0_{\mathbb{R}^n}$, the system's state would get closer/further from the origin, or maintain the same distance.

In the case of a one-dimensional linear system $\frac{\mathrm{d}q}{\mathrm{d}t} = aq(t)$ the state can be explicitly expressed as $q(t) = e^{at}q(0)$.

- If $a = 0$, then $q(t) = q(0), \forall t$ therefore the system is stable.
- If $a < 0$, then $\lim_{t\to\infty} q(t) = 0$ therefore the system is asymptotically stable.
- If $a > 0$, then $q(t)$ clearly diverges from the origin, therefore the system is unstable.

The same reasoning can be applied for an n-dimensional system, as $q(t) = e^{At}q(0)$, therefore the eigenvalues can help determine the stability at the equilibrium (the origin). Let $\rho(A)$ be the spectrum of A.

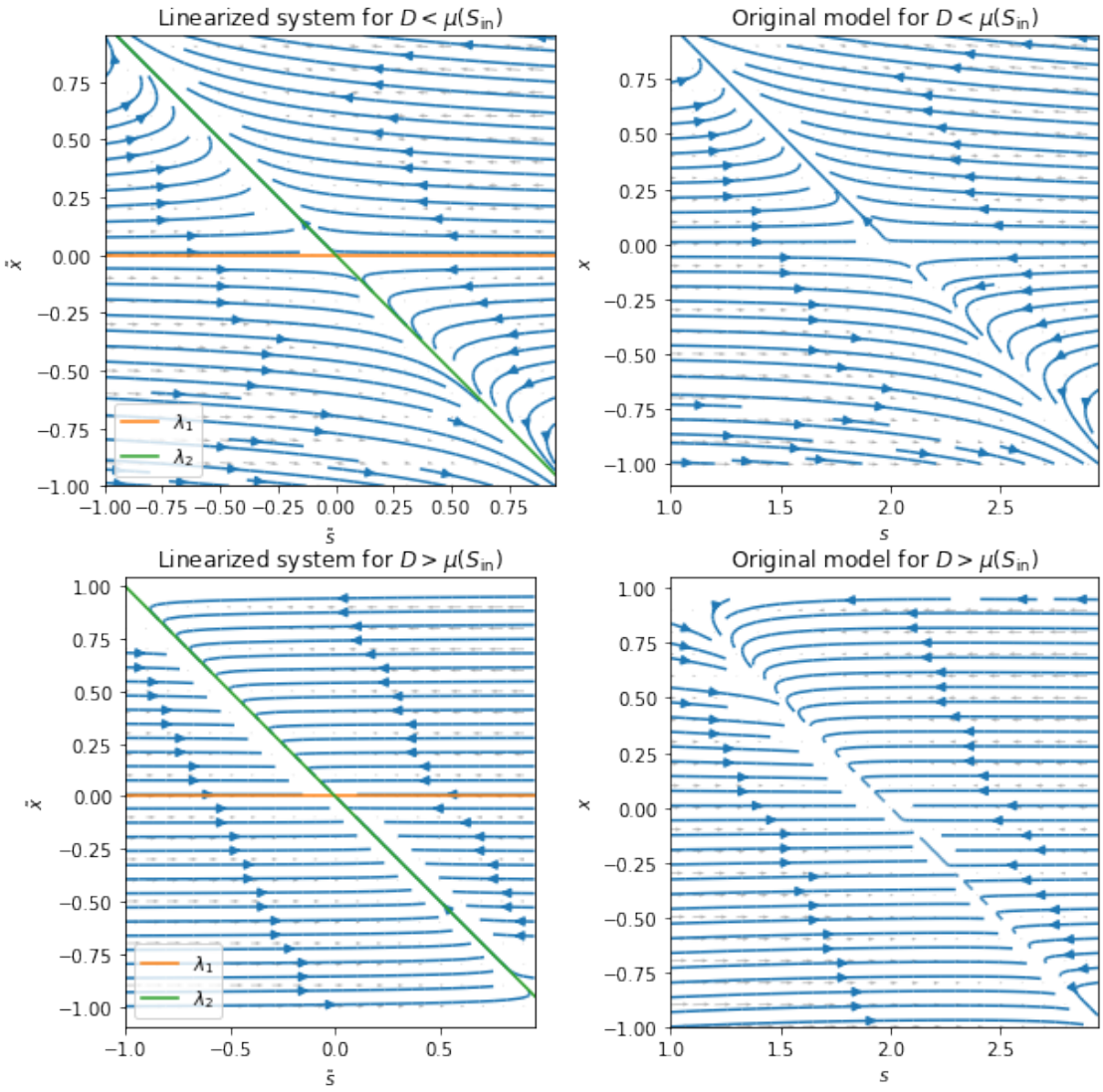
- If $\forall \lambda \in \rho(A), \mathrm{Re}(\lambda) \leq 0$ the system is stable.
- If $\forall \lambda \in \rho(A), \mathrm{Re}(\lambda) < 0$ the system is asymptotically stable.
- If $\exists \lambda \in \rho(A), \mathrm{Re}(\lambda) > 0$ the system is unstable.

Local stability around an equilibrium The local stability around an equilibrium is determined by the stability of the linearized system around that equilibrium. Since the Jacobian of the system at the equilibrium is the matrix of the linearized system, we study its spectrum to study its stability.

In the case of $E_0 = (S_{\text{in}}, 0)$ the spectrum of the Jacobian is $\rho = \{-D, \mu(S_{\text{in}}) - D\}$. Since $\lambda_1 = -D < 0$, the stability is solely determined by $\lambda_2 = \mu(S_{\text{in}}) - D$. - If $D < \mu(S_{\text{in}})$ then $\lambda_2 > 0$, then the system is locally unstable around E_0. - If $D > \mu(S_{\text{in}})$ then $\lambda_2 < 0$, then E_0 is a local attractor (i.e. points close enough to E_0 converge to E_0).

[3]:
```
f = growth_rate
E0 = equilibria(**params)[0]
size = 2
slim0, xlim0 = set_lims(size=size)
slim, xlim = set_lims(center=E0, size=size)
plot_params = dict(mesh=0.1, step=0.05, dF_params=params, scale=20, density=1,␣
↪alpha=0.3)

def A(s, x, D=1.0, Sin=2.0,  max=1.0, ks=0.2, **kw):
      = mu(Sin,  max= max, ks=ks)
    ds = -D*s - *x
    dx = ( -D)*x
    return ds, dx

fig, ax = plt.subplots(2, 2)

params.update(D=0.8)
phase_portrait(slim0, xlim0, A, ax=ax[0,0], **plot_params)
plot_eigenspaces([1,0], [1,-1], ax=ax[0,0])
phase_portrait(slim, xlim, f, ax=ax[0,1], **plot_params)
```

```
params.update(D=0.92)
phase_portrait(slim0, xlim0, A, ax=ax[1,0], **plot_params)
plot_eigenspaces([1,0], [1,-1], ax=ax[1,0])
phase_portrait(slim, xlim, f, ax=ax[1,1], **plot_params)

label = '{} for $D{}\\mu(S_\\mathrm{{in}})$'
ax[0,0].set_title(label.format('Linearized system','<'))
ax[0,1].set_title(label.format('Original model','<'))
ax[1,0].set_title(label.format('Linearized system','>'))
ax[1,1].set_title(label.format('Original model','>'))
for i in [0,1]:
    ax[i,0].legend()
    ax[i,0].set_xlabel('$\\tilde{s}$')
    ax[i,0].set_ylabel('$\\tilde{x}$')
    ax[i,1].set_xlabel('$s$')
    ax[i,1].set_ylabel('$x$')
plt.show()
```



We observe around E_0 two cases:

- Case $D < \mu(S_{\text{in}})$: the linearized system is unstable, as solutions have hyperbolic trajectories with $S_{A,1} = \text{span}\{(1,0)\}$ and $S_{A,2} = \text{span}\{(1,-1)\}$ as asymptotes for the hyperbolas, diverging along the $S_{A,2}$ axis, the eigenspace associated to the positive eigenvalue $\lambda_2 = \mu(S_{\text{in}}) - D > 0$.
- Case $D > \mu(S_{\text{in}})$: the linearized system is asymptotically stable (E_0 is an attractor), as solutions have parabolic trajectories with $S_{A,2}$ as an axis of symmetry for the parabolas, the eigenspace associated to $\lambda_1 = -D$ as $|\lambda_1| > |\lambda_2|$.

System around E_1 We remind that E_1 is non-empty only for $D < \mu(S_{\text{in}})$ with $E_1 = \{(\mu^{-1}(D), S_{\text{in}} - \mu^{-1}(D))\}$. Let's denote $(s_1, x_1) = (\mu^{-1}(D), S_{\text{in}} - \mu^{-1}(D))$, the linearized system

matrix is

$$B = J(s_1, x_1) = \begin{pmatrix} -\mu'(s_1)x_1 - D & -D \\ \mu'(s_1)x_1 & 0 \end{pmatrix}$$

Eigenvalues and Eigenvectors

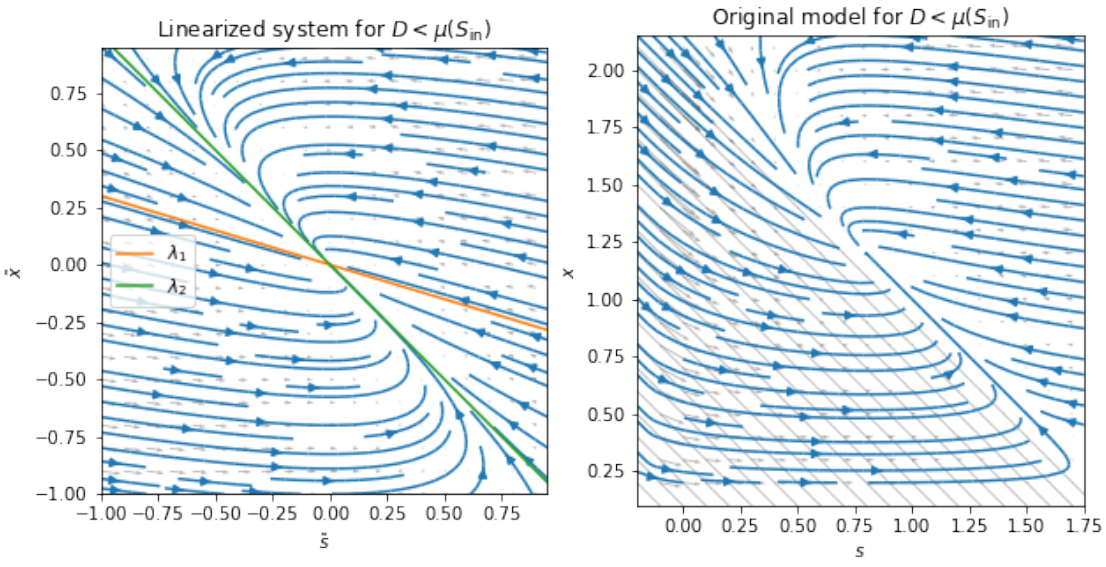
$$\chi_\lambda = \det(B - \lambda Id) = \begin{vmatrix} -\mu'(s_1)x_1 - D - \lambda & -D \\ \mu'(s_1)x_1 & -\lambda \end{vmatrix} = \lambda^2 + (\mu'(s_1)x_1 + D)\lambda + \mu'(s_1)x_1 D \implies \begin{cases} \lambda_1 = -D < 0 \\ \lambda_2 = -\mu'(s_1)x_1 < 0 \end{cases}$$

- $S_{B,1} = \ker(B - \lambda_1 Id) = \ker(B + DId) = \ker \begin{pmatrix} -\mu'(s_1)x_1 & -D \\ \mu'(s_1)x_1 & D \end{pmatrix} = \text{span}\left\{ \begin{pmatrix} D \\ -\mu'(s_1)x_1 \end{pmatrix} \right\}$
- $S_{B,2} = \ker(B - \lambda_2 Id) = \ker(B + (\mu'(s_1)x_1)Id) = \ker \begin{pmatrix} -D & -D \\ \mu'(s_1)x_1 & \mu'(s_1)x_1 \end{pmatrix} = \text{span}\left\{ \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right\}$

Stability around E_1 Since $\forall \lambda \in \rho(B), \lambda < 0$, E_1 is a local attractor.

[4]:
```
params.update(D=0.8)
E1 = equilibria(**params)[1]
size = 2
slim0, xlim0 = set_lims(size=size)
slim, xlim = set_lims(center=E1, size=size)
plot_params = dict(mesh=0.1, step=0.05, dF_params=params, scale=20, density=1,␣
 ↪alpha=0.3)

 x = E1[1] * mu_deriv(E1[0], **params)
def B(s, x, D=1.0, Sin=2.0,  max=1.0, ks=0.2, **kw):
    s1 = mu_inv(D,  max= max, ks=ks)
    x1 = Sin - s1
     x = x1 * mu_deriv(s1,  max= max, ks=ks)
    ds = -( x+D)*s -D*x
    dx =  x*s
    return ds, dx

fig, ax = plt.subplots(1, 2)

phase_portrait(slim0, xlim0, B, ax=ax[0], **plot_params)
plot_eigenspaces([params['D'],- x], [1,-1], t=[-size,size], ax=ax[0])
phase_portrait(slim, xlim, f, ax=ax[1], **plot_params)

label = '{} for $D{}\\mu(S_\\mathrm{{in}})$'
ax[0].set_title(label.format('Linearized system','<'))
ax[1].set_title(label.format('Original model','<'))
ax[0].legend()
ax[0].set_xlabel('$\\tilde{s}$')
ax[0].set_ylabel('$\\tilde{x}$')
ax[1].set_xlabel('$s$')
ax[1].set_ylabel('$x$')
```

```
plt.show()
```



E_1 is a singleton for $D < \mu(S_{\text{in}})$ if $D < \mu_{\max}$, in such case, the linearized system is asymptotically stable (E_1 is an attractor), same as before, having disctinct negative eigenvalues, the solutions have parabolic trajectories with the eigenspace associated to the larger eigenvalue in absolute value as the axis of symmetry for the parabolas, in the plot above $|\lambda_1| > |\lambda_2|$.

	$D < \mu(S_{\text{in}})$	$D > \mu(S_{\text{in}})$
E_0	unstable	attractor
E_1	attractor	undefined

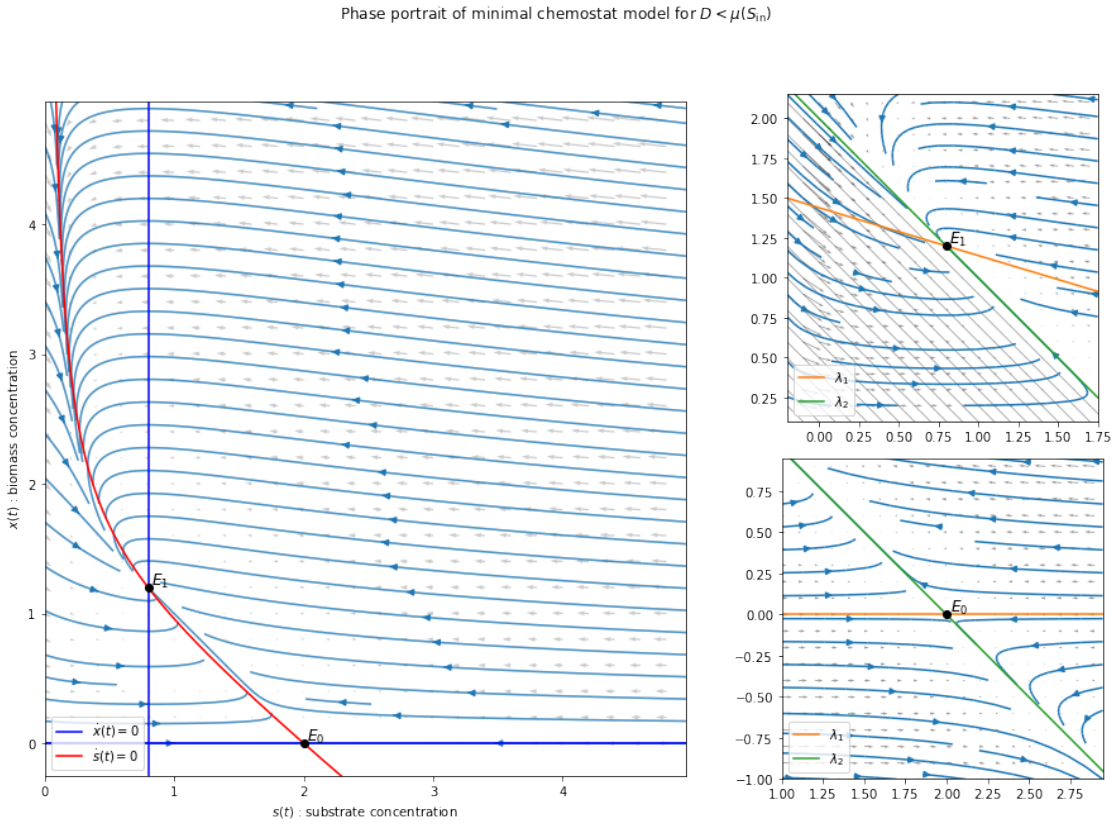
Summary for local stability

1.6 Global stability

Along with studying the linearized systems, plotting the phase portrait is a great way to study the global behaviour of a dynamical system.

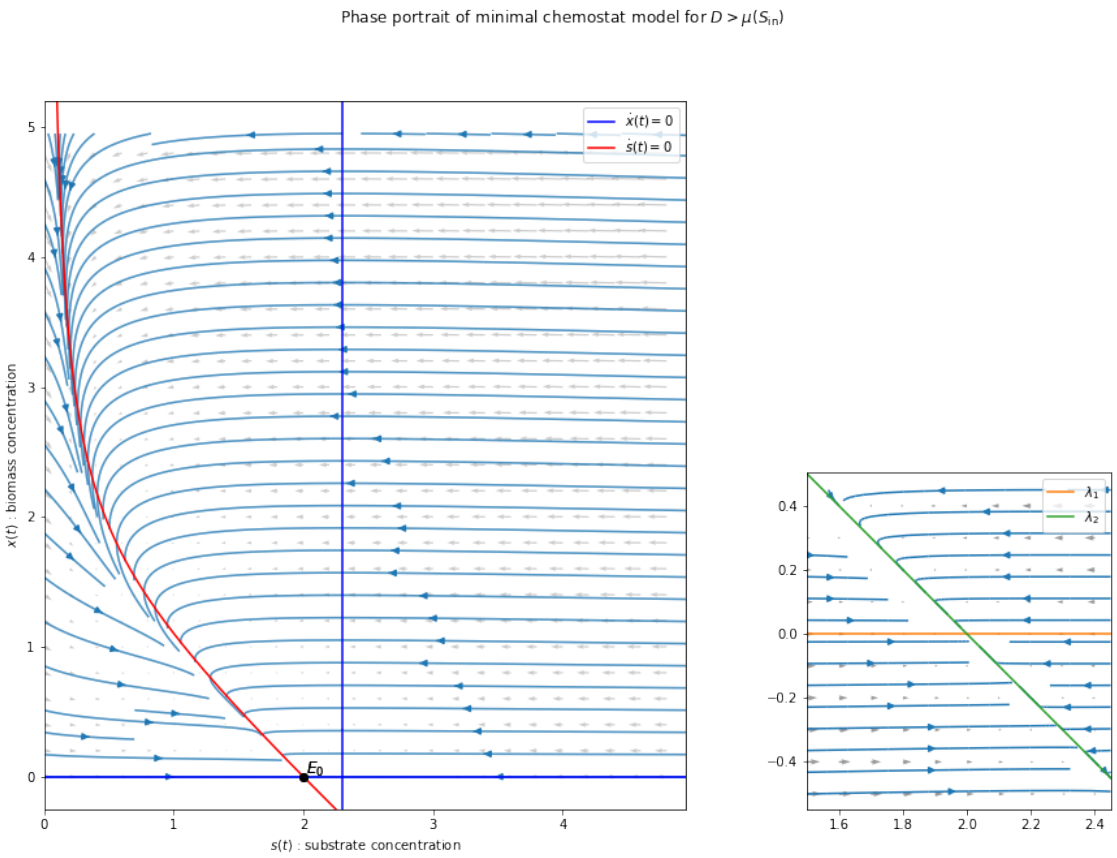
[5]:
```
from matplotlib import gridspec

title = 'Phase portrait of minimal chemostat model for␣
 ↪$D{}\\mu(S_\\mathrm{{in}})$'
params.update(D=0.8)
```

```
fig = plt.figure(figsize=(15,10))
gs = gridspec.GridSpec(2,2, width_ratios=(4,2), hspace=0.1)

ax = fig.add_subplot(gs[:,0])
phase_portrait((0,5), (0,5), growth_rate, mesh=0.2, step=0.05, dF_params=params,
               scale=30, density=(25, 1), title=title.format("<"), ax=ax,
               xlabel='$s(t)$ : substrate concentration',
               ylabel='$x(t)$ : biomass concentration')
plot_isocline_dx(ax, **params)
plot_isocline_ds(ax, **params)

E = equilibria(**params)
eq_text = ['$E_0$', '$E_1$']
plot_equilibria(E, name=eq_text, ax=ax)
ax.legend()

# equilibria plots
plot_params = dict(mesh=0.1, step=0.05, dF_params=params, scale=20, density=0.
 ↪5, alpha=0.4)
size = 2

ax0 = fig.add_subplot(gs[1,1])
slim, xlim = set_lims(center=E[0], size=size)
phase_portrait(slim, xlim, f, ax=ax0, **plot_params)
plot_eigenspaces([1,0], [1,-1], pt=E[0], t=[-size,size], ax=ax0)
plot_equilibria(E[:1], name=eq_text[:1], ax=ax0)
ax0.legend()

ax1 = fig.add_subplot(gs[0,1])
slim, xlim = set_lims(center=E[1], size=size)
phase_portrait(slim, xlim, f, ax=ax1, **plot_params)
plot_eigenspaces([params['D'],- x], [1,-1], pt=E[1], t=[-size,size], ax=ax1)
plot_equilibria(E[1:], name=eq_text[1:], ax=ax1)
ax1.legend()

plt.show()
```



[6]:
```
params.update(D=0.92)
fig = plt.figure(figsize=(15,10))
gs = gridspec.GridSpec(2,2, width_ratios=(4,2), hspace=0.1)

ax = fig.add_subplot(gs[:,0])
phase_portrait((0,5), (0,5), growth_rate, mesh=0.2, step=0.05, dF_params=params,
               scale=30, density=(25, 1), title=title.format(">"), ax=ax,
               xlabel='$s(t)$ : substrate concentration',
               ylabel='$x(t)$ : biomass concentration')
plot_isocline_dx(ax, **params)
plot_isocline_ds(ax, **params)
E = equilibria(**params)
plot_equilibria(E[:1], name=eq_text[:1], ax=ax)
ax.legend()

# equilibria plots
plot_params = dict(mesh=0.1, step=0.05, dF_params=params, scale=20, density=0.
 ↪5, alpha=0.4)
size = 1

ax0 = fig.add_subplot(gs[1,1])
```

```
slim, xlim = set_lims(center=E[0], size=size)
phase_portrait(slim, xlim, f, ax=ax0, **plot_params)
plot_eigenspaces([1,0], [1,-1], pt=E[0], t=[-2*size,2*size], ax=ax0)
plot_equilibria(E[:1], name=eq_text[:1], ax=ax)
ax0.legend()

plt.show()
```



1.7 Toward less minimalism

Now that we have studied the dynamics of the minimal model, we would like to add two terms to the equations :

- A decay term ($-k_d v x$) to the biomass growth equation to account for the natural death of microorganisms.
- A maintenance term ($-k_m v x$) to the substrate consumption equation to account for the substrate used for biomass sustenance.

$$\begin{cases} \frac{\mathrm{d}(vx)}{\mathrm{d}t} & = \mu(\cdot)vx - F_{\text{out}}x - k_d vx \\ \frac{\mathrm{d}(vs)}{\mathrm{d}t} & = -\frac{\mu(\cdot)}{Y}vx + F_{\text{in}}S_{\text{in}} - F_{\text{out}}s - k_m vx \end{cases}$$

Similarly to before, we can express these equations in terms of concentrations

$$\begin{cases} \frac{\mathrm{d}s}{\mathrm{d}t} = -\left(\frac{\mu(\cdot)}{Y} + k_m\right)x + D(S_{\text{in}} - s) \\ \frac{\mathrm{d}x}{\mathrm{d}t} = (\mu - D - k_d)x \end{cases}$$

We can now study the model dynamics

1.7.1 Nullclines

Biomass nullcline $\frac{\mathrm{d}x}{\mathrm{d}t} = 0$

$$I_x = \left\{(s,x) \in \mathbb{R}^2_+ \,\middle|\, \frac{\mathrm{d}x}{\mathrm{d}t} = 0\right\} = \left\{(s,x) \in \mathbb{R}^2_+ \,|(\mu(s) - D - k_d)x = 0\right\} = \left\{(s,0) \in \mathbb{R}^2_+\right\} \cup \left\{(\mu^{-1}(D+k_d), x) \in \mathbb{R}^2 \,|D -\right.$$

We have as in the minimal model $\{(s,0)\}$ the horizontal axis, which corresponds to no biomass ($x = 0$).

The set $\left\{(\mu^{-1}(D+k_d), x)\right\}$ corresponds to a vertical line defined if $D + k_d < \mu_{\max}$ for any Monod-type function.

The biomass nullcline is therefore the union of these two lines (if the vertical line is well-defined). The vertical line here, if it exists, is simply the translation of the vertical isocline from the minimal model to the right.

Substrate nullcline $\frac{\mathrm{d}s}{\mathrm{d}t} = 0$ Similarly as for the biomass, we aim to find the states (s,x) that leave the substrate concentration invariant.

$$I_s = \left\{(s,x) \in \mathbb{R}^2_+ \,\middle|\, \frac{\mathrm{d}s}{\mathrm{d}t} = 0\right\} = \left\{(s,x) \in \mathbb{R}^2_+ \,\middle|\, -\left(\frac{\mu(s)}{Y} + k_m\right)x + D(S_{\text{in}} - s) = 0\right\} = \left\{\left(s, \frac{D(S_{\text{in}} - s)}{\left(\frac{\mu(s)}{Y} + k_m\right)}\right) \in \mathbb{R}^2_+\right\}$$

1.7.2 Equilibria

$E = I_s \cap I_x = E_0 \cup E_1$ where

- $E_0 = \{(S_{\text{in}}, 0)\}$
- $E_1 = \left\{s, \left(s, \frac{D(S_{\text{in}}-s)}{\frac{\mu(s)}{Y}+k_m}\right) \middle| s = \mu^{-1}(D+k_d), \mu(S_{\text{in}}) > D + k_d\right\}$

While E_0 exists, and lays on the $x = 0$ nullcline. However, similarly to the minimal model equilibrium point, E_1 is empty if $\mu(S_{\text{in}}) < D + k_d$ as the equilibrium falls in the fourth quadrant ($x < 0$). In the case $\mu(S_{\text{in}}) = D + kd$ we have $E_0 = E_1$.

Moreover, $\mu^{-1}(D + k_d)$ is only defined for $D + k_d < \mu_{\max}$. Hence, E_1 is non-empty for $D + k_d \in]\mu(S_{\text{in}}), \mu_{\max}[$.

1.7.3 Stability around equilibria

We start by calculating the Jacobian of the system

$$\begin{cases} \frac{\mathrm{d}s}{\mathrm{d}t} = -\left(\frac{\mu(s)}{Y} + k_m\right) x + D(S_{\text{in}} - s) \\ \frac{\mathrm{d}x}{\mathrm{d}t} = (\mu(s) - D - k_d)x \end{cases}$$

$$J(s, x) = \begin{pmatrix} -\frac{\mu'(s)}{Y}x - D & -\frac{\mu(s)}{Y} - k_m \\ \mu'(s)x & \mu(s) - D - k_d \end{pmatrix}$$

System around E_0

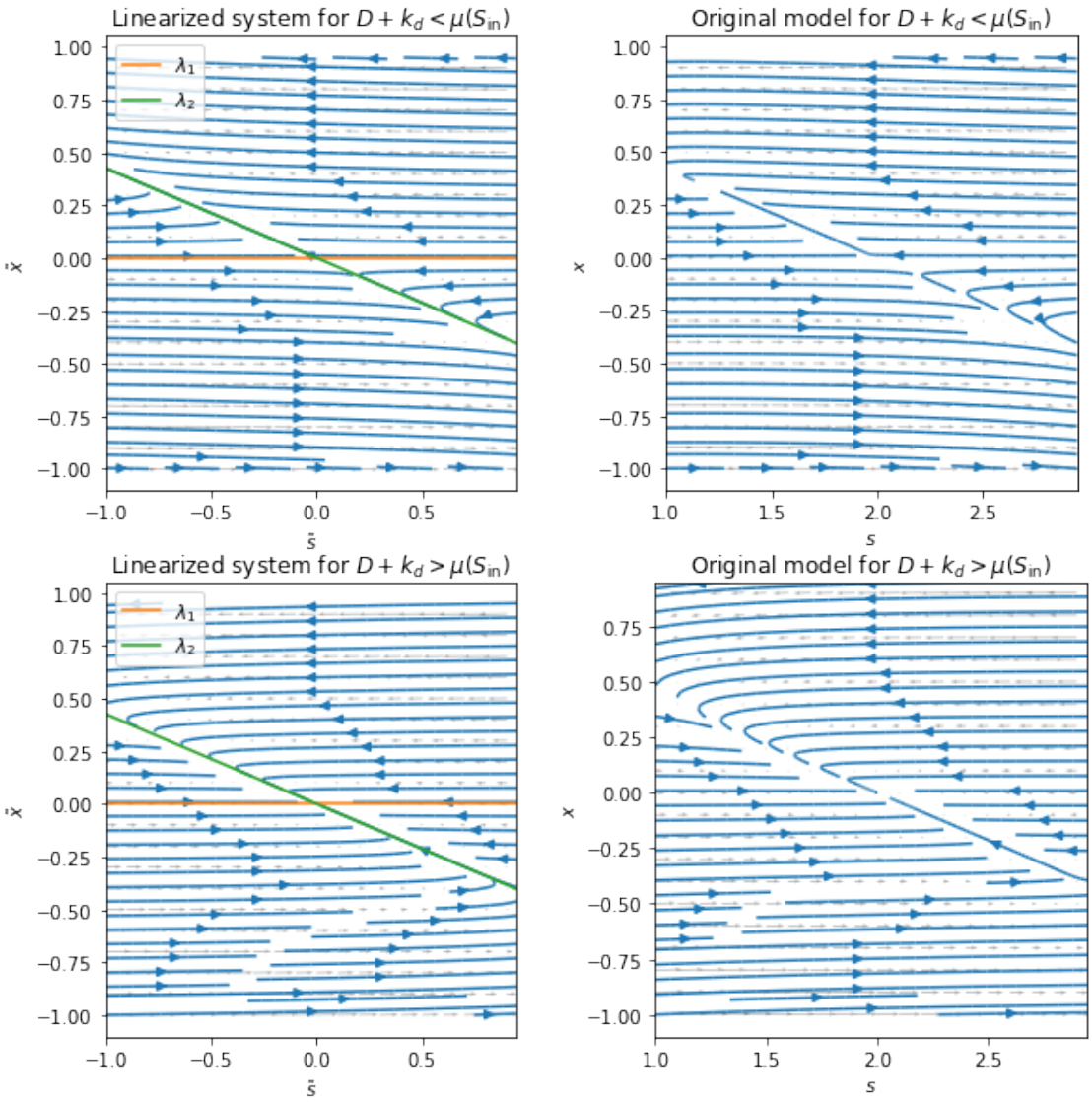
$$A = J(S_{\text{in}}, 0) = \begin{pmatrix} -D & -\frac{\mu(S_{\text{in}})}{Y} - k_m \\ 0 & \mu(S_{\text{in}}) - D - k_d \end{pmatrix}$$

$$\chi_\lambda = \det(A - \lambda Id) \implies \begin{cases} \lambda_1 = -D \\ \lambda_2 = \mu(S_{\text{in}}) - D - k_d \end{cases}$$

- $S_{A,1} = \ker(A - \lambda_1 Id) = \ker \begin{pmatrix} 0 & -\frac{\mu(S_{\text{in}})}{Y} - k_m \\ 0 & \mu(S_{\text{in}}) - k_d \end{pmatrix} = \text{span}\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\}$
- $S_{A,2} = \ker(A - \lambda_2 Id) = \ker \begin{pmatrix} -\mu(S_{\text{in}}) + k_d & -\frac{\mu(S_{\text{in}})}{Y} - k_m \\ 0 & 0 \end{pmatrix} = \text{span}\left\{ \begin{pmatrix} \frac{\mu(S_{\text{in}})}{Y} + k_m \\ -\mu(S_{\text{in}}) + k_d \end{pmatrix} \right\}$

[7]:
```
E0 = equilibria(**params)[0]
size = 2
slim0, xlim0 = set_lims(size=size)
slim, xlim = set_lims(center=E0, size=size)
plot_params = dict(mesh=0.1, step=0.05, dF_params=params, scale=20, density=1,␣
↪alpha=0.3)

def A(s, x, D=1.0, Sin=2.0,  max=1.0, ks=0.2, Y=1.0, km=0, kd=0):
      = mu(Sin,  max= max, ks=ks)
    ds = -D*s -( /Y + km)*x
    dx = ( -D-kd)*x
    return ds, dx

fig, ax = plt.subplots(2, 2)

params.update(Y=0.5, kd=0.05, km=0.2)
def S_A2(**kw):
```

```
      = mu(kw['Sin'], **kw)
    s =  /kw['Y'] + kw['km']
    x = - +kw['kd']
    return [s, x]

params.update(D=0.8)
phase_portrait(slim0, xlim0, A, ax=ax[0,0], **plot_params)
plot_eigenspaces([1,0], S_A2(**params), ax=ax[0,0])
phase_portrait(slim, xlim, f, ax=ax[0,1], **plot_params)

params.update(D=0.9)
phase_portrait(slim0, xlim0, A, ax=ax[1,0], **plot_params)
plot_eigenspaces([1,0], S_A2(**params), ax=ax[1,0])
phase_portrait(slim, xlim, f, ax=ax[1,1], **plot_params)

label = '{} for $D+k_d{}\\mu(S_\\mathrm{{in}})$'
ax[0,0].set_title(label.format('Linearized system','<'))
ax[0,1].set_title(label.format('Original model','<'))
ax[1,0].set_title(label.format('Linearized system','>'))
ax[1,1].set_title(label.format('Original model','>'))
for i in [0,1]:
    ax[i,0].legend()
    ax[i,0].set_xlabel('$\\tilde{s}$')
    ax[i,0].set_ylabel('$\\tilde{x}$')
    ax[i,1].set_xlabel('$s$')
    ax[i,1].set_ylabel('$x$')
plt.show()
```



System around E_1

$$B = J(s_1, x_1) = \begin{pmatrix} -\frac{\mu'(s_1)}{Y}x_1 - D & -\frac{D+k_d}{Y} - k_m \\ \mu'(s_1)x_1 & 0 \end{pmatrix}$$

$$\chi_\lambda = \det(B - \lambda Id) = \lambda^2 - \mathrm{tr}(B)\lambda + \det(B)$$

The added terms for decay and maintenance changes the behaviour around E_1. For certain values of k_d and k_m the eigenvalues of B are real distinct negative values. However, in other cases B can have complex conjugate eigenvalues whose real part is negative. We can make the conclusion that the system is asymptotically stable around E_1.

The implicit expressions for the eigenvalues are not simple, we can trace the phase portrait around E_1 to help visualize the system behaviour around E_1

[8]:
```
params.update(D=0.8,  max=1.5, Y=0.5, kd=0.05, km=0.2)
size = 2
slim0, xlim0 = set_lims(size=size)
plot_params = dict(mesh=0.1, step=0.05, dF_params=params, scale=20, density=1,␣
↪alpha=0.3)

def jacobian_E1(**params):
    x1 = mu_deriv(E1[0], **params) * E1[1]
    B = [
        [- x1/params['Y']-params['D'], -(params['D']+params['kd'])/
↪params['Y']-params['km']],
        [ x1, 0]
    ]
    return np.array(B)

def linear_E1(s, x, **params):
    B = jacobian_E1(**params)
    return B[0,0]*s + B[0,1]*x, B[1,0]*s + B[1,1]*x

fig, ax = plt.subplots(2, 2)

params.update(D=0.8)
E1 = equilibria(**params)[1]
phase_portrait(slim0, xlim0, linear_E1, ax=ax[0,0], **plot_params)
w, v = np.linalg.eig(jacobian_E1(**params))
plot_eigenspaces(v[:,0], v[:,1], ax=ax[0,0])
slim, xlim = set_lims(center=E1, size=size)
phase_portrait(slim, xlim, f, ax=ax[0,1], **plot_params)
print('for D = {}, we have  1 = {},  2 = {}'.format(params['D'], w[0], w[1]))

params.update(D=0.9)
E1 = equilibria(**params)[1]
phase_portrait(slim0, xlim0, linear_E1, ax=ax[1,0], **plot_params)
slim, xlim = set_lims(center=E1, size=size)
phase_portrait(slim, xlim, f, ax=ax[1,1], **plot_params)
w, v = np.linalg.eig(jacobian_E1(**params))
print('for D = {}, we have  1 = {},  2 = {}'.format(params['D'], w[0], w[1]))

label = '{} with {} eigenvalues'
ax[0,0].set_title(label.format('Linearized system','real'))
ax[0,1].set_title(label.format('Original model','real'))
ax[1,0].set_title(label.format('Linearized system','complex'))
ax[1,1].set_title(label.format('Original model','complex'))
for i in [0,1]:
```

```
    ax[i,0].set_xlabel('$\\tilde{s}$')
    ax[i,0].set_ylabel('$\\tilde{x}$')
    ax[i,1].set_xlabel('$s$')
    ax[i,1].set_ylabel('$x$')
ax[0,0].legend()
plt.show()
```

```
for D = 0.8, we have  1 = -1.7287746161513815,  2 = -1.1329797698135304
for D = 0.9, we have  1 = (-1.1649999999999996+0.3798354907061741j),  2 =
(-1.1649999999999996-0.3798354907061741j)
```