# MATHEMATICAL AND COMPUTER MODELING OF NONLINEAR BIOSYSTEMS I

## COMPUTER LABORATORY X: Kuznetsov model (solutions, behavior)

Ph. D. Programme 2013/2014

PhD STUDIES

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Project co-financed by European Union within the framework of European Social Fund

# Kuznetsov model

The dimensionless form of the model studied by Kuznetsov is as follows:

$$\begin{cases} \dot{x} = \sigma + \frac{\rho x y}{\eta + y} - \mu x y - \delta x\,, \\ \dot{y} = \alpha y(1 - \beta y) - x y\,, \end{cases} \tag{1}$$

where $x$ denotes the dimensionless density of effector cells (ECs) and $y$ stands for the dimensionless density of the population of tumor cells (TCs).

The model captures the **sneaking-trough mechanism**. Sneaking through refers to a situation in which for some initial level of TCs, when the initial level of ECs is sufficiently small, the state of tumor dormancy is achieved in the organism, but if the initial level of ECs is higher, then this initially high level of ECs decreases due to the small and constant level of TCs and, when the level of ECs is sufficiently small, the tumor cells start to proliferate and they break through the immune defense and successfully generate the tumor.

# Exercise 1 - model solution

Implement MATLAB function for solving the Kuznetsov model

$$\begin{cases} \dot{x} = \sigma + \frac{\rho x y}{\eta + y} - \mu x y - \delta x \,, \\ \dot{y} = \alpha y (1 - \beta y) - x y \,. \end{cases} \tag{2}$$

Requirements:

- 3 input arguments:
  1. time till which we want to have solution;
  2. model parameters in a structure;
  3. initial condition.
- directly return solution calculated by the ode solver.

# Exercise 1 - solution

```
function sol = solutionKuznetsov( Tend, par, init )

    opt = odeset('RelTol',1e-8,'AbsTol',1e-8);
    sol = ode45(@RHS,[0 Tend],init,opt);

    function dy = RHS(~,x)
       dy=zeros(2,1);
       dy(1)=par.sigma+par.rho*x(1)*x(2)/(par.eta+x(2))-...
             par.mu*x(1)*x(2)-par.delta*x(1);
       dy(2)=par.alpha*x(2)*(1-par.beta*x(2))-x(1)*x(2);
    end

end
```

# Exercise 2 - exemplary trajectories

Implement MATLAB function that plots multiple model trajectories in a phase plane. Use initial conditions for trajectories from the specified uniformly spaced grid.

Requirements:

- 6 input arguments:
  1. range for $x$ variable;
  2. range for $y$ variable;
  3. number of the interior points of mesh for both $x$ and $y$ variables;
  4. structure with parameters;
  5. variable indicating if we want to use logarithmic y-axis;
  6. figure number

# Exercise 2 - solution

```
function phasePortrait(xR,yR,N,par,log, fig)
    set(0,'DefaultAxesFontSize',18)
    figure(fig)
    clf
    hold on
    x = linspace(xR(1),xR(2),N+2);
    y = linspace(yR(1),yR(2),N+2);
    for xi = x(2:end-1)
        for yj = y(2:end-1)
            sol = solutionKuznetsov( 1000, par, [xi, yj]);
            plot(sol.y(1,:),sol.y(2,:),'r','LineWidth',1);
            plot(xi,yj,'Marker','o','MarkerFaceColor','k')
        end
    end
    xlabel('x - effector cells'); ylabel('y - tumor cells');
    xlim(xR); ylim(yR);
    if log
        set(gca,'YScale','log');
    end
    hold off
end
```

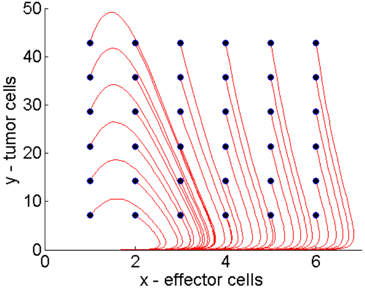
# Exercise 2 - solution (examplary evaluation)

```
clear all;

par.sigma = 0.318; par.rho = 1.131;
par.eta = 20.19; par.mu = 0.00311;
par.delta = 0.1908; par.alpha = 1.636;
par.beta = 0.002;

phasePortrait([0 7],[0 50],6,par,0,1)

par.delta = 0.545;
par.sigma = 0.318;
phasePortrait([0 3.5],[0 450],6,par,0,2)

par.delta = 0.545;
par.sigma = 0.182;
phasePortrait([0 3],[0 450],6,par,0,3)

par.delta = 0.545;
par.sigma = 0.073;
phasePortrait([0 4.5],[0 450],6,par,1,4)
```

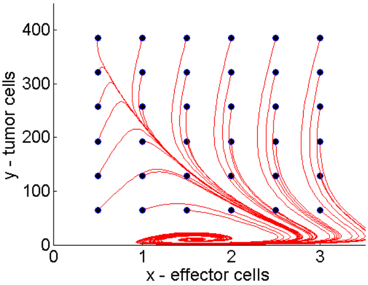
# Exemplary results - tumor dormancy

**Parameters:** $\rho = 1.131$; $\eta = 20.19$; $\mu = 0.00311$; $\alpha = 1.636$; $\beta = 0.002$;

$\delta = 0.1908$; $\sigma = 0.318$;



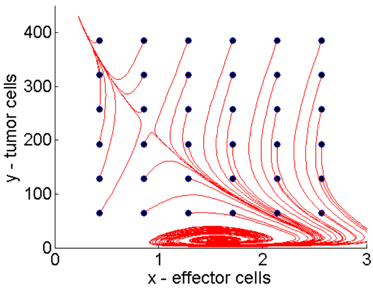

$\delta = 0.545$; $\sigma = 0.318$;



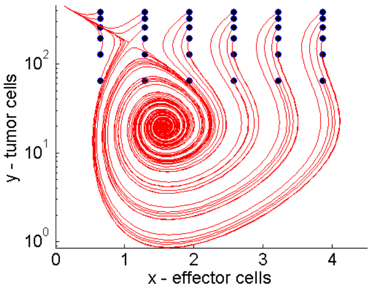
# Exemplary results - sneaking-trough mechanism

**Parameters:** $\rho = 1.131$; $\eta = 20.19$; $\mu = 0.00311$; $\alpha = 1.636$; $\beta = 0.002$;

$\delta = 0.545$; $\sigma = 0.182$;




$\delta = 0.545$; $\sigma = 0.073$;



# Exercise 3 - steady state

Implement MATLAB function that calculates steady states of the Kuznetsov model.

Extend the function from ex. 2 (trajectories in the phase plane) and plot steady states of the model.

Requirements:

- 1 input argument: structure with model parameters;

**Hint:** The $y$ coordinate of the positive steady state fulfils the equation $y^3\alpha\beta\mu + y^2\alpha(\beta(\delta + \eta\mu - \rho) - \mu) + y(\alpha(\delta(\beta\eta - 1) - \eta\mu + \rho) + \sigma) + \eta(\sigma - \alpha\delta) == 0$.

# Exercise 3 - solution

```
function S = steadyStates( par )
    S = [0; par.sigma/par.delta];

    %finding roots of polynomial a0+a1x+a2x^2+a3x^3
    a0 = par.eta*(-par.alpha*par.delta+par.sigma);
    a1 = par.alpha*(par.delta*(-1+par.beta*par.eta)-par.eta*par.mu+...
         par.rho)+par.sigma;
    a2 = par.alpha*(-par.mu+par.beta*(par.delta+par.eta*par.mu-...
         par.rho));
    a3 = par.alpha*par.beta*par.mu;

    R = roots([a3 a2 a1 a0]);
    R = R(R == real(R));
    R = R(R>=0);
    Xr = par.alpha*(1-par.beta*R);
    R = R(Xr>=0);
    Xr = Xr(Xr>=0);

    S = [S [R'; Xr']];
end
```

# Exercise 3 - solution

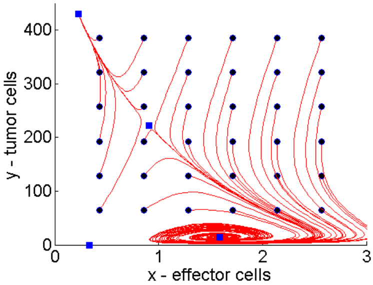
**Add the following lines to function from ex. 1**

```
S = steadyStates(par);
plot(S(2,:),S(1,:),'LineStyle','none',...
     'Marker','s','MarkerFaceColor','b');
```

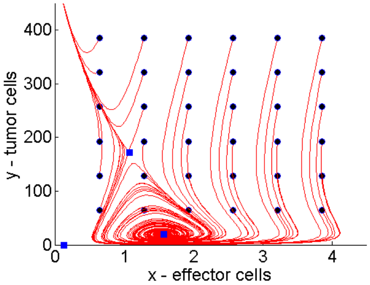
# Exercise 3 - exemplary results

**Parameters:** $\rho = 1.131$; $\eta = 20.19$; $\mu = 0.00311$; $\alpha = 1.636$; $\beta = 0.002$;

$\delta = 0.545$; $\sigma = 0.182$;




$\delta = 0.545$; $\sigma = 0.073$;



# Exercise 4 - local stability of steady states

Implement MATLAB function that checks if given steady states are locally asymptotically stable.

Extend the function from ex. 3 (trajectories in the phase plane) and differentiate stable and unstable steady states on the plot.

Requirements:

- 2 input arguments:
  1. structure with model parameters;
  2. coordinates of the steady states;

**Hint:** The Jacobian matrix of the system reads:

$$\begin{pmatrix} \frac{y\rho}{y+\eta} - y\mu - \delta & x\left(\frac{\eta\rho}{(y+\eta)^2} - \mu\right) \\ -y & -x - 2y\alpha\beta + \alpha \end{pmatrix}$$

# Exercise 4 - solution

```
function stab = stabilitySS( par, SS )

    stab = zeros(1,size(SS,2));
    for i=1:size(SS,2)
       J = Jacobian(par,SS(2,i),SS(1,i));
       E = eig(J);
       if all(real(E)~=0)
        stab(i) = any(real(E)>0)+2*(all(real(E)<0));
       end
    end

    function J = Jacobian(par,xs,ys)
        J = zeros(2,2);
        J(1,1)=-par.delta-ys*par.mu+par.rho*ys/(ys+par.eta);
        J(1,2)=xs*(-par.mu+par.eta*par.rho/(ys+par.eta)^2);
        J(2,1)=-ys;
        J(2,2)=-xs+par.alpha-2*ys*par.alpha*par.beta;
    end
end
```

# Excercise 4 - solution

**Add the following lines to function from ex. 1**

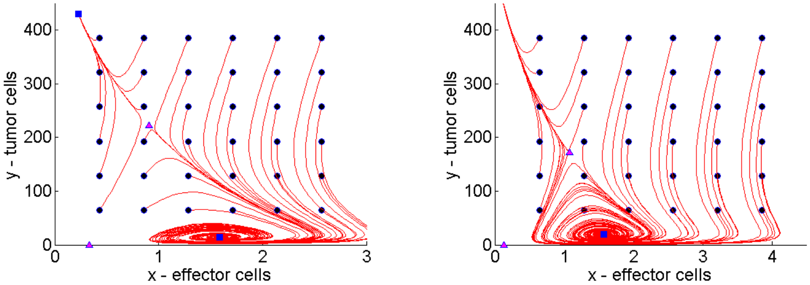
```
S = steadyStates(par);
Stab = stabilitySS(par,S);

plot(S(2,Stab==1),S(1,Stab==1),'LineStyle','none',...
     'Marker','^','MarkerFaceColor','m');
plot(S(2,Stab==2),S(1,Stab==2),'LineStyle','none',...
     'Marker','s','MarkerFaceColor','b');
plot(S(2,Stab==3),S(1,Stab==3),'LineStyle','none',...
     'Marker','?','MarkerFaceColor','r');
```

# Exercise 4 - exemplary results

**Parameters:** $\rho = 1.131$; $\eta = 20.19$; $\mu = 0.00311$; $\alpha = 1.636$; $\beta = 0.002$;

$\delta = 0.545$; $\sigma = 0.182$;

$\delta = 0.545$; $\sigma = 0.073$;



# Exercise 5 - convergence of the trajectory

Implement MATLAB function that for given initial condition determines to which steady state corresponding trajectory converges.

Extend the function from ex. 4 and plot the trajectories tending to different steady states in different colors.

Requirements:

- 4 input arguments:
  1. initial condition for the trajectory;
  2. structure with model parameters;
  3. steady states coordinates;
  4. vector indicating type of stability of the steady states.

# Exercise 5 - solution

```
function which = convergence( init, par, SS, stab )
 opt = odeset('Events',@events);
 sol = ode45(@RHS,[0 Inf],init,opt);
 which = sol.ie(end);

 function [value, isterminal, direction] = events(~,y)
  value = sqrt((y(1) - SS(2,:)).^2+(y(2) - SS(1,:)).^2)'-0.1;
  isterminal = (stab == 2);
  direction = zeros(size(value));
 end

 function dy = RHS(~,x)
  dy=zeros(2,1);
  dy(1)=par.sigma+par.rho*x(1)*x(2)/(par.eta+x(2))-...
        par.mu*x(1)*x(2)-par.delta*x(1);
  dy(2)=par.alpha*x(2)*(1-par.beta*x(2))-x(1)*x(2);
 end
end
```

# Exercise 5 - solution

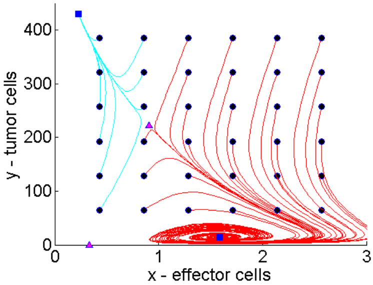
**Add/modify the following lines to function from ex. 4**

```
S = steadyStates(par);
Stab = stabilitySS(par,S);

cls = jet(size(S,2));

x = linspace(xR(1),xR(2),N+2);
y = linspace(yR(1),yR(2),N+2);
for xi = x(2:end-1)
    for yj = y(2:end-1)
        sol = solutionKuznetsov( 1000, par, [xi, yj]);
        which = convergence( [xi, yj], par, S, Stab );
        plot(sol.y(1,:),sol.y(2,:),'r','LineWidth',1,...
             'Color',cls(which,:));
        plot(xi,yj,'Marker','o','MarkerFaceColor','k')
    end
end
```

# Exercise 5 - exemplary results

**Parameters:** $\rho = 1.131$; $\eta = 20.19$; $\mu = 0.00311$; $\alpha = 1.636$; $\beta = 0.002$;

$\delta = 0.545$; $\sigma = 0.182$;




$\delta = 0.545$; $\sigma = 0.073$;