MATHEMATICAL AND COMPUTER MODELING OF NONLINEAR BIOSYSTEMS I

COMPUTER LABORATORY IX: Models with various types of antigens

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

Basic model of response of immune system to antigen

Basic model of interaction between the antigen (V) and effector (E) is the following

$$\begin{cases} \dot{V} = g(V) - \gamma VE \\ \dot{E} = f(V) - \nu VE \end{cases} \tag{1}$$

where all parameters are positive, $f(V) = \eta V$ and

- $g(V) = \beta V$ — for active antigens (bacteria, viruses);
- $g(V) = -\beta V$ — for passive antigens (toxins);
- $g(V) = \beta - \delta V$ — for autoantigens (autoimmune diseases, transplantations).

The particular formulas for $f(V)$ and $g(V)$ are arbitrary and they don't incorporate every possible aspect of the immune response, e.g. one should incorporate carrying capacity for the antigen ($g(V) = \beta V(1 - V/K)$).

Excercise 1 - model implementation

Implement MATLAB function for solving the system

$$\begin{cases} \dot{V} = g(V) - \mu VE \\ \dot{E} = f(V) - \nu VE \end{cases} \tag{2}$$

where functions g, f are passed as a function parameter.

Requirements:

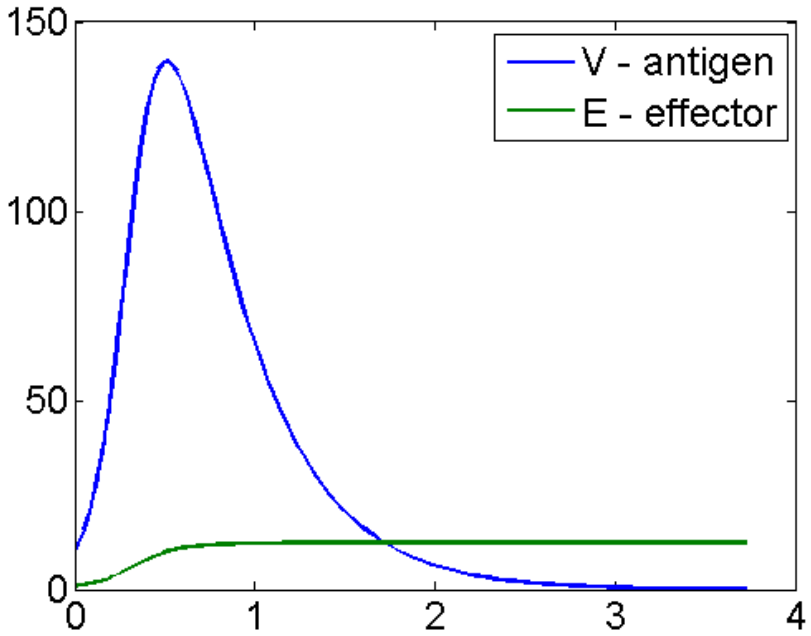
- All parameters and initial conditions should be passed in a single structure.
- Solver should stop if the level of antigen (V) reaches certain level (upper and lower, both supplied as a parameter).
- Function should return the variable indicating if solver stopped prematurely and which limit was reached.

Excercise 1 - solution

```
function [sol, flag] = solveModel( Tspan, par)
    opt = odeset('Events',@events,'RelTol',1e-8,'AbsTol',1e-8);
    sol = ode45(@RHS, Tspan, par.init, opt);
    flag = 0;
    if ~isempty(sol.ie)
       flag = sol.ie;
    end

    function y = RHS(~,x)
       y = zeros(2,1);
       y(1) = feval(par.g,x(1),par)-par.gamma*x(1)*x(2);
       y(2) = feval(par.f,x(1),par)-par.nu*x(1)*x(2);
    end
    function [val, isterminal, direction] = events(~,x)
       val = x(1) - [par.tresholdD, par.tresholdU];
       direction = [-1 1];
       isterminal = [1 1];
    end
end
```

Plot of exemplary solution

```
par.beta = 10; par.eta = 0.5; par.gamma = 1; par.nu = 0.04;
par.g = @(V,par)(par.beta*V); par.f = @(V,par)(par.eta*V);
par.init = [10; 1]; par.tresholdD = 0.1; par.tresholdU = 1e5;

sol = solveModel([0 Inf],par);
plot(sol.x,sol.y,'LineWidth',2)
legend({'V - antigen','E - effector'},'Location','NorthEast');
```



New immunotherapy type

Nowadays it is possible to collect and culture patients effector cells (e.g. T Cells).

After intensive culture we may reintroduce those cells into the patients body.

That type of therapy is intensively studied in case of cancer and there were few cases with spectacular disease regression after the treatment.

Excercise 2 - simulating immunotherapy

Implement MATLAB function that simulates immunotherapy based on reintroduction of cultured effectors. Assume that each injection is instantaneous, i.e. at each injection moment stop the solver, modify the initial condition and start solving again.

Use function from ex. 1.

Requirements

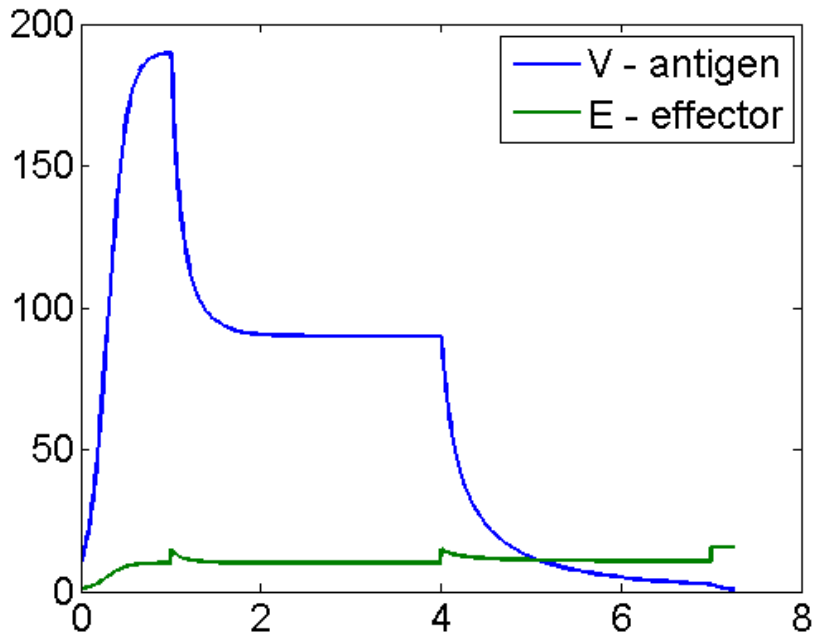
- Input parameters: terminal time for the solution, moments of effectors injection, number of injected effectors, model parameters;
- Output parameters: solution, variable indicating if solver stopped because soltuion reached certain upper/lower limit (see ex. 1).

Excercise 2 - solution

```
function [sol, flag] = solveVaccinations(Tend,moments,dose,par)
    sol.x = 0;
    sol.y = par.init;
    ti = 0;

    for i=unique([moments(moments<Tend) Tend])
        [solTmp, flag] = solveModel([ti i],par);
        sol.x = [sol.x solTmp.x(2:end)];
        sol.y = [sol.y solTmp.y(:,2:end)];

        if ~isempty(solTmp.ie)
            break;
        end

        par.init = solTmp.y(:,end);
        par.init(2) = par.init(2) + dose;
        ti = i;
    end
end
```

Plot of exemplary solution

```
par.beta = 10; par.eta = 0.5; par.gamma = 1; par.nu = 0.05;
par.g = @(V,par)(par.beta*V); par.f = @(V,par)(par.eta*V);
par.init = [10; 1]; par.tresholdD = 0.1; par.tresholdU = 1e5;

sol = solveVaccinations(11,1:3:12,5,par);
plot(sol.x,sol.y,'LineWidth',2)
legend({'V - antigen','E - effector'},'Location','NorthEast');
```



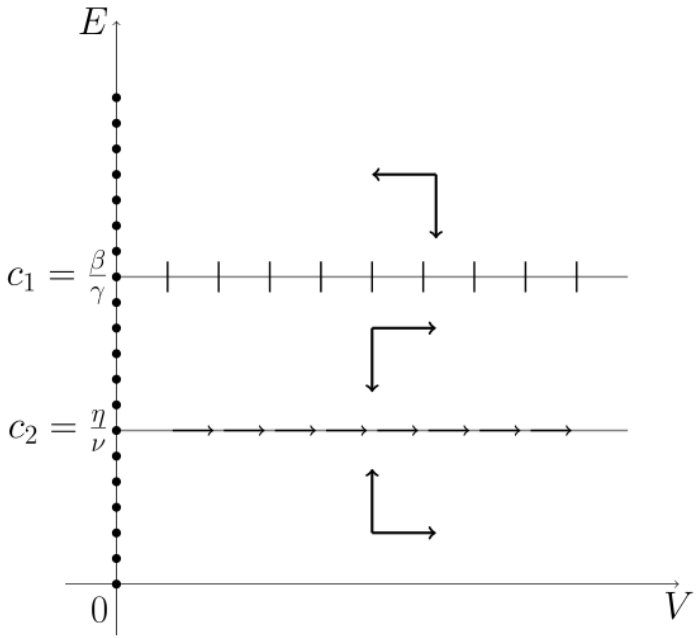
Mathematical properties of active antigen model

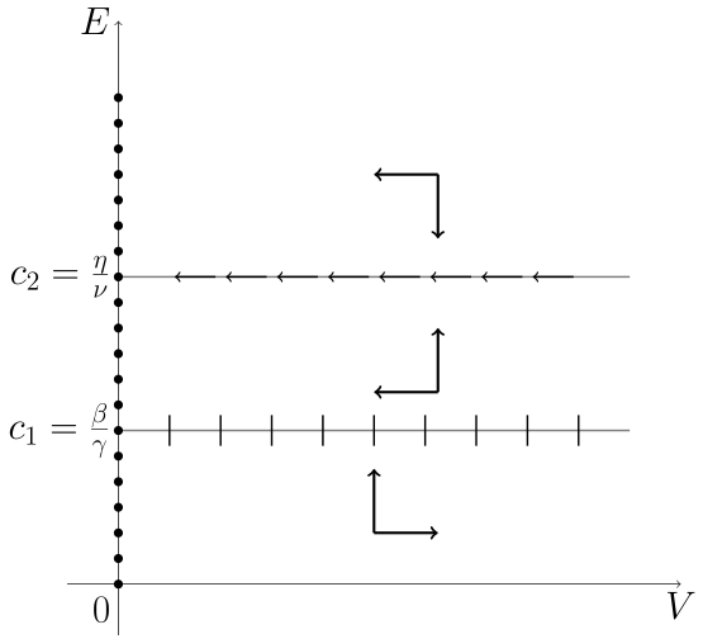
Let us define $c_1 = \beta/\gamma$ and $c_2 = \eta/\nu$.

From the lecture we know that if $c_1 < c_2$ or $E(0) > c_1 > c_2$ and

$$V(0) \leqslant \frac{\gamma}{\nu}\left(E(0) - c_1 + (c_1 - c_2)\ln\frac{E(0) - c_2}{c_1 - c_2}\right)$$

then every solution tends to a steady state with zero value of antigen (V=0), i.e. antigen is completely removed from the system.





Exercise 3 - checking if disease free state is acheivable

Implement MATLAB function which for given parameters values and initial conditions returns true if and only if the corresponding solution will reach disease free state.

Conditions once again:
If $c_1 < c_2$ or $E(0) > c_1 > c_2$ and
$V(0) \leqslant \frac{\gamma}{\nu}\left(E(0) - c_1 + (c_1 - c_2)\ln\frac{E(0)-c_2}{c_1-c_2}\right)$.

Exercise 3 - solution

```
function selfCure = condition( par )
    c1 = par.beta/par.gamma;
    c2 = par.eta/par.nu;

    selfCure = (c1<c2) || (par.init(2)> c2 &&...
                c1 < c2 && ...
                par.init(1) <= F(par));

    function y = F(par)
       y = par.gamma/par.nu*(par.init(1)-c1+...
           (c1-c2)*log((par.init(1)-c2)/(par.c1-par.c2)));
    end

end
```

Finding minimal dose

Assume that we know exact parameters values for a patient.

We can inject effector cells only once (let us assume that on the next day).

Question 1: What is the minimal number of effector cells that we need to inject in order to get the number of antigens under the specified threshold?

Question 2: What is the dependence of the minimal dose on the parameter ν?

Exercise 4 - finding minimal dose

Implement MATLAB function that for given model parameters and injection moment will return the minimal number of effector cells that we need to inject in order to get the number of antigens under the specified threshold (healthy state).

Input parameters: terminal time for the solution, moment of effectors injection, model parameters.

Use the function from ex. 3.

Exercise 4 - solution

```
function dose = findMinimalDose( Tend, moment, par)
    dose = 0;
    if condition(par)
        warning('No vaccination needed!');
    else
        dose = Inf; doseTmp = 2; step = 1; tol = 1e-5;
        while step>tol
        [~, flag] = solveVaccinations(Tend,moment,doseTmp,par);
            if flag == 1 %cured
                dose = doseTmp; doseTmp = dose - step;
            else
                if dose < Inf
                    step = step/2; doseTmp = dose - step;
                else
                    step = step*2; doseTmp = doseTmp + step;
                end
            end
        end
    end
```

Dependence of minimal dose on parameter ν

```
par.beta = 10;
par.eta = 0.5;
par.gamma = 1;
par.g = @(V,par)(par.beta*V);
par.f = @(V,par)(par.eta*V);
par.init = [10; 1];
par.tresholdD = 0.1;
par.tresholdU = 1e5;

nuV = 0.03:0.01:0.15;
out = zeros(1,length(nuV));
for i = 1:length(nuV)
    par.nu = nuV(i);
    display(nuV(i));
    out(i) = findMinimalDose(14, 1, par);
end

plot(nuV,out)
```

Dependence of minimal dose on ν