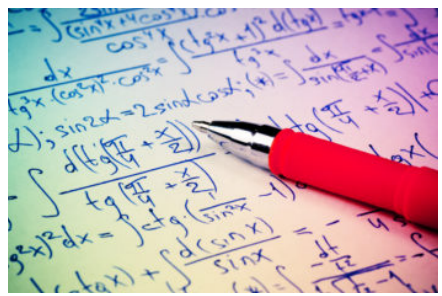
Numeričko rješavanje običnih diferencijalnih jednadžbi u Matlabu



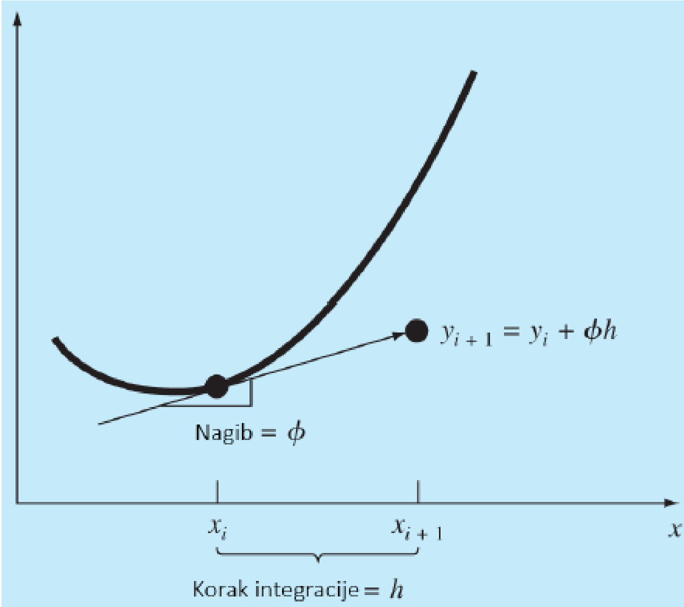
Željka Ujević Andrijić
Sveučilište u Zagrebu
Fakultet kemijskog inženjerstva i tehnologije

zujevic@fkit.unizg.hr

Rješavanje običnih diferencijalnih jednadžbi (ODJ)

Opća ODJ prvog reda: $dy/dx = f(x,y)$

- Diskretiziramo područje (interval) nezavisne varijable u kojem tražimo rješenje.
- Tražimo način kako procijeniti sljedeću točku na intervalu diskretizacije.



Nova vrijednost = stara vrijednost + nagib **· korak**

Rekurzivni zapis: $\boldsymbol{y_{i+1} = y_i + \phi h}$

- Vrijednosti od kojih krećemo zadane su **početne** vrijednosti.
- **Kako odrediti nagib (prvu derivaciju) $\boldsymbol{\phi}$?**

Eulerova metoda

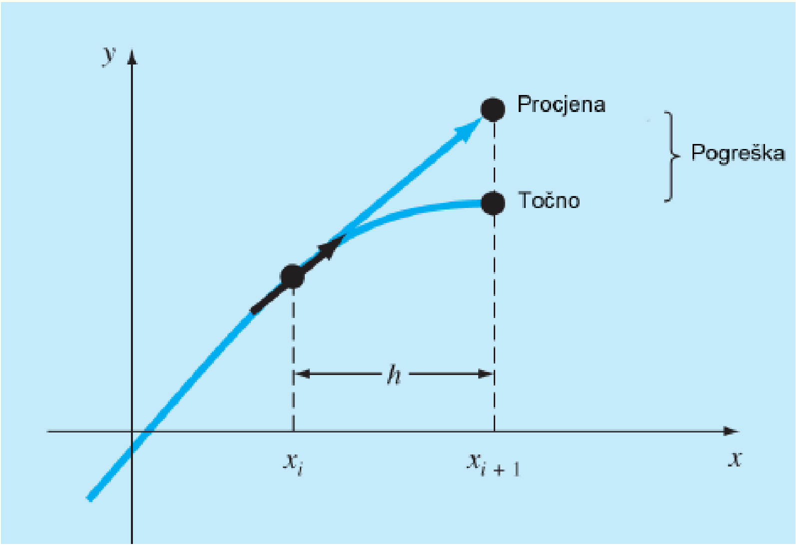
Najjednostavnije:

$$\phi = f(x_i, y_i)$$

Funkcija se aproksimira na intervalu (x_i, x_{i+1}) s nagibom na početku intervala.

$$y_{i+1} = y_i + h \cdot f(x_i, y_i)$$

Eulerova metoda



Runge Kutta metode

OPĆI OBLIK: $y_{i+1} = y_i + \underbrace{\phi f(x_i, y_i, h)}_{\text{Funkcija prirasta}} \cdot h$

$$\phi = a_1 k_1 + a_2 k_2 + \dots + a_n k_n$$

$$\begin{aligned}
k_1 &= f(x_i, y_i) \\
k_2 &= f(x_i + p_1 h, y_i + q_{11} k_1 h) \\
k_3 &= f(x_i + p_2 h, y_i + q_{12} k_1 h + q_{22} k_2 h) \\
&. \\
&. \\
&. \\
k_n &= f(x_i + p_{n-1} h, y_i + q_{n-1,1} k_1 h + q_{n-1,2} k_2 h + \dots q_{n-1,n-1} k_{n-1} h)
\end{aligned}$$

k - koeficijenti nagiba
n – red metode
a_i, p_i, q_i - konstante

Konstante se određuju izjednačavanjem s Taylorovim redom.

Runge Kutta metode

Red	Naziv	Oblik	k
n=1	Eulerova metoda	$y_{i+1} = y_i + k_1 h$	$k_1 = f(x_i, y_i)$
n=2	Heune a_2=1/2	$y_{i+1} = y_i + \left(\frac{1}{2}k_1 + \frac{1}{2}k_2\right)h$	$k_1 = f(x_i, y_i)$ $k_2 = f(x_i + h, y_i + k_1 h)$
	Heune (poligonalna) a_1=1	$y_{i+1} = y_i + k_2 h$	$k_1 = f(x_i, y_i)$ $k_2 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1 h)$
	Ralston a_2=2/3	$y_{i+1} = y_i + \left(\frac{1}{3}k_1 + \frac{2}{3}k_2\right)h$	$k_1 = f(x_i, y_i)$ $k_2 = f(x_i + \frac{3}{4}h, y_i + \frac{3}{4}k_1 h)$
n=3	Runge Kutta 3. reda	$y_{i+1} = y_i + \frac{1}{6}(k_1 + 4k_2 + k_3)h$	$k_1 = f(x_i, y_i)$ $k_2 = f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1 h)$ $k_3 = f(x_i + h, y_i - k_1 h + 2k_2 h)$
n=4	Runge Kutta 4. reda	$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)\,\mathrm{h}$	$k_1 = f(x_i, y_i)$ $k_2 = f\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1 h\right)$ $k_3 = f\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_2 h\right)$ $k_4 = f(x_i + h, y_i + k_3 h)$

Adaptivne metode rješavanja ODJ

- Za razliku od prethodnih metoda u kojima je korak integracije *h* konstantan kod adaptivnih metoda ***h* se može mijenjati u svakom koraku**, pa jednokoračnu metodu možemo pisati u obliku:

$$y_{i+1} = y_i + h_i\,\phi(x_i, y_i, h_i)$$

- Određuje se duljina koraka h_i tako da bude postignuta unaprijed zadana točnost → potrebno je procijeniti lokalnu pogrešku uslijed diskretizacije.

- Pogreška se procjenjuje iz razlike predikcija dobivenih iz dviju metoda:

  1. Iz razlike dviju RK metoda istog reda, ali uz različite korake integracije: **Adaptivna RK metoda**

  2. Iz razlike između dvije RK metode različitog reda (npr. 5. i 4. reda) **Runge-Kutta Fehlberg metoda)**

- *U programskim alatima postoje **ugrađene funkcije** za rješavanje ODJ!*

Adaptivne metode rješavanja ODJ

- **ode23** primjenjuje RK 2. i 3. reda za rješavanje ODJ **i podešavanje koraka integracije**
- **ode45** primjenjuje RK 4. i 5. reda za rješavanje ODJ **i podešavanje koraka integracije**. Preporuča se da se rješavanje počinje s ovom funkcijom!
- **ode113** je funkcija s višekoračnim solverom.

Sintaksa **ode** naredbe:

```
[t, y] = ode45(@odefun, tspan, y0)
```

y: niz rješenja, gdje je svaki stupac jedna varijabla,
a svaki red odgovara vremenu u t vektoru

odefun: funkcija

tspan: vremenski raspon na kojemu se traži rješenje

y0: vektor početnih vrijednosti (početni uvjet)

Koraci rješavanja ODJ u Matlabu

1. **Kreiranje funkcije** u Matlabu
   u koju se upisuje diferencijalna jednadžba.

2. **Unošenje konstantnih vrijednosti**
   iz ODJ u funkciju ili m-skriptu.

3. **Specificiranje inicijalnih vrijednosti** zavisnih varijabli i **područja** nezavisnih varijabli unutar kojeg se traži rješenje.

4. **Primjena *solvera*** (funkcija ode, Euler ili Runge Kutta metoda) za rješavanje obične diferencijalne jednadžbe.

Primjer rješavanja običnih diferencijalnih jednadžbi

Primjer:

Izradite program za numeričko rješavanje linearne diferencijalne jednadžbe I. reda s konstantnim koeficijentima **Eulerovom** metodom, metodom **Runge-Kutta IV** i naredbom **ode45**.

$$y' + y = \sin(x) + 0.5$$

uz početni uvjet: $y_{(0)} = 0,5$

Program u Matlabu:

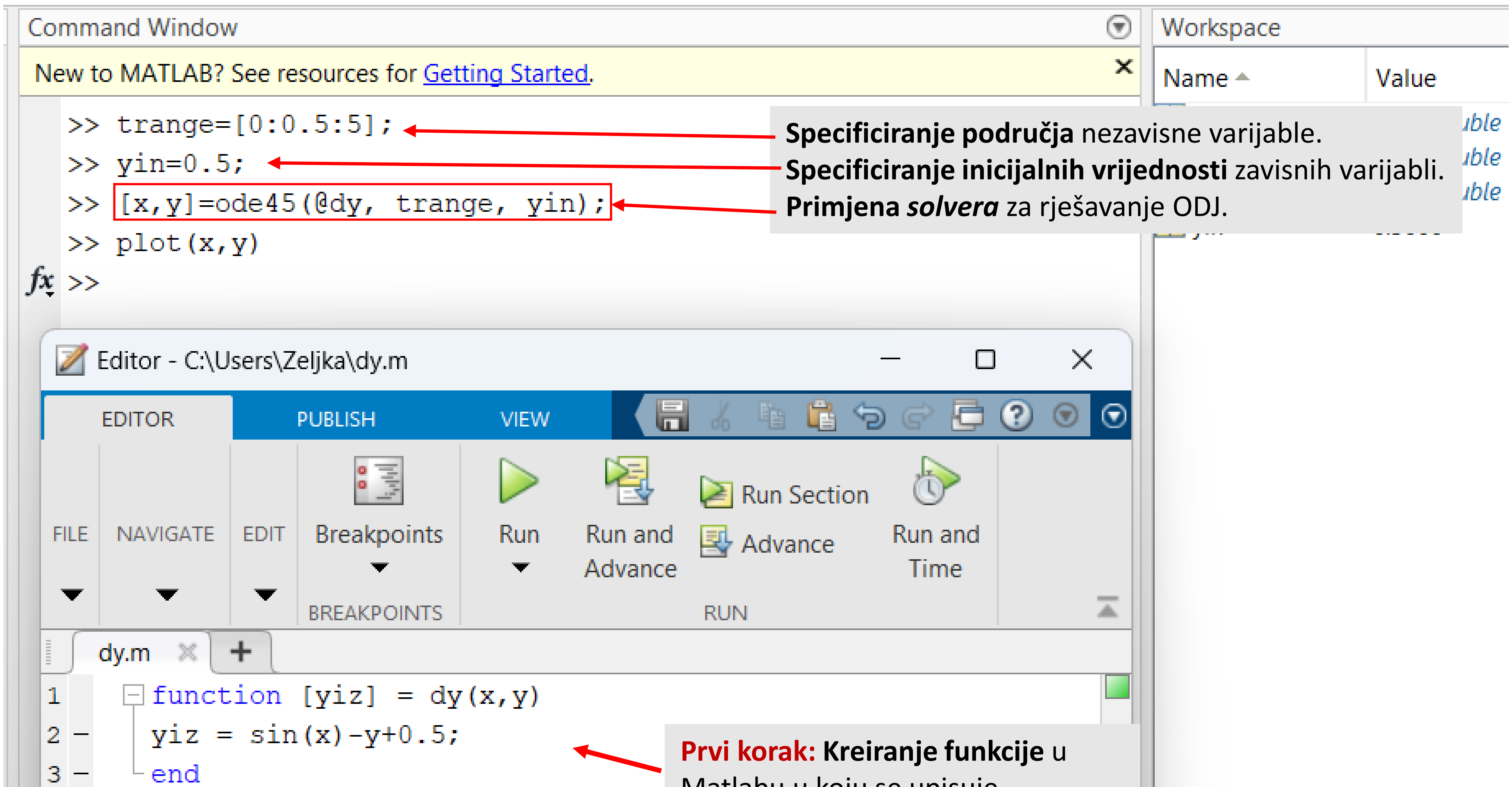
```
clear all ; clc ;  close all

x1=0;     % pocetna tocka
y1_0=0.5; % pocetni uvjet
xmax=5;   % zavrsna tocka
bkr=10;   % broj koraka
h=(xmax-x1)/bkr; % korak integracije
y1_rk=y1_0; % pocetna vrijednost za RK-IV
y1_e=y1_0;  % pocetna vrijednost za Euler
xmax=x1+h*bkr;
br=0;
```

Primjer rješavanja ODJ

```
for x=x1:h:xmax
    br=br+1;
    xc(br)=x;
    yrk(br)=y1_rk;    % RK-IV
    y1e(br)=y1_e;     % Euler

    y1_e = y1_e + h*dy(x,y1_e); % Euler

    k1 = h*dy(x,y1_rk);  % RK-IV algoritam
    k2 = h*dy(x+h/2,y1_rk+k1/2);
    k3 = h*dy(x+h/2,y1_rk+k2/2);
    k4 = h*dy(x+h,y1_rk+k3);
    y1_rk = y1_rk + (k1+2*k2+2*k3+k4)/6;

   fprintf('\nx=%.5f\ty_RK4=%.5f\ty_E=%.5f\t%.3f%%', x,y1_rk,y1_e)
end
fprintf('\n');

[xx, yy] = ode45('dy',[x1 xmax],y1_0);  % naredba ode45
%[xx, yy] = ode45(@dy,[x1 xmax],y1_0);  % u novijim verzijama

plot(xc,yrk,'-x',xc,y1e,'-o',xx,yy,'-r')
xlabel('x'); ylabel('y');
legend('R-K-IV','Euler','ode45');
grid on
```

```
function yiz = dy(x, y)
% Definicija funkcije dy
 yiz = sin(x) - y + 0.5;
end
```

Primjer rješavanja ODJ – preko naredbe ode45



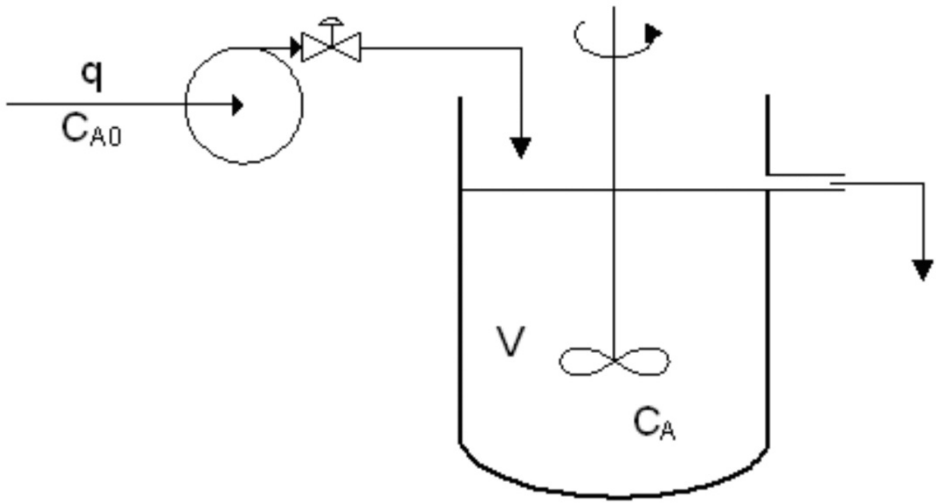
Specificiranje područja nezavisne varijable.
Specificiranje inicijalnih vrijednosti zavisnih varijabli.
Primjena *solvera* za rješavanje ODJ.

Prvi korak: Kreiranje funkcije u Matlabu u koju se upisuje diferencijalna jednadžba.

Primjer: Protočno kotlasti reaktor

Zadatak:

Odredite prijelazni odziv koncentracije u protočno kotlastom reaktoru, c_A



Zadani podaci:

q = 0,085 m^3/min - protok kroz reaktor

V = 2,1 m^3 - volumen reaktora

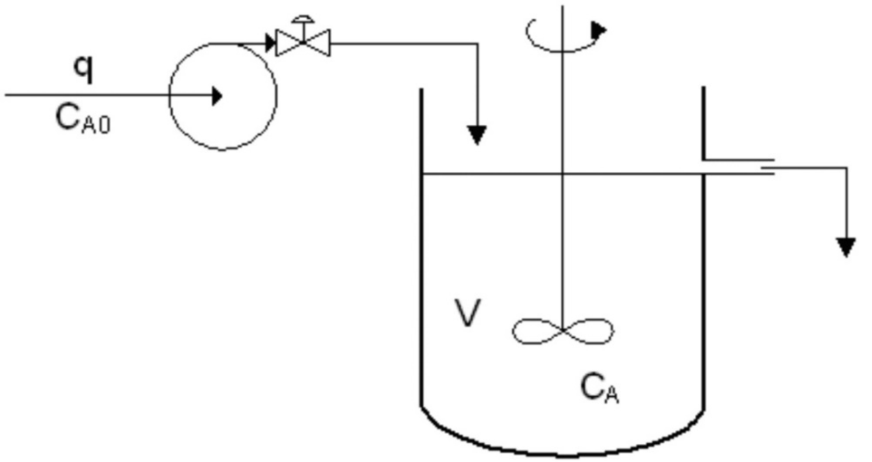
c_{A0} = 1,85 mol/m^3 - koncentracija tvari A u ulaznoj struji

$c_{A,poč}$ = 0 mol/m^3 - početna koncentracija tvari A u reaktoru

Primjer: Protočno kotlasti reaktor (PKR)

Bilanca množine tvari: $\dfrac{dn_A}{dt} = \dot{n}_d - \dot{n}_o$

$q_{dov} = q_{odv} \Rightarrow V = \text{konst}$



$$V \cdot \frac{dc_A}{dt} = q \cdot c_{A0} - q \cdot c_A$$

$$\frac{V}{q} \cdot \frac{dc_A}{dt} = c_{A0} - c_A$$

$$\frac{dc_A}{dt} = (c_{A0} - c_A)\frac{q}{V}$$

Treba riješiti ovu diferencijalnu jednadžbu.

Primjer: PKR – Primjena Euler-ove metode

```
 clear ; clf ; axis ('auto'); clc
 %   Zadani podaci iz procesa
V = 2.1;             % volumen reaktora  1 [m3]
q = 0.2;             % protok [m3/min]
ca0 = 1.85;          % koncentracija na ulazu [mol/m3]

%   PODACI ZA SIMULACIJU
delt = 0.5;          % korak integracije
tstart = 0;
tend = 120;          % trajanje procesa

%   POCETNI UVJETI I VEKTORI ZA SPREMANJE
ca1 = 0;            % [mol/m3]
n = round(tend/delt);       % 70/0.1 = 700 vrijeme (broj vrem. koraka)

%  SIMULACIJA
for cnt = 1:n

%  NUMERICKA SIMULACIJA, Euler
   ca1dot = q/V*(ca0 - ca1);        % dca1 = ...
   ca1 = ca1 + delt * ca1dot;

%  Pohrana rezultata za graficki prikaz (u obliku matrica)
   CA1(cnt) = ca1;
   CA0(cnt)= ca0;
   t(cnt) = cnt*delt;

end

% odziv Ca1
plot (t,CA1)
xlabel ('vrijeme [min]')
ylabel ('Ca1 [mol/m3]')
title ('Primjer PKR')
```

Primjer: PKR – preko naredbe ode45

D:\Zeljka\Documents\MATLAB\PKR.m

```
function [ f ] = PKR( t,ca1 )
V=2.1;
q=0.2;
ca0=1.85;
f=q/V*(ca0-ca1);
end
```

Kreiranje funkcije u Matlabu u koju se upisuje diferencijalna jednadžba.

Unošenje konstantnih vrijednosti iz ODJ u funkciju.

Command Window

New to MATLAB? Watch this Video, see Demos, or read Getting Started.

```
>> trange=[0:0.5:120];
>> ca1in=0;
>> [t,ca1]=ode45(@PKR, trange, ca1in);
>> plot(t,ca1)
>>
```

Workspace

Stack: Base | Select data to plot

Name	Value	Min	Max
ca1	<241x1 double>	0	1.8500
ca1in	0	0	0
t	<241x1 double>	0	120
trange	<1x241 double>	0	120

Specificiranje područja nezavisne varijable.
Specificiranje inicijalnih vrijednosti zavisnih varijabli.
Primjena *solvera* za rješavanje ODJ.

Primjer: Hlađenje kapljevine u spremniku pomoću okolnog zraka

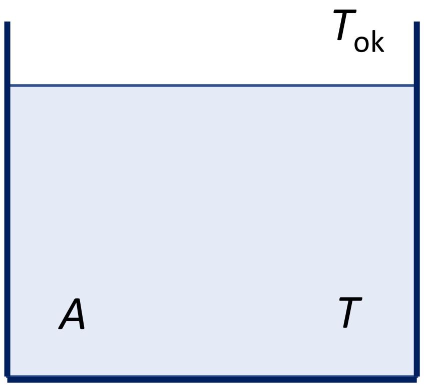
Fizikalni model hlađenja u spremniku

Prema **zakonu očuvanja energije** i **Newtonovom zakonu hlađenja**, toplinski tok:

$$Q = h\,A\,(T - T_{ok})$$

$$\boxed{m c_p \frac{dT}{dt} = -\,h\,A\,(T - T_{ok})}$$

$$\frac{dT}{dt} = -\frac{h\,A}{m c_p}(T - T_{ok})$$



Zadane vrijednosti:

Parametar	Vrijednost	Jedinica
h	50	W/m²·K (konvekcija)
A	1.2	m² (površina spremnika)
m	4	kg
c_p	4180	J/kg·K (voda)
T_0	80	°C (početna)
T_{ok}	24	°C (okolina)

Program u Matlabu:

```
close all
clf
h = 10;                  % W/m^2K
A = 1.5;                 % m^2
m = 12;                  % kg
cp = 4180;               % J/kgK (za vodu)
T_ok = 24;               % °C, temperatura okoline
T0 = 80;                 % °C, početna temperatura u spremniku

% Vrijeme simulacije
tspan = [0 15000];  % simulacija (u sekundama)

% Rješavanje diferencijalne jednadžbe
[t, T] = ode45(@(t, T) -(h*A/(m*cp))*(T - T_ok), tspan, T0);

% Prikaz rezultata
figure;
plot(t/60, T);
xlabel('Vrijeme [min]');
ylabel('Temperatura [°C]');
title('Hlađenje spremnika');
grid on;
```

$$\frac{dT}{dt} = -\frac{h\,A}{mc_p}\,(T - T_{ok})$$