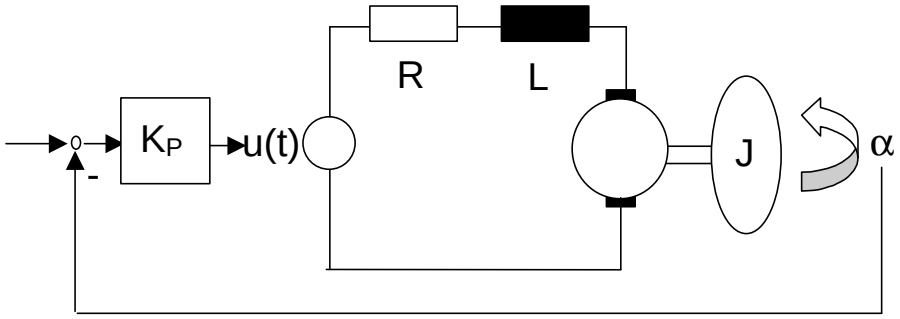
Exercise 5:

Model of a position control

Position control is used in many application of mechatronic systems e. g. for robots. In this exercise a model of a robot control system is given. This model must be implemented using Matlab/Simulink.



The equations are:

$$u(t) = R \cdot i_m(t) + L \cdot \frac{d}{dt}\left(i_m(t)\right) + u_m(t);$$
$$u_m(t) = Q \cdot \dot{a}_m(t);$$
$$J_m \cdot \ddot{a}_m(t) = M_i(t) - D \cdot \dot{a}_m(t);$$
$$M_i(t) = K \cdot i_m(t);$$

The voltage $u(t)$ is the input of the system, $a_M(t)$ the output. The system will be controlled by a proportional-controller. The parameters are (without units):

L = 0.23e-3;
K = 23.4e-3;
R = 2.4;
Q = 0.0234;
J = 0.23e-6;
D = 0.4191e-5;

a) Build up a simulation model of the model and test it!

b) Change the parameters of the controller to get a good control behavior!

c) Try different solvers (use additionally solvers for stiff systems) and compare the results! Plot the simulated motor-current, calculated with different solver in the same plot!

Document and print your work using word.

Exercise 4:

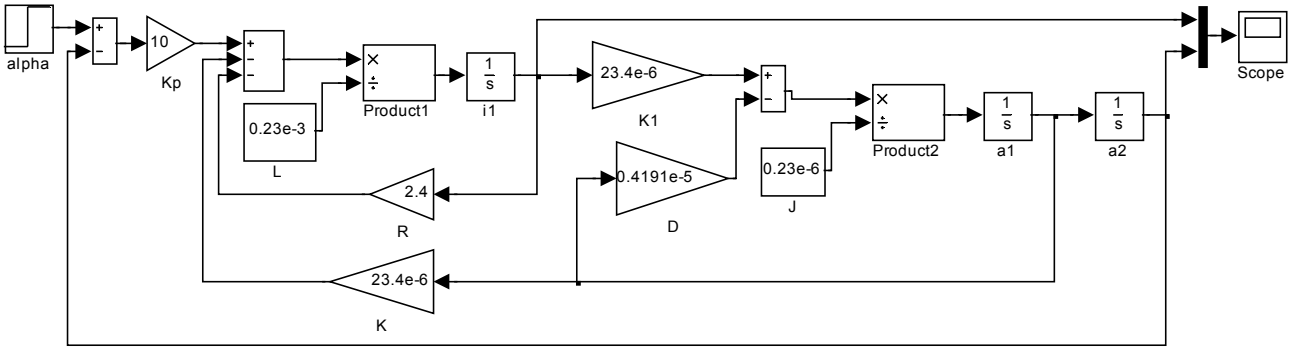
Model of a position control

a) Model zum Experimentieren

Zuerst explizite Dgl 1.Ordnung aufstellen:

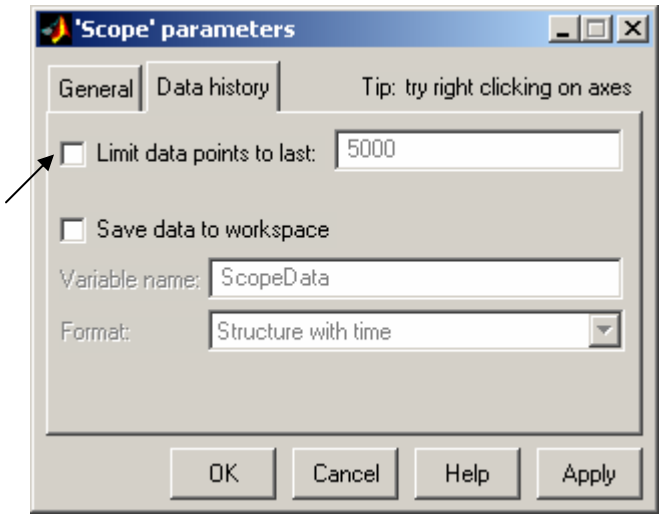
$$\dot{i}_m = \frac{1}{L} \cdot (U - R \cdot i_m - K \cdot \omega_m)$$

$$\dot{\omega}_m = \frac{1}{J} \cdot (K \cdot i_m - D \cdot \omega_m)$$



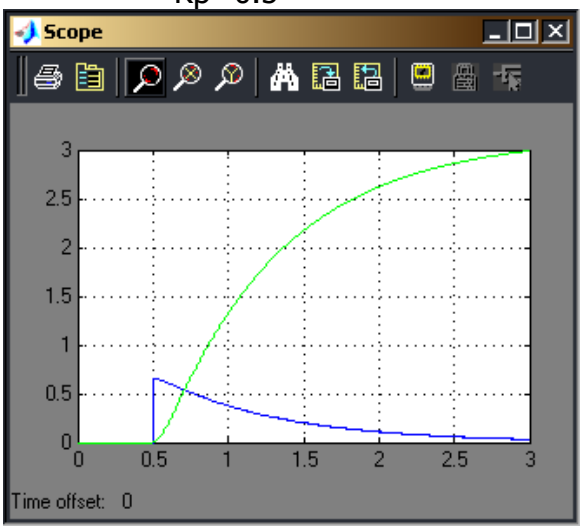
<u>Wichtig bei den Scope Einstellungen:</u>

- Limit data points deaktivieren wenn keine Daten an Matlab übergeben werden. Sonst wird evt. nur ein Teil des Graphen ausgegeben!!!

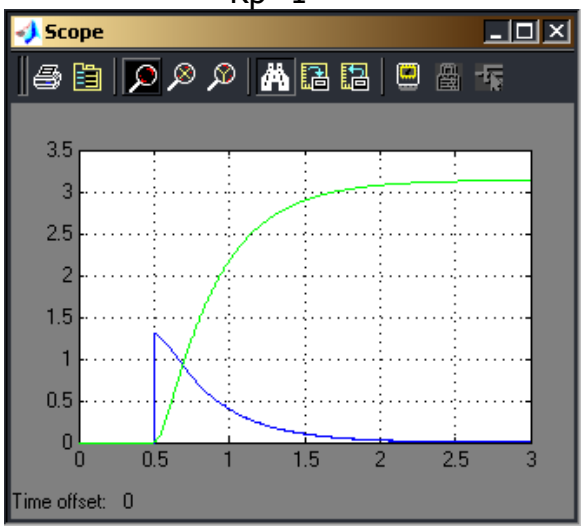


b) Unterschiedliche Parameter für K_P bei U=3,14 und Ode45

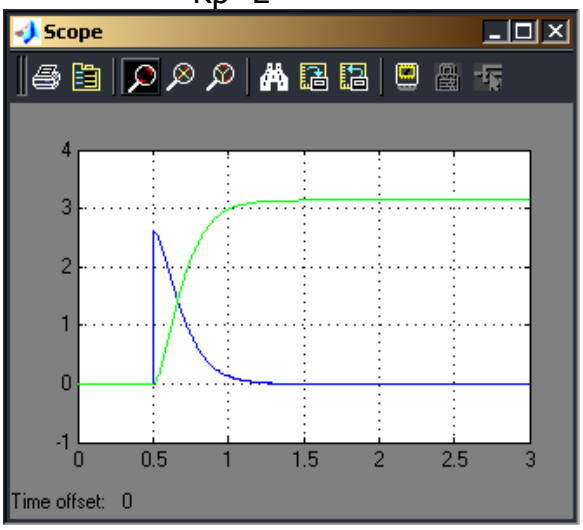
Kp=0.5



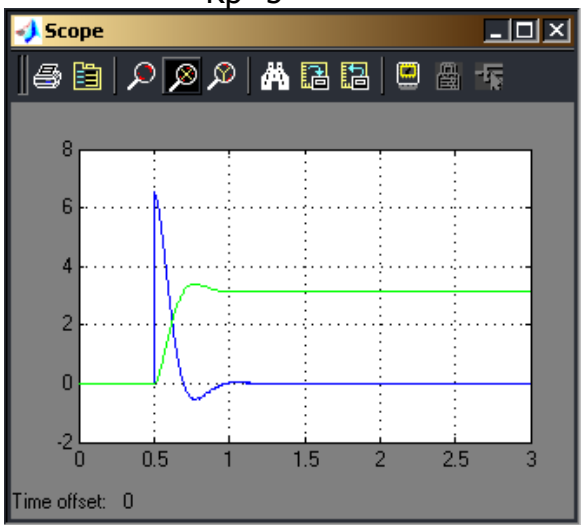
Kp=1



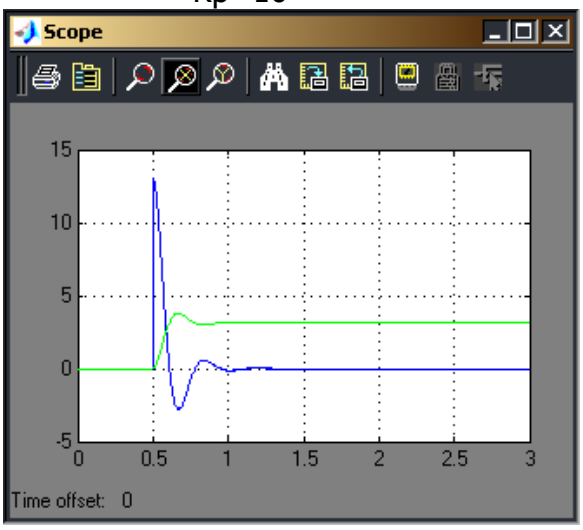
Kp=2



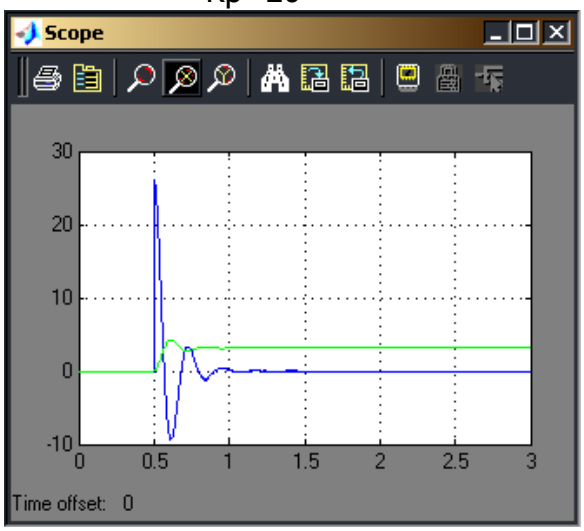
Kp=5



Kp=10



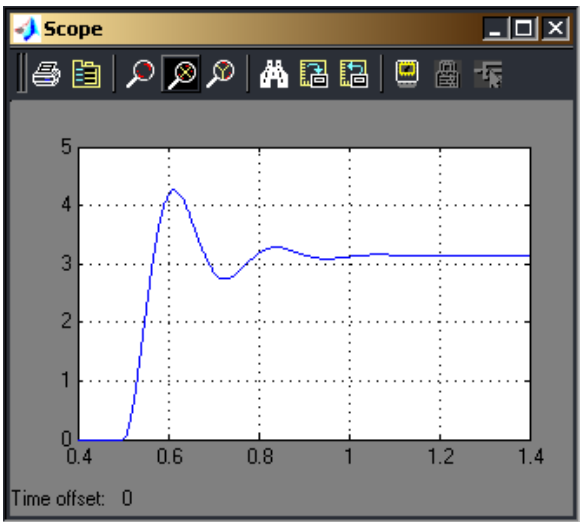
Kp=20



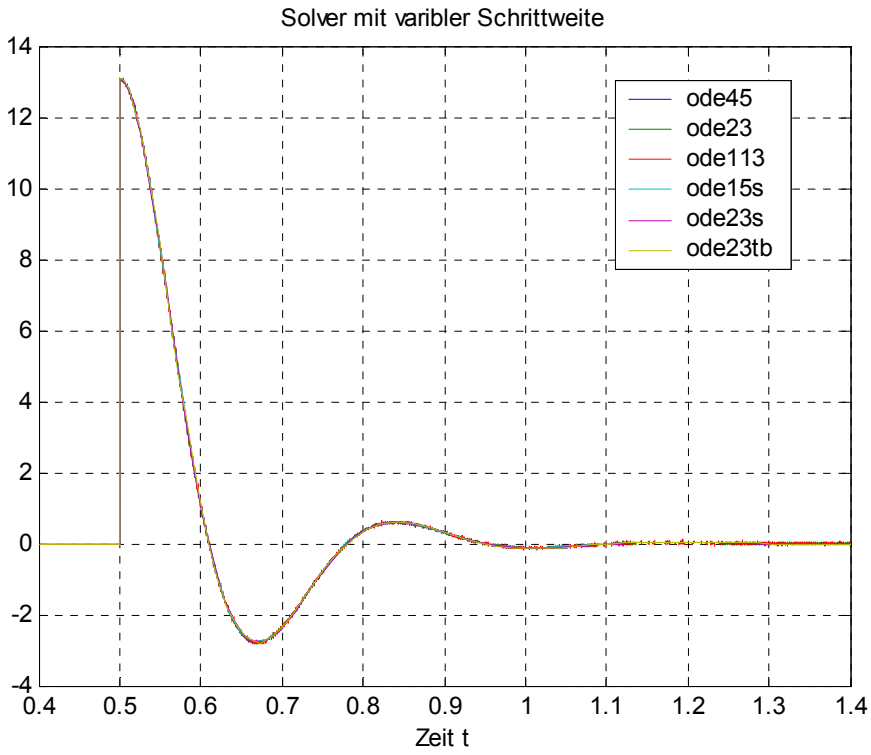
Auswertung:

- Mit steigendem Kp wird der Regelkreis immer schneller, allerdings nimmt auch der Überschwinger zu. Idealer Wert für Kp liegt zwischen 5 und 10 je nachdem was für Überschwinger akzeptiert werden.

  Detailaufnahme Für Kp=20



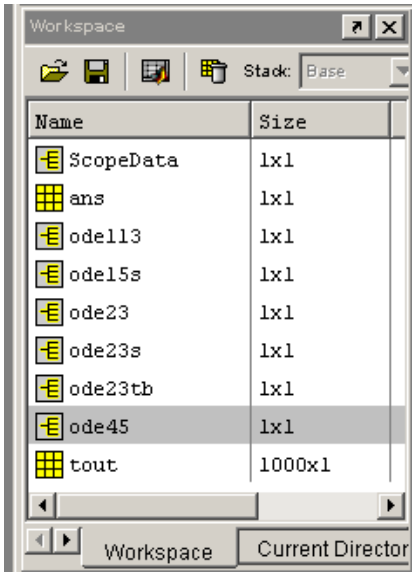
c) Unterschiedliche Solver



Solver mit fester Schrittweite versagen bei dieser Simulation.
Sie könne den steilen Anstieg des Motorstroms nicht verarbeiten.

Solver mit variabler Schrittweise liefern identische Ergebnisse. Sie variieren aber in ihrer Werteanzahl und Simulationsdauer.

Variablen (typ struct)



m-File zur Ploterstellung

```
plot(ode45.time, ode45.signals.values,...
    ode23.time, ode23.signals.values, ...
    ode113.time, ode113.signals.values,...
    ode15s.time, ode15s.signals.values,...
    ode23s.time, ode23s.signals.values,...
    ode23tb.time, ode23tb.signals.values)
title('Solver mit variabler Schrittweite')
xlabel('Zeit t')
legend('ode45', 'ode23','ode113','ode15s',...
      'ode23s','ode23tb')
grid
```