重写调速控制器模型 Controller，并进行参数整定，实现更好的控制性能，评价指标为：

1）稳态速度误差尽可能小。

2）电机速度输出的动态特性好。

3）电机加减速过程中，电流平稳变化无明显振荡。

在满足上述要求的情况下，可完成仿真作业的提高要求，即把电机和惯量负载通过阻尼弹簧系统相连，阻尼系数 d=0.2，弹簧系数 c=10，设计控制器并整定参数，满足上述三点控制要求。

几点说明事项：

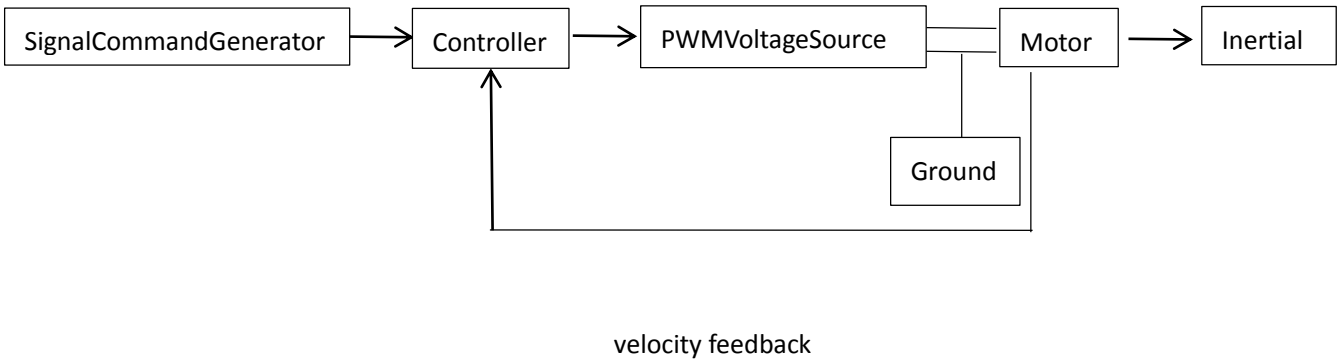
1）为了简化仿真，没有对调压装置和电机的电流值进行限制，但调速控制器的输出最大为正负 10V，如果采用了积分控制，请考虑积分饱和效应。

2）鼓励使用速度环和电流环双环控制。

3）撰写仿真报告，说明控制器的结构，参数整定的方法和过程，对仿真结果进行分析。

解：

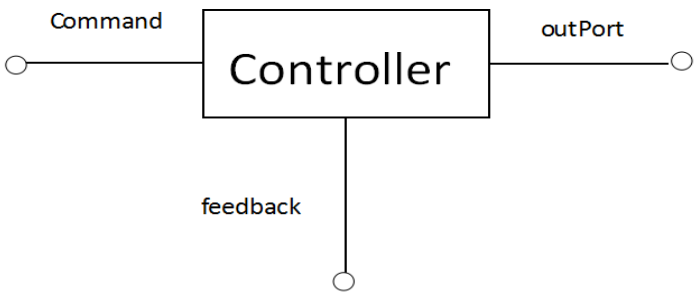
系统整体结构如下：



其中 Controller 部分的代码如下

```
block Controller

  InPort command(n=1);
  InPort feedback(n=1);
  OutPort outPort(n=1);

  Real error;
  Real pout;
  parameter Real Kp=10;
  parameter Real Max_Output_Pos = 10;
  parameter Real Max_Output_Neg = -10;

//  parameter Real Ki=1;

algorithm

   error := command.signal[1] -  feedback.signal[1];
   pout := Kp * error;

   if pout > Max_Output_Pos then
     outPort.signal[1] := Max_Output_Pos;
   elseif pout < Max_Output_Neg then
     outPort.signal[1] := Max_Output_Neg;
   else
     outPort.signal[1] := pout;
   end if;


end Controller;
```

结构如下：



其中 `error := command.signal[1] - feedback.signal[1];` 计算反馈信号和指令信号的偏差。且这个 controller 的控制是一个简单的比例控制，且输出的值在 Max_Output_Pos 和 Max_Output_Neg 之间。

我们的目的就是重写这一部分代码，使用最经典的 PID 控制算法来优化控制结果。评价的指标为

1）稳态速度误差尽可能小。

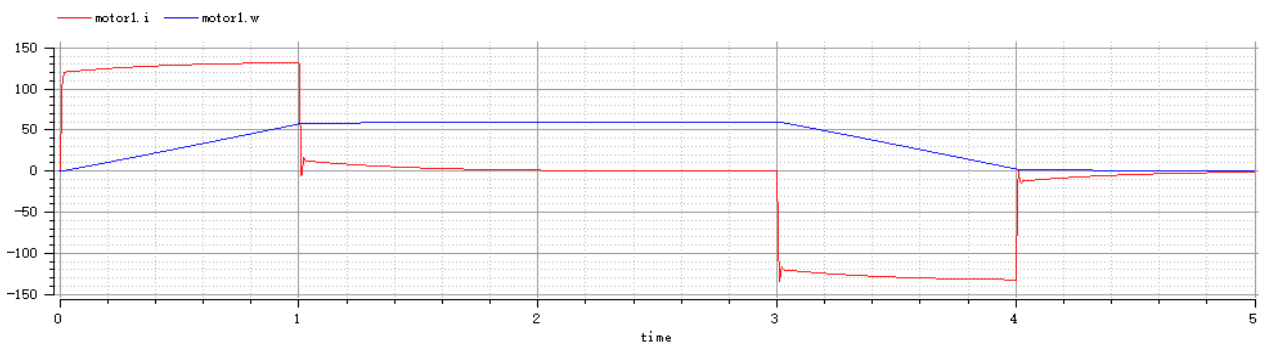
2）电机速度输出的动态特性好。

3）电机加减速过程中，电流平稳变化无明显振荡。

重写后如下：

```
block Controller

  InPort command(n=1);
  InPort feedback(n=1);
  OutPort outPort(n=1);

  Real error;
  Real pout;
  Real poutIntegral;
  parameter Real Kp=2;    //proportional
  parameter Real Ti=1;     //integral
  parameter Real Td=0.25;        //deferential
  parameter Real Max_Output_Pos = 10;
  parameter Real Max_Output_Neg = -10;

//  parameter Real Ki=1;

equation
   error = command.signal[1] -  feedback.signal[1];
   der(poutIntegral)=error/Ti;

algorithm
   pout := Kp * (error + Td * der(error) + poutIntegral);

   if pout > Max_Output_Pos then
     outPort.signal[1] := Max_Output_Pos;
   elseif pout < Max_Output_Neg then
     outPort.signal[1] := Max_Output_Neg;
   else
     outPort.signal[1] := pout;
   end if;


end Controller;
```

加入了 PID 调节，其中参数 Kp=2，Ti=1，Td=0.25。先调节 Kp 得到比较好的 tr，再加入 Td 减小 Mp，然后加入 Ti 消除稳态误差，最后整体调节 Kp，Ti，Td 得到比较好的相应曲线。如下：



可以看出：

1）稳态速度误差基本为 0。

2）电机速度输出的动态特性好。

3）电机加减速过程中，电流平稳变化，有轻微震荡，这是为了加快响应速度。

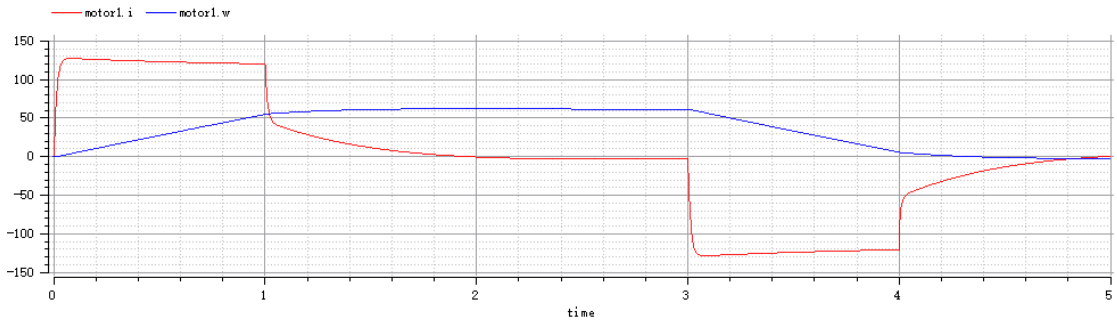
附：

- 上述 PID 算法是针对速度的闭环控制，然后我尝试使用电流速度双闭环的控制写 PID 发现很多问题，按照书上的双闭环控制系统的框图写了如下 controller：

```
block Controller

  InPort command(n=1);
  InPort feedback(n=1);
  InPort currentfeedback(n=1);
  OutPort outPort(n=1);

  Real error;
  Real pout;
  Real poutIntegral;
  Real currenterror;
  Real currentpout;
  Real currentpoutIntegral;
  parameter Real Kp=2;    //proportional
  parameter Real Ti=1;     //integral
  parameter Real Td=0.25;        //deferential
  parameter Real Kpc=1.49;    //proportional
  parameter Real Tic=1;     //integral
  parameter Real Tdc=0.01;        //deferential
  parameter Real Max_Output_Pos = 10;
  parameter Real Max_Output_Neg = -10;
```

```
equation
   error = command.signal[1] -  feedback.signal[1];
   der(poutIntegral)= error/Ti;

algorithm
   pout := Kp * (error + Td * der(error) + poutIntegral);

   if pout > Max_Output_Pos then
     pout := Max_Output_Pos;
   elseif pout < Max_Output_Neg then
     pout := Max_Output_Neg;
   else
     pout := pout;
   end if;

equation
   currenterror = pout - currentfeedback.signal[1]/17;
   der(currentpoutIntegral)= currenterror/Tic;

algorithm
   currentpout := Kpc * (currenterror + Tdc * der(currenterror) + currentpoutIntegral);

   if currentpout > Max_Output_Pos then
     outPort.signal[1] := Max_Output_Pos;
   elseif currentpout < Max_Output_Neg then
     outPort.signal[1] := Max_Output_Neg;
   else
     outPort.signal[1] := currentpout;
   end if;

end Controller;
```

即将原来的 PID 算法的输出与 currentfeedback 的电流作差值，其中的 17 是为了满足饱和速度控制输出量和最大反馈电流之间的数量值接近，也就是电流反馈系数。然后再进行 PID 控制，最后调整一番后，输出的波形如下



与上述单闭环控制的图像相比电流几乎没有震荡，但是明显速度的控制受到了影响，可能是我参数没有调好的缘故。

- 把电机和惯量负载通过阻尼弹簧系统相连，阻尼系数 d=0.2，弹簧系数 c=10，设计控制器并整定参数，满足上述三点控制要求。

这个扩展题目，我尝试写了一个阻尼弹簧的 model，然后再系统中连接起来发现不能通过：

```
model DamperSpring
  Angle phi                "Absolute rotation angle of component";
  RotFlange_a rotFlange_a  "(left) driving flange (axis directed into plane)";
  RotFlange_b rotFlange_b  "(right) driven flange (axis directed out of plane)";
  parameter Real k=1;
  parameter Real b=1;
  AngularVelocity     w;
equation
  phi = rotFlange_a.phi - rotFlange_b.phi;
  w = der(phi);
  rotFlange_a.tau = k*phi + b*w;
  rotFlange_b.tau = k*phi + b*w;
end DamperSpring;
```

```
model DCMotorControlSystem

  Ground       ground1;
  DamperSpring dspring1(k = 10,b = 0.2,rotFlange_b(phi(fixed = true)));
  Inertia      inertia1(J = 3, w(fixed = true));
  DCMotor      motor1(J = 1,R = 0.6,L = 0.01,Kt=1.8, Ke= 1.8,rotFlange_b(phi(fixed = true)));
  CommandSignalGenerator  sg1;
  Controller   con1;
  PWMVoltageSource PowerSource1;
equation
  connect(sg1.outPort, con1.command);
  connect(con1.feedback, motor1.SensorVelocity);
  connect(con1.outPort, PowerSource1.Command);
  connect(PowerSource1.p, motor1.p);
  connect(motor1.rotFlange_b, dspring1.rotFlange_a);
  connect(dsping1.rotFlange_b, inertia1.rotFlange_a);
  connect(PowerSource1.n, ground1.p);
  connect(ground1.p, motor1.n);
end DCMotorControlSystem;
```

出现错误如下：

```
simulate(DCMotorControlSystem,stopTime=5)
```

```
record SimulationResult
    resultFile = "",
    messages = "Failed to build model: DCMotorControlSystem"
end SimulationResult;
OMC-ERROR:
"[<interactive>:16:3-16:54:writable] Error: Variable dsping1.rotFlange_b not found in scope DCMotorControlSystem.
Error: Error occurred while flattening model DCMotorControlSystem
"
```

不知道如何解决。