Control an IT1-Process with a P-Controller (symbolically):

```
by L.Billmann 06/2011
File: IT1withP_sym.m
Doku: IT1withP_sym.pdf
Task: To check the controlability of an IT1-process with a simple
      P-controller and determine the expected time rsponses.
ext. Files:   -
M-Block:      -
LBtools/math: -
```

```
clc, clear all
```

Check stability by pole-placement:

```
syms Kr Ti T1 p
```

- Determine the reference transfer function Fw

```
Fr = Kr;
fprintf('    with   Fr(p) = %s\n',char(Fr))
Fs = 1/(Ti*p*(1+T1*p));
fprintf('    and    Fs(p) = %s\n',char(Fs))
disp(   '    we get Fw(p) =')
Fw = Fr*Fs/(1 + Fr*Fs); Fw=simplify(Fw);pretty(Fw)
```

```
    with   Fr(p) = Kr
    and    Fs(p) = 1/Ti/p/(1+T1*p)
    we get Fw(p) =

                                   Kr
                          --------------------
                                     2
                          Ti p + Ti p  T1 + Kr
```

- Look for stable pole placement

```
[num,den] = numden(Fw); poles = solve(den,p);
fprintf('    from the characteristic polynomial Q(p) = %s\n',...
         [char(den) '= 0'])
disp('    we get p1 ='), pretty(poles(1))
disp('    and    p2 ='), pretty(poles(2))

disp(' ')
Krsemi = solve(poles(1),Kr);
fprintf('    For the stability limit case (poles=0), we get Kr = %s\n',...
        char(Krsemi))
disp('    So we can state out the stability condition:')
fprintf('\n    >>> stable, if Kr > %s\t\t\t\t(1)\n',char(Krsemi))
```

```
    from the characteristic polynomial Q(p) = Ti*p+Ti*p^2*T1+Kr= 0
    we get p1 =

                                 2               1/2
                         -Ti + (Ti   - 4 Ti T1 Kr)
                     1/2 ---------------------------
                                    Ti T1
    and    p2 =

                                 2               1/2
                          Ti + (Ti   - 4 Ti T1 Kr)
                    - 1/2 --------------------------
                                    Ti T1

    For the stability limit case (poles=0), we get Kr = 0
    So we can state out the stability condition:

    >>> stable, if Kr > 0                               (1)
```

- Check for aperiodic/periodic poles

```
disp('    Behavior becomes periodic if a pole becomes complex,')
disp('    which happens if radicant becomes negative.')
disp('    Looking at the so called aperiodic limit case,')
disp('    where we change from periodic to aperiodic while')
disp('    the radicant Ti^2-4*Ti*T1*Kr becomes zero.')
disp('    With this we get:')
apg=solve(Ti^2-4*Ti*T1*Kr,Kr);
fprintf('\n    >>> aperiodic if Kr < %s\t\t\t(2)\n',char(apg))
```

```
    Behavior becomes periodic if a pole becomes complex,
    which happens if radicant becomes negative.
    Looking at the so called aperiodic limit case,
    where we change from periodic to aperiodic while
    the radicant Ti^2-4*Ti*T1*Kr becomes zero.
    With this we get:

    >>> aperiodic if Kr < 1/4*Ti/T1                     (2)
```

Determine the expected reference reaction:

```
disp('    With the PT2 reference transfer function Fw, we get the')
fprintf('    amplification to Kw = %s\n',char(subs(Fw,p,0)))
fprintf('\n    >>> no permanent control deviation expected\n')

disp(' ')
DGLw='Ti*T1*D2x + Ti*D1x + Kr*x = Kr';
fprintf('    With the corresponding differential equation: %s\n',DGLw)
disp('    we get the solution for an input unit step to x(t) =')
h= dsolve(DGLw,'Dx(0)=0','x(0)=0'); pretty(h)
```

```
    With the PT2 reference transfer function Fw, we get the
    amplification to Kw = 1

    >>> no permanent control deviation expected

    With the corresponding differential equation: Ti*T1*D2x + Ti*D1x + Kr*x = Kr
    we get the solution for an input unit step to x(t) =

                 (Ti - %1) t
      exp(- 1/2 -----------) (-%1 - Ti + 4 Kr T1)
                    Ti T1
- 1/2 -------------------------------------------
                     -Ti + 4 Kr T1

                           (Ti + %1) t
             exp(- 1/2 -----------) (-Ti + %1 + 4 Kr T1)
                              Ti T1
       - 1/2 ------------------------------------------- + 1
                              -Ti + 4 Kr T1

           2           1/2
  %1 := (Ti  - 4 Ti T1 Kr)
```

Determine the expected input disturbance reaction:

```
disp(   '    With Fr(p) and Fs(p) we get Fzy(p) =')
Fzy = Fs/(1 + Fr*Fs); Fzy=simplify(Fzy);pretty(Fzy)
disp('    This is a PT2 transfer function with the')
fprintf('    amplification Kzy = %s\n',char(subs(Fzy,p,0)))
fprintf('\n    >>> permanent control deviation expected\n')

disp(' ')
DGLzy='Ti*T1*D2x + Ti*D1x + Kr*x = 1';
fprintf('    With the corresponding differential equation: %s\n',DGLzy)
disp('    we get the solution for an input unit step to x(t) =')
h= dsolve(DGLzy,'Dx(0)=0','x(0)=0'); pretty(h)
```

```
    With Fr(p) and Fs(p) we get Fzy(p) =

                                              1
                                 --------------------
                                                2
                                 Ti p + Ti p  T1 + Kr
    This is a PT2 transfer function with the
    amplification Kzy = 1/Kr

    >>> permanent control deviation expected

    With the corresponding differential equation: Ti*T1*D2x + Ti*D1x + Kr*x = 1
    we get the solution for an input unit step to x(t) =

                 (Ti - %1) t
      exp(- 1/2 -----------) (-%1 - Ti + 4 Kr T1)
                    Ti T1
- 1/2 -------------------------------------------
                  (-Ti + 4 Kr T1) Kr

                           (Ti + %1) t
             exp(- 1/2 -----------) (-Ti + %1 + 4 Kr T1)
                              Ti T1                          1
       - 1/2 ------------------------------------------- + ----
                         (-Ti + 4 Kr T1) Kr                 Kr

           2           1/2
  %1 := (Ti  - 4 Ti T1 Kr)
```

Determine the expected output disturbance reaction:

```
disp(   '    With Fr(p) and Fs(p) we get Fzi(p) =')
Fzi = 1/(1 + Fr*Fs); Fzi=simplify(Fzi);pretty(Fzi)
disp('    This is a D2T2 transfer function with the')
fprintf('    amplification Kzy = %s\n',char(subs(Fzi,p,0)))
fprintf('\n    >>> no permanent control deviation expected\n')

disp(' ')
DGLzi='Ti*T1*D2x + Ti*D1x + Kr*x = Ti*D1y + Ti*T1*D2y';
fprintf('    With the corresponding differential equation: %s\n',DGLzi)
disp('    we get the solution for an input unit step to x(t) =')
h= dsolve(DGLzi,'Dx(0)=0','x(0)=0'); pretty(h)
```

```
    With Fr(p) and Fs(p) we get Fzi(p) =

                                   Ti p (1 + T1 p)
                                 --------------------
                                                2
                                 Ti p + Ti p  T1 + Kr
    This is a D2T2 transfer function with the
    amplification Kzy = 0

    >>> no permanent control deviation expected

    With the corresponding differential equation: Ti*T1*D2x + Ti*D1x + Kr*x = Ti*D1y + Ti*T1*D2y
    we get the solution for an input unit step to x(t) =

              /   t
              |  /
              | |
  x(t) = Ti | |
              | |
              |/
              \   0

        /               /  2          \   \             (-Ti + %1) _z1
        |/ d          \   | d           |   |  exp(- 1/2 --------------) d_z1
        ||---- y(_z1)| + |----- y(_z1)| T1|                Ti T1
        |\d_z1        /   |    2        |   |
        \                 \d_z1         /   /

                (Ti + %1) t
```

```
        exp(1/2 -----------) -
                   Ti T1

           t
          /  /                /  2         \   \
         |   |/ d         \   | d          |   |          (Ti + %1) _z1
         |   ||---- y(_z1)| + |----- y(_z1)| T1| exp(1/2 -------------) d_z1
         |   |\d_z1       /   |    2       |   |              Ti T1
        /    \                \d_z1        /   /
          0

                                \
                                |
                  (-Ti + %1) t  |        t      /                  1/2
        exp(- 1/2 ------------)| exp(- ----)  /  (Ti (Ti - 4 Kr T1))
                     Ti T1      |        T1   /
                                |
                                /

                            1/2
%1 := (Ti (Ti - 4 Kr T1))
```

Published with MATLAB® 7.4