Exercise 1: Introduction Matlab

Matlab can be used like a programming language. This exercise trains some features of Matlab in evaluation of functions, writing programs and plotting functions.

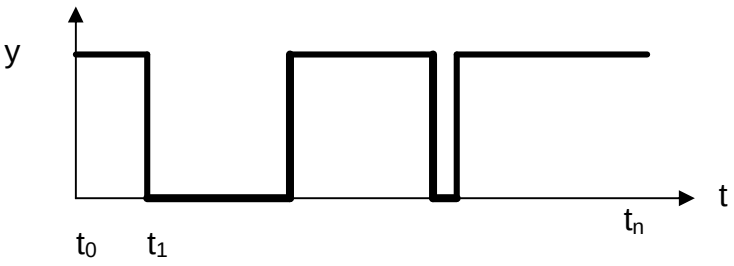
a) Plot the functions sin, cos, tan, arcsin, arccos arctan in one plot using subplot. Comment the axis and functions in the graphic.

b) Plot the functions logarithm, exponent, square-root and power in one plot using subplot. Comment the axis and function in the graphic

c) Plot the polynomial $y = a_0 + a_1 x + a_2 x^2 + a_3 x^3$ in one plot. Show the influence of the coefficients.

d) Construct for each of the PT1 $G_{PT1}(s) = \frac{K}{1+Ts}$; and PT2 $G_2(s) = \frac{K}{1+2dTs+T^2s^2}$ control blocks a plot that contains the bode-diagram, the nyquist plot and the step-response. Show the influence of the parameters to the plot:

e) Build a function that creates a switching function with variable switch time points [t,y]=stepstep(y0,dt,[t0 t1 t2 t3 ...tn]). Check the input arguments on errors!



f) Compute with the matrix and the vectors:

$$\vec{a} = \begin{bmatrix} 2 \\ 0 \\ 5 \end{bmatrix}, \vec{b} = \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix}, \vec{c} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}; \underline{A} = \begin{bmatrix} 2 & 5 & 2 \\ 4 & 34 & 8 \\ 4 & 5 & 2 \end{bmatrix}; \underline{B} = \begin{bmatrix} 2 & 4 & 0 \\ 3 & 2 & 0 \\ 6 & 2 & 0 \end{bmatrix}$$

$$\vec{a} \cdot \vec{b}; \vec{a} + \vec{b}; \underline{A} \cdot \vec{b}; \underline{A}^T \cdot \vec{c}; |\underline{A}|; |\underline{B}|; \underline{A}^{-1}; \underline{B}^{-1};$$

Document all you work including plots and source-code in Word or Powerpoint.,

Exercise 1: Introduction Matlab

a)

1. Aufgabenstellung siehe Deckblatt

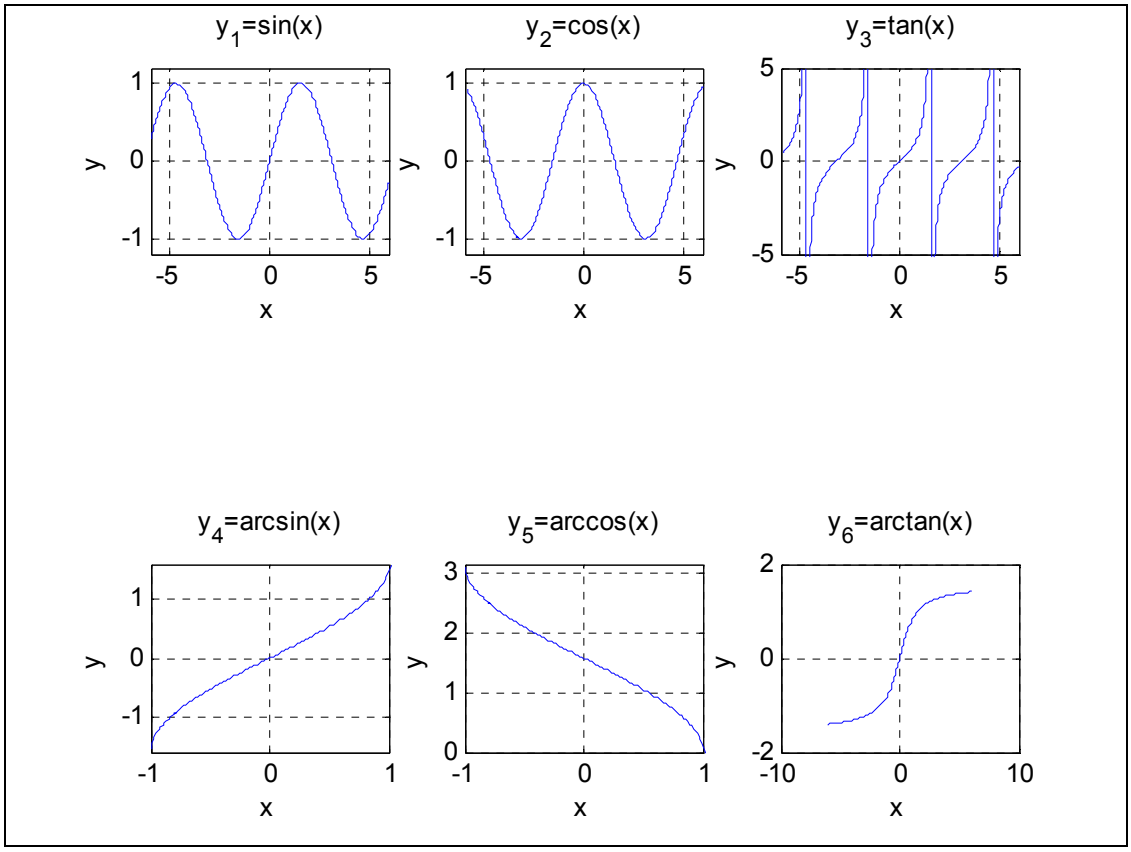
```
--------------------------------------------------------------------------------
        %% Eingabe des Startwerts der Berechnung und Zuweisung an Variable a
a=input('Bitte geben Sie den Startwert ein:');
        %% Eingabe des Endwerts der Berechnung und Zuweisung an Variable b
b=input('Bitte geben Sie den Endwert ein:');
        %% Lösche aktuellen Inhalt der aktuellen Figur (falls bereits vorhanden)
clf;
        %% Vektor Erstellung mit 1000 linear unterteilten Werten von a bis b
x=linspace(a,b,1000);
        %% Weise y1 Sinus von Vektor x zu (y1 ist nun Vektor mit Erg. von x)
y1=sin(x);
        %% Weise y2 Kosinus von Vektor x zu (y2 ist nun Vektor mit Erg. von x)
y2=cos(x);
y3=tan(x);
y4=asin(x);
y5=acos(x);
y6=atan(x);
        %% Erstelle Supblot mit Fenteraufteilung 3x3 und plotte in Fenster 1
        %% Titel des Subplots sei 'Y_1=sin(x)', x-Achsenbezeichnung: 'x'
        %% y-Achseenbezeichnung: 'x'
        %% plotte x-Achse im Bereich für a < x < b
        %% plotte y-Achse im Bereich -1.2 < y < -1.2
        %% grid on = schalte Gitternetzlinien an
subplot(3,3,1), plot(x,y1), title('y_1=sin(x)'), xlabel('x'), ylabel('y'),...
    axis([a, b, -1.2, 1.2]), grid on;
        %% dito oben mit cosinus
subplot(3,3,2), plot(x,y2), title('y_2=cos(x)'), xlabel('x'), ylabel('y'),...
    axis([a, b, -1.2, 1.2]), grid on;
subplot(3,3,3), plot(x,y3), title('y_3=tan(x)'), xlabel('x'), ylabel('y'),...
    axis([a, b, -5, 5]), grid on;
        %% Subplot 4,5,6 bewusst ausgeblendet um die Diag.titel korrekt anzuzeigen
subplot(3,3,7), plot(x,y4), title('y_4=arcsin(x)'), xlabel('x'), ylabel('y'),...
    axis([-1, 1, -pi/2, pi/2]),  grid on;
subplot(3,3,8), plot(x,y5), title('y_5=arccos(x)'), xlabel('x'), ylabel('y'),...
    axis([-1, 1, 0, pi]), grid on;
subplot(3,3,9), plot(x,y6), title('y_6=arctan(x)'), xlabel('x'), ylabel('y'),...
    grid on;
--------------------------------------------------------------------------------
```

Diagrammbeschriftungen mit z.B. `title('y_5=arccos(x)')` erscheinen als $y_5=arccos(x)$.
Ebenso z.B. `title('y^5=arccos(x)')` als $y^5=arccos(x)$.

Nach Programmaufruf erfolgt Aufforderung zur Eingabe der Berechnungsbereiche:

```
Bitte geben Sie den Startwert ein:-6
Bitte geben Sie den Endwert ein:6
```

2. Nach Eingabe der Bereichswerte -6 und 6 ergibt sich folgende Graphik:

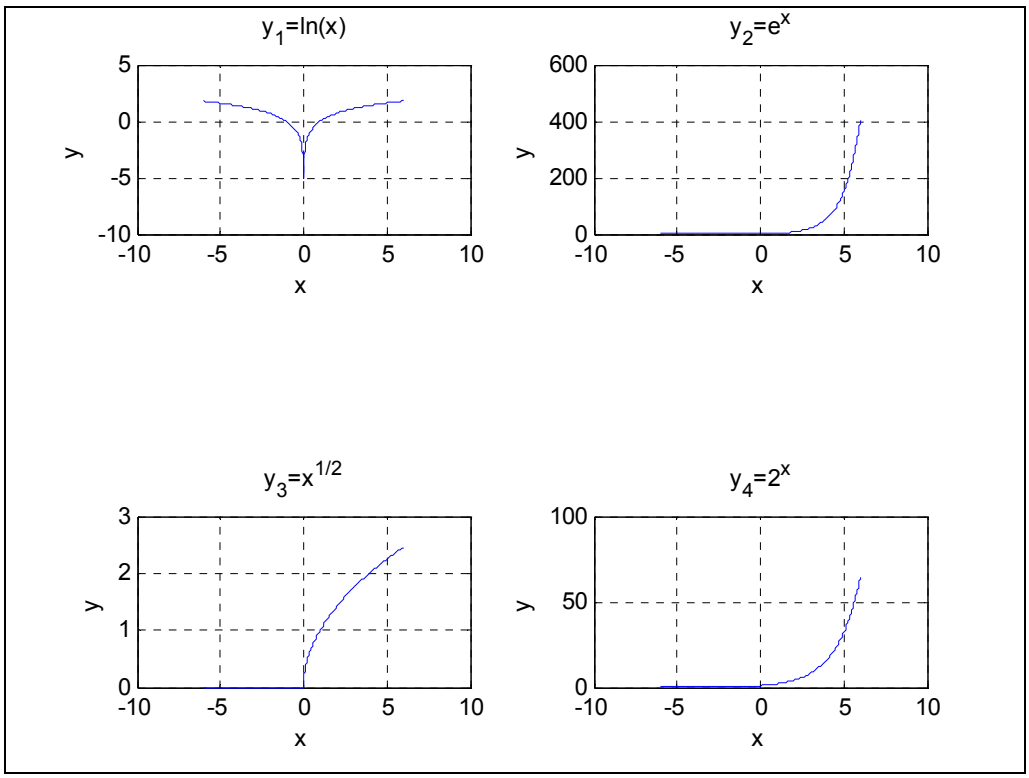


b)

1. Aufgabenstellung siehe Deckblatt

```
--------------------------------------------------------------------------------
        %% Eingabe des Startwerts der Berechnung und Zuweisung an Variable a
a=input('Bitte geben Sie den Startwert ein:');
        %% Eingabe des Endwerts der Berechnung und Zuweisung an Variable b
b=input('Bitte geben Sie den Endwert ein:');
        %% Lösche aktuellen Inhalt der aktuellen Figur (falls bereits vorhanden)
clf;
        %% Vektor Erstellung mit 1000 linear unterteilten Werten von a bis b
x=linspace(a,b,1000);
        %% Weise y1 Vektor mit Ergebnissen von log(x) zu
y1=log(x);
        %% Weise y2 Vektor mit Ergebnissen von e^x zu
y2=exp(x);
        %% Weise y3 Vektor mit Ergebnissen von (x)^(1/2) zu
y3=sqrt(x);
        %% Weise y4 Vektor mit Ergebnissen von 2^x zu
y4=2.^x;
        %% Erstelle Supblot mit Fenteraufteilung 3x2 und plotte in Fenster 1
        %% Titel des Subplots sei 'Y_1=ln(x)', x-Achsenbezeichnung: 'x'
        %% y-Achseenbezeichnung: 'y'
        %% da axis[...] fehlt erfolgt die Axeneinteilung automatisch
        %% grid on = schalte Gitternetzlinien an
subplot(3,2,1), plot(x,y1), title('y_1=ln(x)'), xlabel('x'), ylabel('y'), grid on;
        %%  dito oben nur andere Funktion und Subplotfenster 2
subplot(3,2,2), plot(x,y2), title('y_2=e^x'), xlabel('x'), ylabel('y'), grid on;
        %% Subplot 3und4 bewusst Ausgeblendet um die Diag.titel korrekt anzuzeigen
subplot(3,2,5), plot(x,y3), title('y_3=x^1^/^2'), xlabel('x'), ylabel('y'),grid on;
subplot(3,2,6), plot(x,y4), title('y_4=2^x'), xlabel('x'), ylabel('y'), grid on;
--------------------------------------------------------------------------------
```

2. Resultierend Grafik



c)

1. Aufgabenstellung siehe Deckblatt

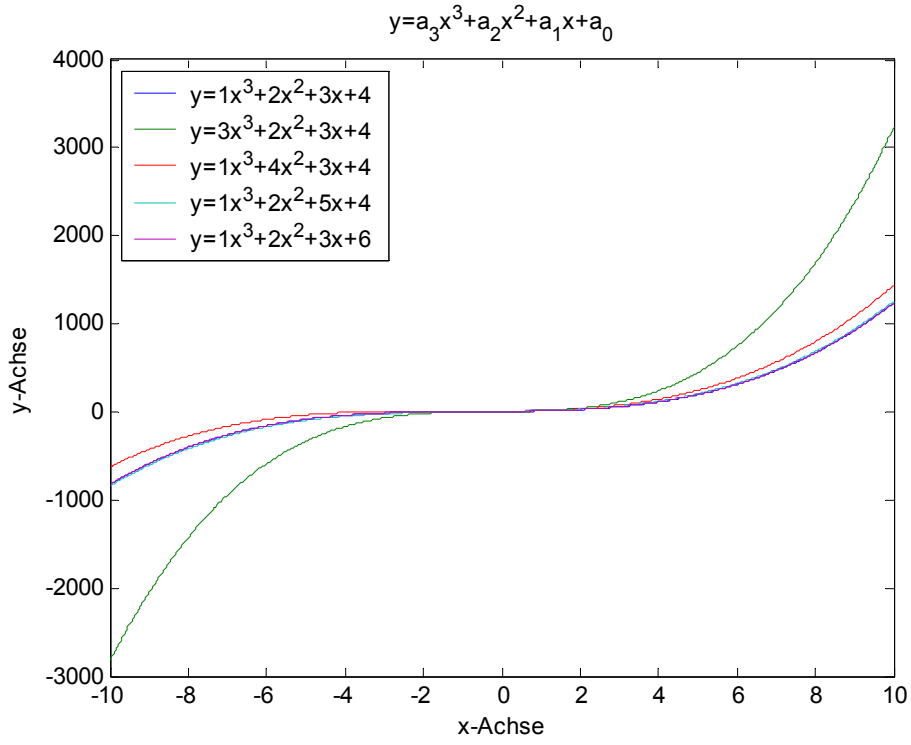
```
--------------------------------------------------------------------------------
        %%              Optional
        %%
        %%  erfolgt die Parametereingabe nicht über das Command Window
        %%  können die Parameter auch duch den Funktionsaufruf mit
        %%  aufgabec(a3, a2, a1, a0, z, a, b) erfolgen.
        %%  Hierfür müssen einfach die Parameterabfragen x=input('xxx')
        %%  als Kommentar mit % gekenzeichnet werden (sind somit nicht
        %%  aktiv). Die folgende Zeile muss dagegen aktiviert werden durch
        %%  entfernen des % - Zeichens vor function

 %function h= aufgabec(a3, a2, a1, a0, z, a, b)


        %%  löscht Command Window
clc;
        %%  löscht noch bestehende Variablen im Workspace
clear;
        %%  Beginn der Parameterabfrage
a=input('Bitte geben Sie den Startwert ein:');
b=input('Bitte geben Sie den Endwert ein:');
z=input('Bitte geben Sie den Variationswert ein:');
        %%  disp('..xx..') gibt eine Textzeile im Command Window aus
disp('Es erfolgt eine Variation des Polynoms y=a3x^3+a2x^2+a1x+a0');
disp('Bitte geben Sie nun die Parameter ein');
        %%  weitere Parametereingaben und ihre Zuweisung
a3=input('a3:');
a2=input('a2:');
a1=input('a1:');
a0=input('a0:');
        %%  Vektor Erstellung mit 1000 linear unterteilten Werten von a bis b
x=linspace(a,b,1000);
        %%  erstellt leere Variable y (Deklaration), wird später benötigt
y=[];
        %%  erzeugt einen Zeilenvektor mit den Koeffizienten a3, a2,...
        %%  beginnend mit der höchsten Potenz a3*x^3 !!!
        %%  Achtung es muss sich um einen Zeilenvektor handeln (nicht mit ;
        %%  trennen --> Spaltenvektor)
p=[a3, a2, a1, a0];
        %%  polyval interpretiert p als a3x^3+a2x^2+a1x+a0
        %%  und setzt die für x vorgegebenen Werte (x ist Zeilenvektor) ein
        %%  die jeweiligen Ergebnisse werden in die erste Zeile von y geschrieben
        %%  y ist nun auch ein Zeilenvektor
        %%  y((1),:) bedeutet: schreibe in erste Zeile und alle Spalten
y((1),:)=polyval(p,x);
        %%  Nun wird ein String aus den Parametern zusammengesetzt um
        %%  ihn als Legende verwenden zu können
        %%  Achtung c(1)={} erzeugt ein cell-Array
        %%  num2str(p(1)) wandelt den 1 Wert von p also a3 in einen string
        %%  um
c(1)={['y='   num2str(p(1))   'x^3+'   num2str(p(2))   'x^2+'   num2str(p(3))   'x+'
num2str(p(4))]};

        %%  in dieser Schleife wird die Koeffizientenvariation durchgeführt
        %%  und die Strings für die Legende erzeugt (in cell-array)
        %%  schleife wird 4 mal durchlaufen um alle Koeffizienten zu
        %%  variieren
for n=1:4
        %%  bsp 1.Durchlauf
        %%  Der Koeffizienten a3 wird um z erhöht
    p(n)=p(n)+z;
        %%  für das neue Polynom werden die Ergebnisse ermittelt
        %%  und y in der 2. Zeile zugeordnet
```

```
    y((n+1),:)=polyval(p,x);
        %%  neuer String wird mit neuen Parametern erzeugt und c{2}
        %%  zugeordnet
    c(n+1)={['y='  num2str(p(1))  'x^3+'  num2str(p(2))  'x^2+'  num2str(p(3))  'x+'
num2str(p(4))]};
        %%  Der Koeffizienten a3 wird um z wieder erniedrigt -->nun wieder
        %%  ursprüngliche Koeffizienten
    p(n)=p(n)-z;
end
        %%  Im folgenden wird das Diagramm erzeugt und die Legende mit allen
        %%  Variationen hinzugefügt
plot(x,y(1,:),x,y(2,:),x,y(3,:),x,y(4,:),x,y(5,:));
title('y=a_3x^3+a_2x^2+a_1x+a_0');
xlabel('x-Achse');
ylabel('y-Achse');
        %%  umwandlung des cell-arrays in ein char-array
c=char(c);
        %%  legende wird diagramm hinzugefügt
        %%  die 2 bedeutet das die Legende oben links angeziegt wird
legend(c,2);
-------------------------------------------------------------------------------
```

Nach Programmaufruf Aufforderung zur Parametereingabe

```
Bitte geben Sie den Startwert ein:-10
Bitte geben Sie den Endwert ein:10
Bitte geben Sie den Variationswert ein:2
Es erfolgt eine Variation des Polynoms y=a3x^3+a2x^2+a1x+a0
Bitte geben Sie nun die Parameter ein
a3:1
a2:2
a1:3
a0:4
```

2. Resultierend Grafik

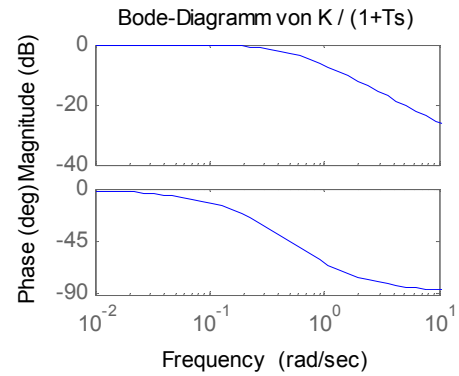


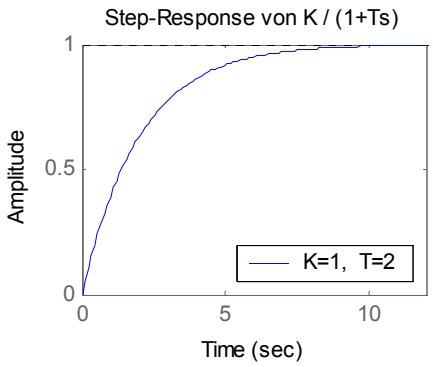
d)

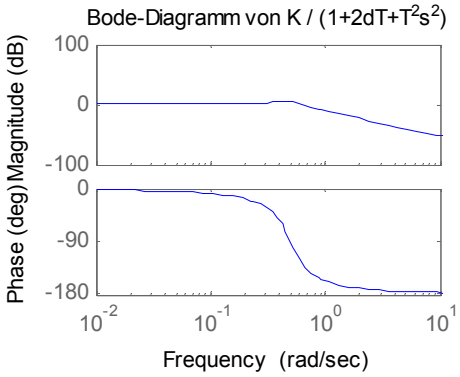
1. Aufgabenstellung siehe Deckblatt

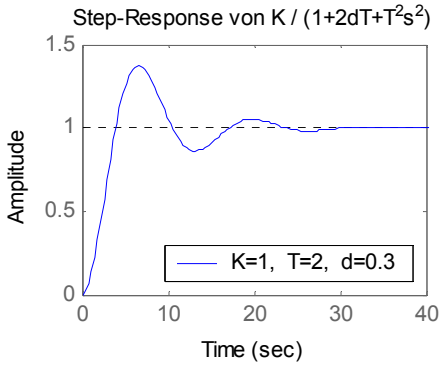
```
--------------------------------------------------------------------------------
% Aufruf der Funktion mit aufgabed(K,T,d)
% Wobei K=Verstärkung T=Zeitkonstante und d=Dämpfung ist.
% Es werden für das PT1 und PT2 Glied jeweils
% die Sprungantwort und das Bodediagramm geplottet
        %% Die obige Erläuterung erscheint bei der Eingabe
        %% help aufgabed
        %% beim Funktionsaufruf werden die Parameter K,T und d
        %% übergeben
function h=aufgabed(K,T,d)
        %% Aufstellung der Transferfunktion mit tf
        %% der als erstes aufgeführte Vektor [K] in tf(...) beinhaltet die
        %% Koefizienten der LaPlace-Gleiung im Zähler in absteigender Reihenfolge
        %% in diesem Fall nur eine Kontante
        %% der zweite Vektor enthält die Koeff. des Zählers
        %% hier (Ts+1)    ---> Gs1 = K/(T*s+1)
Gs1=tf([K],[T 1]);
        %% hier K/(T^2*s^2+2*d*T*s+1)
Gs2=tf([K],[T^2 2*d*T 1]);

        %% Aufstellung eines Subplot Fensters 2x2 und Auwahl des 1.Bezirks
subplot(2,2,1);
        %% Plottet Bodediagramm von Gs1
bode(Gs1);
        %% Tittel des Subplotts
title(['Bode-Diagramm von K / (1+Ts)']);


        %% neuer Subplotbezirk 2
subplot(2,2,2);
        %% Sprungantwort von Gs1
step(Gs1);
        %% Tittel des Subplotts
title(['Step-Response von K / (1+Ts)']);
        %% Erzeuge String für Legende und plottet ihn
legend(['K=' num2str(K) ',  T=' num2str(T)],4);


        %% neuer Subplotbezirk 3
subplot(2,2,3);
        %% Plottet Bodediagramm von Gs2
bode(Gs2);
title(['Bode-Diagramm von K / (1+2dT+T^2s^2)' ]);


        %% neuer Subplotbezirk 4
subplot(2,2,4);
        %% Sprungantwort von Gs2
step(Gs2);
title(['Step-Response von K / (1+2dT+T^2s^2)' ]);
legend(['K=' num2str(K) ',  T=' num2str(T) ',  d=' num2str(d)],4);

--------------------------------------------------------------------------------
```

2. Resultierend Grafik









e)

1. Aufgabenstellung siehe Deckblatt

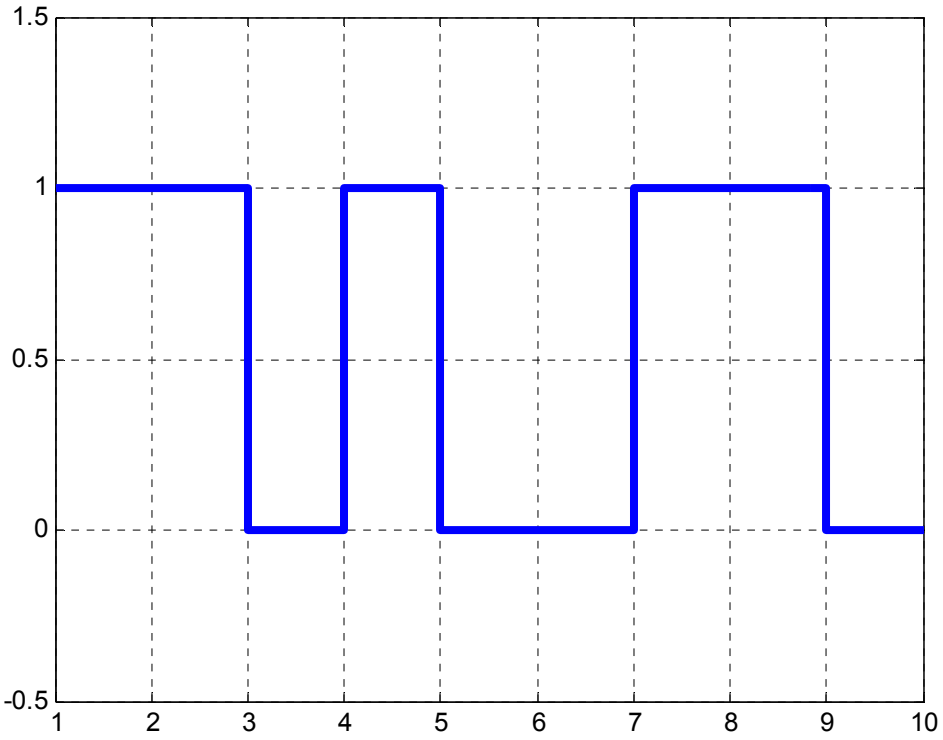
```
--------------------------------------------------------------------------------
%   Switching Funktion With Variable Switch Time Points
%   Funktionsaufruf mit aufgabee(y0, dt, feld)
%   Wobei y0=Amplitude, dt=Auflöungsschritte und feld= vektor mit Switching
%   Time Points
function [x,y] = aufgabee(y0, dt, feld)
        %%  Startet Pc-Zähler zur späteren Ermittlung der rechendauer
tic;
        %%  Überprüft ob dt ein vielfaches von 1 ist, sonst funktion nicht
        %%  ausführbar (findet ...
        %%  Überprüfung erfolt mit Hilfe von Modula, welches den Rest einer
        %%  Division ausgibt Bsp. 5:3=1 rest 2  --> mod(5,3)=2
        %%  wenn rest vorhanden wird Fehlermeldung ausgegeben und funktion
        %%  mit return verlassen
        %%  andernfalls wird mit Programm fortgefahren
if (mod(1,dt)~=0)
    error('Ungültige Paramtereingabe von dt !!! n*dt~=1');
    return
end
        %%  Überprüfung der Eingegebenen Schaltzeiten auf korrekte Eingabe
        %%  lenght(feld) ermittelt die Länge vom Vektor feld
        %%  Schleife um jeden parameter mit seinem darauffolgenden zu
        %%  vergleichen
        %%  ist der folgende Schaltzeitpunkt grösser als der vorherige
        %%  --> fehlermeldung und abbruch der funktion
for n=1:(length(feld)-1)
    if (feld(n)>=feld(n+1))
        error('Ungültige Paramtereingabe der Schaltzeiten !!! t(n+1) <= t(n)');
        return
    end
end
        %% erzeugt x Vektor von feld(1) bis ende feld mit schrittweite dt
x=(feld(1):dt:feld(length(feld)));
        %% Aufsuchen (find) der Stellen wo x identisch mit den Feldparametern ist
        %% diese Stellennummern werden c nacheinander zugeordent
for m=1:(length(feld));
    c(m)=find(x==feld(m));
end


        %%  Schleife um zwischen den Zuständen bei den Schaltzeiten zu
        %%  unterscheiden
        %%  Bsp. beginn mit o=2
        %%  mod(2,2) hat den Rest 0 also Sprung mit switch in case 0
        %%  --> y wird von index c(1) bis index c(2) y0 zugewiesen
        %%  Die Stellen vor index c(1) werden mit Nullen automatisch
        %%  aufgefüllt
        %%  o=3
        %%  mod(3,2) hat Rest --> Sprung zu otherwise
        %%  --> y wird von index c(2) bis index c(3) 0 zugewiesen
for o=2:length(c)
    switch mod(o,2)
        case 0
            y((c(o-1)):c(o))=y0;
        otherwise
            y((c(o-1)):c(o))=0;
    end
end
        %%  Stoppt PC-Zähler und gibt Rechenzeit aus
toc
        %%  lösche figure falls vorhanden
clf;
```

```
        %%  plotte x gegen y Lienenbreite sei 3 ('linewidth', 3)
plot(x,y);
        %%  Achseneinstellungen beim Plotten xmin xmax ymin ymax
axis([min(x) max(x) -0.5 1.5]);
        %%  Gitterlinien an
grid on;
```

---

2. Funktionsaufruf mit resultierender Grafik

```
aufgabee(1, 0.001, [1 3 4 5 7 9 10]);
```



f)

1. Aufgabenstellung siehe Deckblatt

```
        %% lösche Workspace (alle Variablen)
clear
        %% Eingabe der Zeilenvektoren und Variablenzuweisung
a=[2 0 5]
a =
     2     0     5

b=[4 2 1]
b =
     4     2     1

c=[0 1 0]
c =
     0     1     0

        %%  Eingabe der Matrizen und Variablenzuweisung

A=[2 5 2 ; 4 34 8 ; 4 5 2]
A =
     2     5     2
     4    34     8
     4     5     2

B=[2 4 0 ; 3 2 0 ; 6 2 0 ]
B =
     2     4     0
     3     2     0
     6     2     0

        %%  Vektorprodukt a*b
        %%  b' Transponiert Vektor b in einen SpaltenVektor
        %%  Damit Spaltenanzahl von a = Zeilenanzahl von b
a*b'
ans =
    13

        %%  Vektor Addition von a+b
a+b
ans =
     6     2     6

        %%  Matrize A mal Vektor b
        %%  b' damit Spaltenanzahl von A = Zeilenanzahl von b
A*b'
ans =
    20
    92
    28
```

```
        %%  Transponierte Matrix A mal Vektor c
        %%  c' damit Spaltenanzahl von A' = Zeilenanzahl von c
A'*c'
ans =
     4
    34
     8
        %%  Determinante von A
det(A)
ans =
   -56

        %%  Determinante von B
det(B)
ans =
     0

        %%  Inverse Matrix von A
inv(A)
ans =
   -0.5000         0    0.5000
   -0.4286    0.0714    0.1429
    2.0714   -0.1786   -0.8571

        %%  Inverse Matrix von B
        %%  Nicht Berechenbar da det(B)=0
inv(B)
Warning: Matrix is singular to working precision.
(Type "warning off MATLAB:singularMatrix" to suppress this warning.)
> In D:\FH\Matlab_übungen\Exercise1\aufgabef.m at line 34

ans =
   Inf   Inf   Inf
   Inf   Inf   Inf
   Inf   Inf   Inf

>>
```
