# Module 5033 Digitale Signalverarbeitung Dozent: Rolf Vetter

# 

### *Praktische Übung No4*

# Faltung, Bodediagramm

### Ziel

* Digitale Faltung
  + Verstehen, berechnen und programmieren
* Entwickeln und programmieren einer Methode zur Ermittlung des Bodediagramms

### Übung

1. Schreiben Sie eine Funktion y=conv\_p(x,g), welche die Faltung des Signals x(n) mit g(n) mittels Skalarprodukt berechnet.
   1. Testen Sie die Funktion mit
      1. x=[1 0 1 0 1 0 1 0 1]T und gd=[1 -1]T
      2. x=[1 0 1 0 1 0 1 0 1]T und gr=[1 1]T
      3. x=[1 0 1 0 1 0 1 0 1]T und gr=[1 0 0 0 0 1]T
   2. Überprüfen Sie die Resultate « von Hand » und mit der Funktion conv() von MATLAB.
   3. Man nehme einen Dirac und g=[1 -0.9 0.8 -0.7 0.6…..0 ]T und berechne die Faltung mit conv\_p(), conv() und « von Hand ». Was stellen Sie fest?
2. Optional: Schreiben Sie eine Funktion BodeDiagramm(g,fmin,df,fmax), welche das Bodediagramm der Impulsantwort g darstellt. Man wende die Methode an, die auch im Praktikum Signale und Systeme 4.4 für die Identifizierung der Übertragungsfunktion gebraucht wird.

Hinweis: Nacheinander folgend werden harmonische Signale mit einer Amplitude 1 und einer Frequenz die von f= fmin, fmin+df, fmin+2df,…fmax variiert mit der Impulsantwort g gefaltet. Durch das Messen der Amplitude des Ausgangssignals erhält man |G(fmin, fmin+df, fmin+2df,…fmax)|.

Ergebnisse

# Aufgabe 1

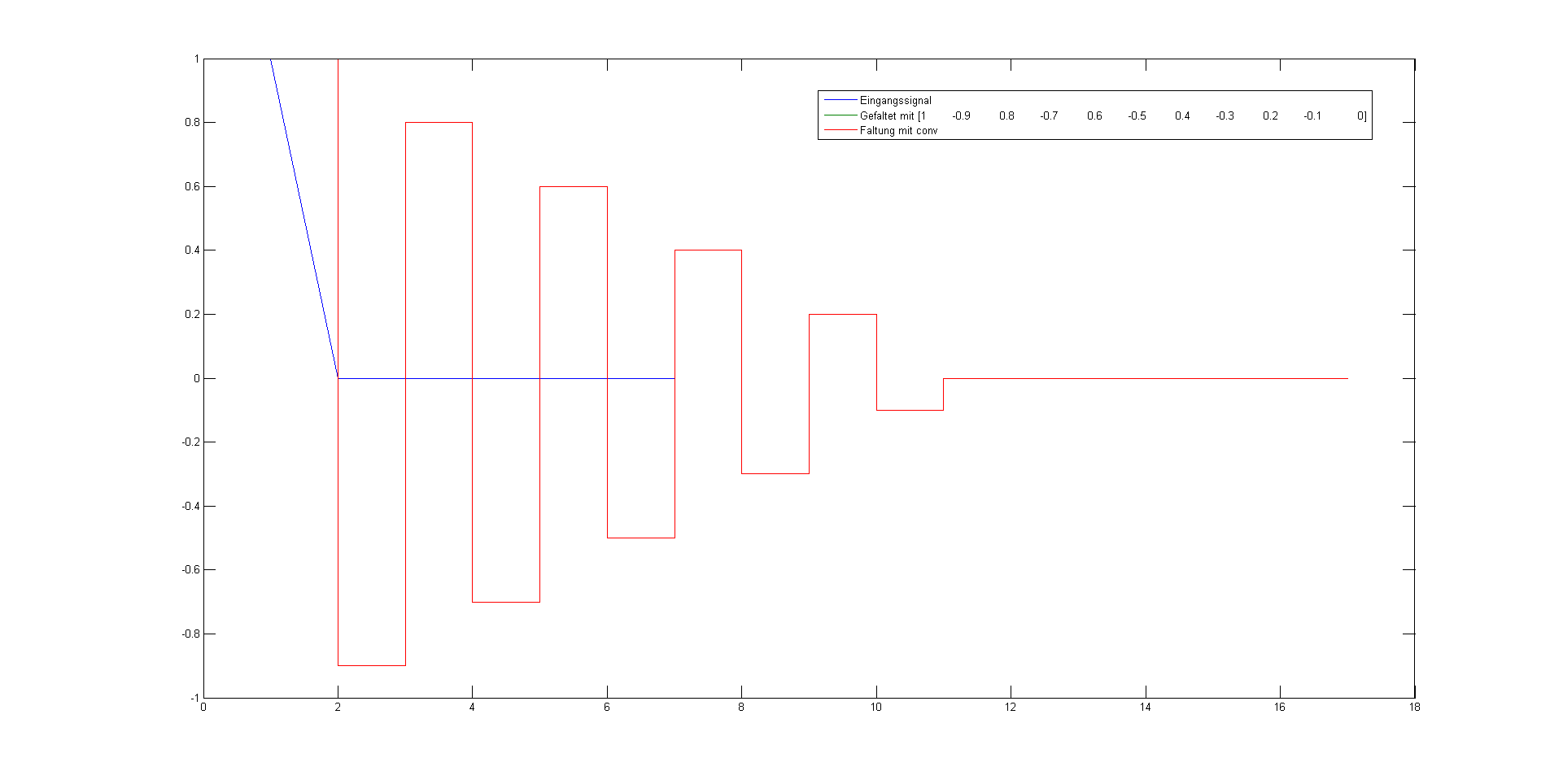
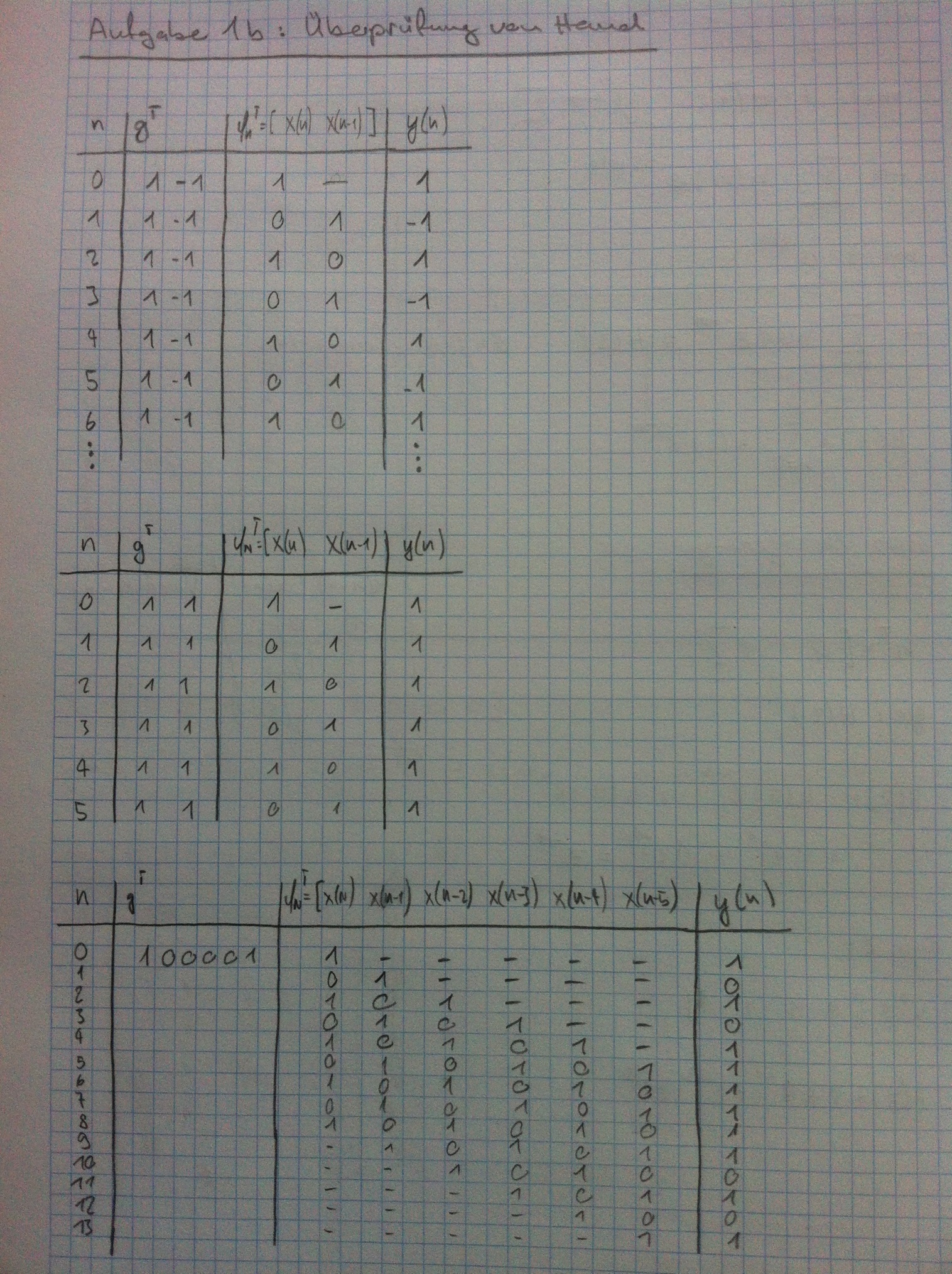
Matlab Code für Faltung mit Skalarprodukt und anschliessender Vergleich mit Matlabfunktion.

|  |
| --- |
| %% Digitale Faltung    close all;  clear all;    % X-Vektor  x = [1 0 1 0 1 0 1 0 1 ]';    % Auswahl an Übertragungsfunktionen  g = [1 -1]';  % g = [1 1]';  % g = [1 0 0 0 0 1]';      N = length(x);  p = length(g)-1;  y = zeros(1,N+p);    % Zeropadding  x\_zp = [zeros(1, p) x' zeros(1, p)]';    % Berechnung der Faltung für alle Stellen  for n = p+1:N+2\*p;  phi\_n = x\_zp(n:-1:n-p);  y(n-p) = g'\*phi\_n;  end    % Kontrolle mit Matlabfunktion (Parameter für Fkt sind: 'full', 'valid', 'same')  y\_k = conv(x, g, 'full');      % Ausgabe an Konsole  disp('Eigene Fkt.: '); y  disp('Matlab Fkt.: '); y\_k'    % Darstellung der Kurve  plot(x);  hold all;  stairs(y);  stairs(y\_k);    legend('Eingangssignal', ['Gefaltet mit [',int2str(g') ,']'], 'Faltung mit conv'); |

Damit man die gesamte Funktion falten kann, muss das Signal mit Nullen erweitert werden. Matlab macht dies, wenn man der Funktion *conv* das Argument ´full´ übergibt.

Die Funktion berechnet für jede Position des Signals das Skalarprodukt mit der Impulsantwort und dem gedrehten Ausschnitts des Signals.

Die Funktion wurde mit der Matlabfunktion conv und von Hand getestet. Dabei stellte sich heraus, dass sie funktioniert. Jedoch stellte sie zuerst nur einen Teil des Ergebnisses dar. Wir erweiterten die Berechnung mit dem Zeropadding und nun funktioniert es einwandfrei.



Gibt man als Signal einen Dirac-Impuls ins System hinein, kann man am Ausgang gerade die Impulsantwort ablesen. Das ist hier gut ersichtlich.