# MATLAB - Signalverarbeitung

## Themen: Faltung – Stoßantwort von Systemen – Quantisierung

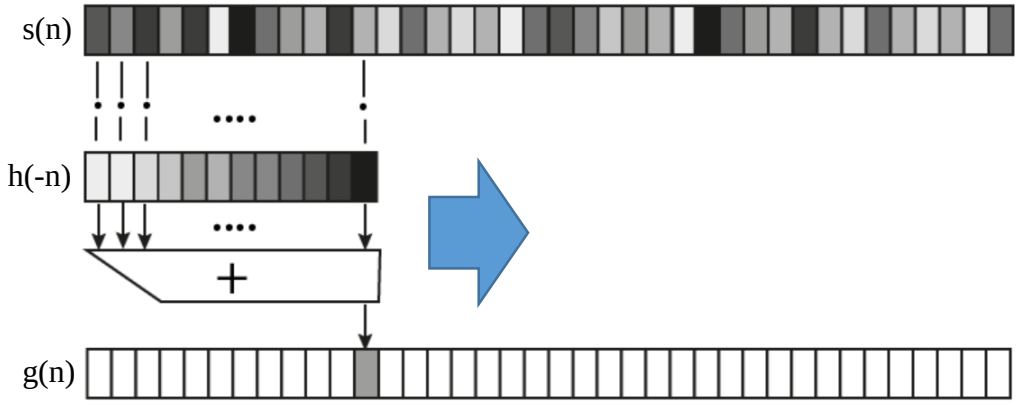
### 1. Faltungshall (Convolutional Reverb)

Entsprechend der Signaltheorie erhält man die Antwort eines Systems auf ein beliebiges Signal (hier Musik) durch die Faltung des Signals mit der Stoßantwort des Filters (hier die Impulsantwort eines Raumes).

Signal s(n)

Impulsantwort h(n)

Faltung




a) Hören Sie sich das unbearbeitete Signal s(n) an (*GitRiff.wav*).
b) Hören Sie sich die Stoßantwort des Raums h(n) an (*TrigRoom.wav*).
c) Schätzen Sie die Anzahl der durchzuführenden Multiplikationen ab.
d) Prüfen Sie die Samplefrequenz und die Größe von s(n) und h(n)
e) Schreiben Sie ein Matlab-Skript (s. nächste Seite) „*Faltungshall.m*“, welches die Systemantwort bestimmt, also: $g(n) = s(n) * h(n)$ und lassen Sie es ablaufen.
**Anm.**: Die Durchführung des Skriptes kann einige Minuten dauern (s. Punkt c).

**Anforderungen**:

- Gute Variablennamen sowie Kommentare
- keine „*Magic Numbers*“

**Nützliche Befehle**: pause, size(), zeros(), sound()

Matlab-Skript „*Faltungshall.m*“ :

a) Laden der Audiodatei „*GitRiff.wav*“
b) Laden der Audiodatei „*TrigRoom.wav*“
c) Größenbestimmung der Dateien,
d) evtl. Speicher für das gefilterte Signal anlegen,
e) Faltung durchführen
   Variante 1: Schreiben Sie eine eigene Funktion MyConv(s_Vek, h_Vek)
   Variante 2: Verwenden Sie die Matlab-Funktion conv(..)
f) Ergebnis Normalisieren
g) Audioausgabe der ungefilterten Datei s(n),
h) Weiter mit „*pause*“ (mit Cursor vor Return in Command-Window gehen)
i) Audioausgabe der Systemantwort g(n).

**Anm.**: Testen Sie mit der Matlab-conv()-Funktion auch die Stoßantworten „*Church.wav*“ und „*InTheSilo.wav*“.

**Abnahme**: Vorführung und Codereview

## 2. Faltungsecho (Convolutional Delay)

Jetzt soll ein Echo mit Hilfe von Faltung realisiert werden. Die Echoverzögerungen und Echostärken sollen in einer Matrix (Echomatrix) konfigurierbar sein (beliebige Anzahl):

```
% ------------------------------
%              Position   Echo-
%              in Sek.    höhe
Echos       = [  0         1.0; ...
                 0.3       0.3; ...
                 0.5       0.2; ...
                 0.7       0.1; ...
                 0.75      0.1];
```

Hieraus soll automatisch eine Faltungsmaske berechnet werden, die anschließend auf das Signal („*GitRiff.wav*“) angewendet wird. Die Länge der Faltungsmaske soll sich automatisch an die Einträge in der Echomatrix anpassen.

Schreiben Sie hierfür ein Skript „*Faltungsecho.m*“.

**Nützliche Befehle**: pause, size(), zeros(), sound(), conv()

## 3. Quantisierungsrauschen

Bei der Analog-Digital-Wandlung eines wertkontinuierlichen Signals kommt es unweigerlich zu Quantisierungsfehlern (z.B. 10bit-AD-Wandler = 1024 Quantisierungsstufen). Die Auswirkungen hiervon sollen in diesem Versuch untersucht werden.

Schreiben Sie ein Matlab-Skript „*Quantisierung.m*“ mit folgenden Schritten:

a) Laden der Audiodatei „*GitRiff.wav*“ nach s(n),
b) Signal s(n) normalisieren,
c) Quantisierung des Signals → q(n) mit

```
% Quantisierung
QSteps = 100;
q = round(s*QSteps);
```

d) Ergebnis q(n) normalisieren
e) Audioausgabe von q(n)
f) subplot q(n) und s(n) im Intervall 15000:15400 (s und q verschiedenfarbig)
g) subplot des max. Quantisierungsfehlers s(n)-q(n) im Intervall 15000:15400
h) Ausgabe des maximalen Absolutwertes von s(n)-q(n) (=max. Quantisierungsfehler)

**Dokumentation**:

- Beschreibung des Versuchszwecks
- Tabelle: max. Quantisierungsfehler in Abhängigkeit von den Quantisierungsstufen (QSteps = 10, 20, 50, 100, 200, 500)
- kurze Bewertung