

Verlässliche Systemsoftware

Übungen zur Vorlesung

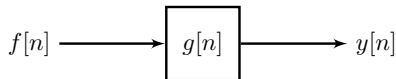
Hinweise zur Aufgabe: Filter

Phillip Raffeck, Simon Schuster, Peter Ulbrich

Technische Universität Dortmund
Lehrstuhl für Informatik 12 (Arbeitsgruppe Systemsoftware)
<https://sys.cs.tu-dortmund.de>

Wintersemester 2020





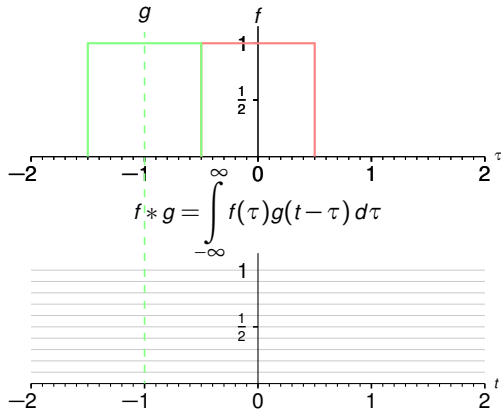
- Objekte identifizieren (z.B. Eingaben)
- Implementierung der Filterung durch **Faltung** (engl. convolution) mit Impulsantwort
 - f Signalwerte, g Filterwerte

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n - m] \quad (1)$$

- Impulsantwort: Reaktion des Systems auf Dirac-Impuls
- Zunächst Verwendung von `float`, anschließend **Festkommaformat**

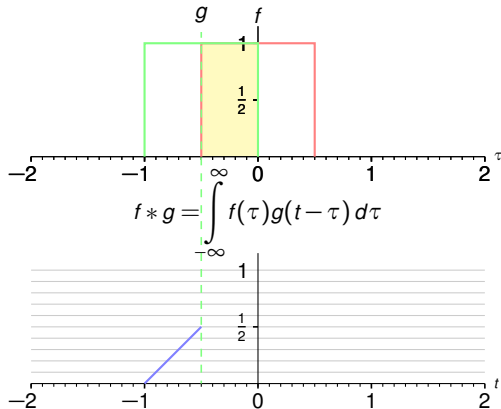


Beispiel: Faltung



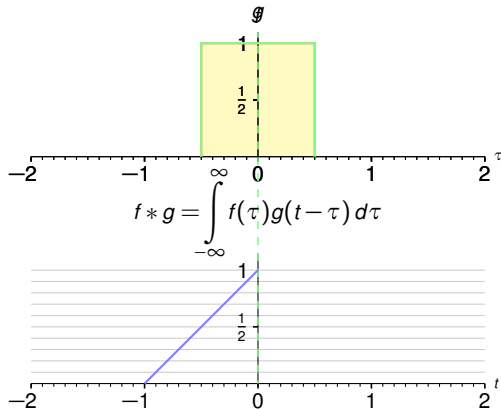
Beispiel: Faltung

$$\text{■ } f(\tau)g(-0.5-\tau)$$



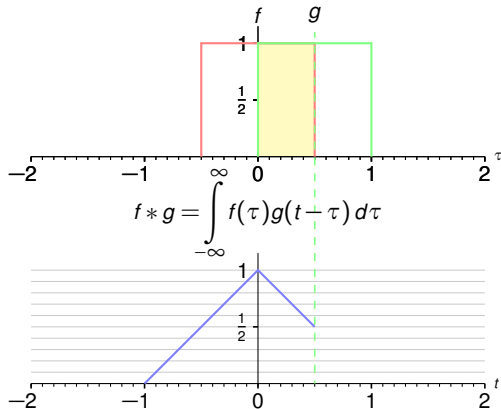
Beispiel: Faltung

$$\text{■ } f(\tau)g(0-\tau)$$



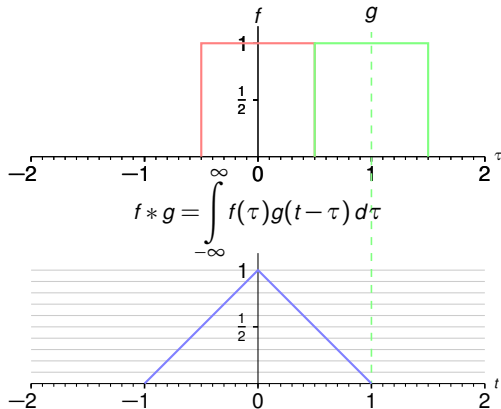
Beispiel: Faltung

$$\text{■ } f(\tau)g(0.5 - \tau)$$

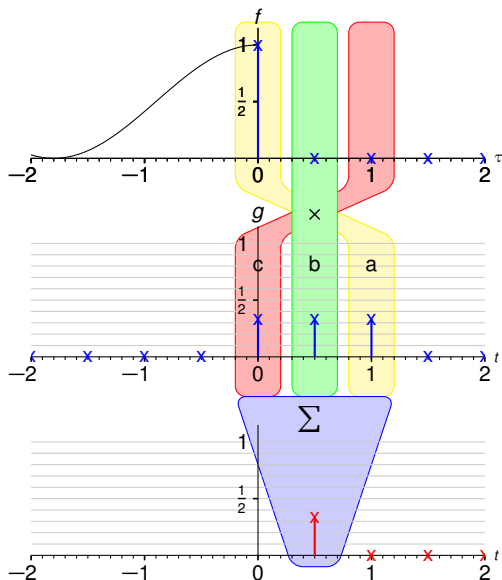


Beispiel: Faltung

$$\text{■ } f(\tau)g(1-\tau)$$



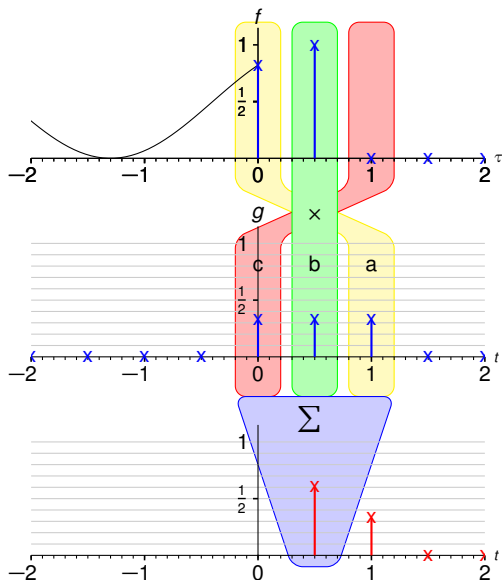
Beispiel: Diskrete Faltung



$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

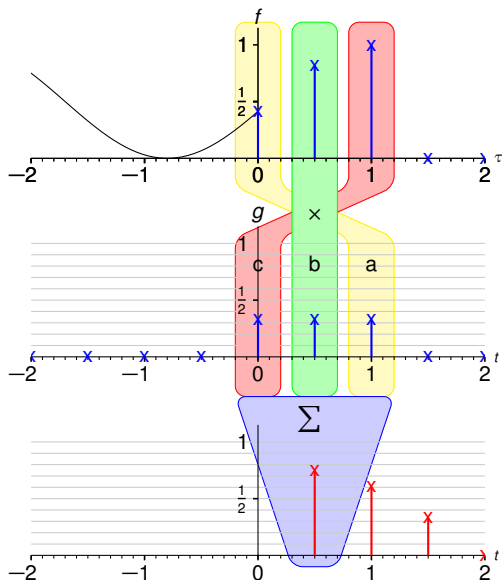


Beispiel: Diskrete Faltung



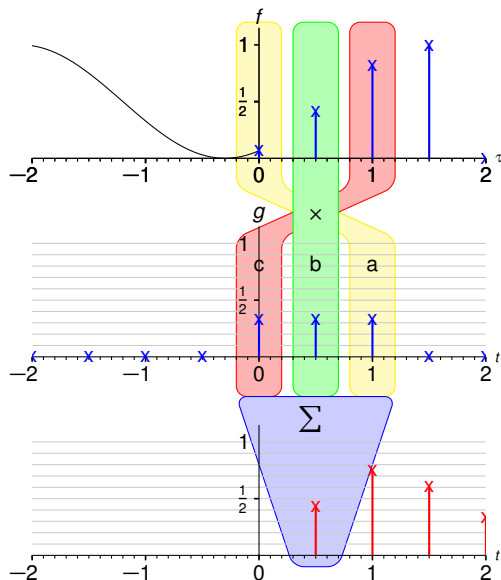
$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

Beispiel: Diskrete Faltung



$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

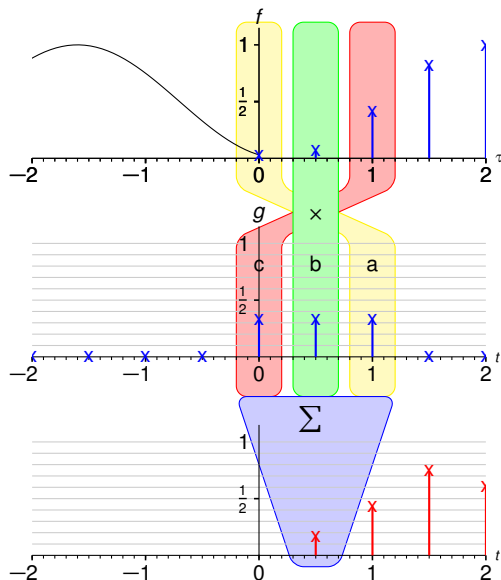
Beispiel: Diskrete Faltung



$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$



Beispiel: Diskrete Faltung



$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$



Exkurs: Prüfsummen

- Verkürzte Repräsentation eines Datensatzes (Bsp.: Git Commithash)
- Aus Eingabedaten errechnet
- Nutzen: Überprüfung der Datenintegrität
- Anforderungen
 - Stabil
 - Effizient berechenbar
 - Zuverlässige Fehlererkennung
- Beispiel: Quersumme mit Zehner-Restklasse:

$$\text{CHECK}(12345) = (1 + 2 + 3 + 4 + 5) \bmod 10 = 5$$



Exkurs: Prüfsummen

- Verkürzte Repräsentation eines Datensatzes (Bsp.: Git Commithash)
- Aus Eingabedaten errechnet
- Nutzen: Überprüfung der Datenintegrität
- Anforderungen
 - Stabil
 - Effizient berechenbar
 - Zuverlässige Fehlererkennung
- Beispiel: Quersumme mit Zehner-Restklasse:

$$\text{CHECK}(12345) = (1 + 2 + 3 + 4 + 5) \bmod 10 = 5$$

$$\text{CHECK}(123\textcolor{red}{5}5) = (1 + 2 + 3 + \textcolor{red}{5} + 5) \bmod 10 = \textcolor{red}{6}$$

~> Fehlermodell: Schützt vor allen Einzifferfehlern



Exkurs: Prüfsummen

- Verkürzte Repräsentation eines Datensatzes (Bsp.: Git Commithash)
- Aus Eingabedaten errechnet
- Nutzen: Überprüfung der Datenintegrität
- Anforderungen
 - Stabil
 - Effizient berechenbar
 - Zuverlässige Fehlererkennung
- Beispiel: Quersumme mit Zehner-Restklasse:

$$\text{CHECK}(12345) = (1 + 2 + 3 + 4 + 5) \mod 10 = 5$$

$$\text{CHECK}(123\textcolor{red}{5}5) = (1 + 2 + 3 + \textcolor{red}{5} + 5) \mod 10 = \textcolor{red}{6}$$

↪ Fehlermodell: Schützt vor allen Einzifferfehlern

↪ jedoch bspw. kein Schutz gegen Vertauschung:

$$\text{CHECK}(123\textcolor{red}{5}4) = (1 + 2 + 3 + \textcolor{red}{5} + \textcolor{red}{4}) \mod 10 = 5$$



Exkurs: Prüfsummen

- Verkürzte Repräsentation eines Datensatzes (Bsp.: Git Commithash)
- Aus Eingabedaten errechnet
- Nutzen: Überprüfung der Datenintegrität
- Anforderungen
 - Stabil
 - Effizient berechenbar
 - Zuverlässige Fehlererkennung
- Beispiel: Quersumme mit Zehner-Restklasse:

$$\text{CHECK}(12345) = (1 + 2 + 3 + 4 + 5) \mod 10 = 5$$

$$\text{CHECK}(123\textcolor{red}{5}5) = (1 + 2 + 3 + \textcolor{red}{5} + 5) \mod 10 = \textcolor{red}{6}$$

↪ Fehlermodell: Schützt vor allen Einzifferfehlern

↪ jedoch bspw. kein Schutz gegen Vertauschung:

$$\text{CHECK}(123\textcolor{red}{5}4) = (1 + 2 + 3 + \textcolor{red}{5} + \textcolor{red}{4}) \mod 10 = 5$$

■ Wichtig: Fehlermodell, welche Arten von Bitfehlern werden erkannt?

- **Einzelnen Filterschritt** implementieren (kein Burst-Filter)
- Verwendung von **Q-Notation**
- Aspekte:
 - Nutzung abstrakter Schnittstellen
 - Einfluss von Schnittstellen auf Verlässlichkeit
 - Entwurfsentscheidungen und -abwägung in der Systemimplementierung

