3. Diskrete Fourier-Transformation

- Definition:
  - Gegeben ist die Zeitreihe $x_n = x(n\Delta t), \quad n = 0, \ldots, N-1.$
  - Die *diskrete Fourier-Transformation* der Zeitreihe ist definiert durch

$$\boxed{X_k = \sum_{n=0}^{N-1} x_n e^{-2\pi i k n / N}, \quad k = 0, \ldots, N-1}$$

- Eigenschaften:
  - Die N reellen Werte der Zeitreihe werden auf N komplexe Werte abgebildet.

3. Diskrete Fourier-Transformation

- Durch die für $k = 0, \dots N-1$ berechneten Werte liegen die Werte für beliebige k fest:
  - Die diskrete Fourier-Transformation ist periodisch mit der Periode N:

$$X_{k+N} = \sum_{n=0}^{N-1} x_n e^{-2\pi i (k+N) n/N} = \sum_{n=0}^{N-1} x_n e^{-2\pi i k n/N} e^{-2\pi i n} = X_k$$

  - Für negative k gilt: $X_{-k} = \bar{X}_k$
  - Daraus folgt: $X_{N-k} = X_{-k+N} = X_{-k} = \bar{X}_k$
  - Für $k = N/2$ folgt: $X_{N/2} = X_{-N/2+N} = X_{-N/2} = \bar{X}_{N/2}$

3. Diskrete Fourier-Transformation

- Inverse Transformation:
  - Für die inverse Transformation muss das lineare Gleichungssystem

$$X_k = \sum_{n=0}^{N-1} x_n e^{-2\pi i k n/N}, \quad k = 0, \ldots, N-1$$

nach x_n aufgelöst werden.

  - Zunächst gilt:

$$X_k e^{2\pi i k \nu/N} = \sum_{n=0}^{N-1} x_n e^{-2\pi i k (n-\nu)/N}, \quad \nu = 0, \ldots N-1$$

3. Diskrete Fourier-Transformation

- Summation über k ergibt:

$$\sum_{k=0}^{N-1} X_k e^{2\pi i k \nu/N} = \sum_{k=0}^{N-1} \sum_{n=0}^{N-1} x_n e^{-2\pi i k(n-\nu)/N} = \sum_{n=0}^{N-1} x_n \left(\sum_{k=0}^{N-1} e^{2\pi i k(\nu-n)/N} \right)$$

- Für die Summe in der Klammer gilt:
  - Für $\nu = n$: $$\sum_{k=0}^{N-1} e^{2\pi i k(\nu-n)/N} = \sum_{k=0}^{N-1} 1 = N$$
  - Für $\nu \neq n$: $$\sum_{k=0}^{N-1} e^{2\pi i k(\nu-n)/N} = \sum_{k=0}^{N-1} \left(e^{2\pi i(\nu-n)/N} \right)^k = \frac{1-\left(e^{2\pi i(\nu-n)/N} \right)^N}{1-e^{2\pi i(\nu-n)/N}}$$ $$= \frac{1-\left(e^{2\pi i} \right)^{\nu-n}}{1-e^{2\pi i(\nu-n)/N}} = 0$$

3. Diskrete Fourier-Transformation

- Dabei wurde die Formel für die Summe einer geometrischen Reihe verwendet:

$$\sum_{k=0}^{N-1} q^k = \frac{1-q^N}{1-q}$$

- Damit folgt: $$\sum_{k=0}^{N-1} X_k e^{2\pi i k n/N} = N x_n$$

- Damit gilt für die inverse Transformation:

$$\boxed{x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{2\pi i k n/N}, \quad n = 0, \ldots, N-1}$$

3. Diskrete Fourier-Transformation

- Schnelle Fourier-Transformation:
  - Für die Berechnung der bei der diskreten Fourier-Transformation auftretenden Summen wurden effiziente Algorithmen entwickelt, die als schnelle Fourier-Transformation (Fast Fourier Transformation, FFT) bezeichnet werden:
    - Gauß 1805
    - Runge 1903
    - Danielson und Lanczos 1942
    - Cooley und Tukey 1965
    - Winograd 1978

3. Diskrete Fourier-Transformation

- Berechnung mit GNU Octave:
  - Die Funktion

```
X = fft(x, N)
```

berechnet die diskrete Fourier-Transformation.

Eingabe:	**x**	Feld mit den Werten der Zeitreihe
	N	Anzahl der Werte (optional)
Ausgabe:	**X**	Feld mit den Werten der Fourier-Transformation

Ist **N** größer als die Anzahl der Elemente von **x**, so wird mit 0 aufgefüllt. Ist **N** kleiner, wird abgeschnitten.

3. Diskrete Fourier-Transformation

- Die Funktion

```
x = ifft(X)
```

berechnet die inverse diskrete Fourier-Transformation.

Eingabe:	**X**	Feld mit den Fourier-Koeffizienten (wie von **fft** ausgegeben)
Ausgabe:	**x**	Feld mit den Werten der Zeitreihe

- Beispiel:
  - Das folgende Skript berechnet die diskrete Fourier-Transformation der Zeitreihe

$$x_n = \sin^2(\pi n / N), \quad n = 0, \ldots, N-1$$

3. Diskrete Fourier-Transformation

```
# Kapitel 3.3: Diskrete Fourier-Transformation
#
# ---------------------------------------------------------------

  N = 12;

# Zeitreihe
  n = 0 : N -1;
  x = sin(pi * n / N).^2;

# Diskrete Fourier-Transformation
  X = fft(x);

# Rücktransformation
  xr = ifft(X);

# Fehler
  e = max(abs(x - xr));
  printf("Abweichung: %11.4e\n", e);
```

3. Diskrete Fourier-Transformation

```
# Darstellung

  figure(1, "position", [100, 500, 1000, 700],
            "paperposition", [0, 0, 25, 14]);
  subplot(2, 1, 1);
     plot(n, x, "or", n, xr, "*b");
     legend("x", "x_rn");
     grid;
     xlabel("n");
  subplot(2, 1, 2);
     plot(n, real(X), "or", n, imag(X), "*b");
     legend("Re(X)", "Im(X)");
     grid;
     xlabel("k");

  print("v3_3.svg", "-dsvg");
```

Ausgabe: **`Abweichung:  1.1102e-16`**

3. Diskrete Fourier-Transformation