Berufsakademie Sachsen
Staatliche Studienakademie Leipzig

Anwendung der diskreten Sinustransformation

Hausarbeit
zur Erlangung des akademischen Grades eines
Bachelor of Science (B. Sc.)
in der Studienrichtung Informatik

Eingereicht von: Simon Nikolaidis
xxxxxxxxxxxxxxx
xxxxxxxxxxxxxxx
xxxxxxxxxxxxxxx
xxxxxxxxxxxxxxx

Eingereicht am: 24.09.2023

Inhaltsverzeichnis

1 Einleitung

Diese Arbeit befasst sich mit der diskreten Sinustransformation (DST), welche in den Bereich der Numerik einzuordnen ist. Die Numerik ist ein Teilgebiet der Mathematik, welches sich mit der Konstruktion und Analyse von Algorithmen für kontinuierliche mathematische Probleme beschäftigt.
Zuerst sollen die Grundlagen der diskreten Sinustransformation beschrieben werden. Anschließend soll dann die Anwendung der DST anhand einer praktischen Berechnung dargestellt werden.

Transformationen werden in erster Linie zur Reduzierung der Komplexität in mathematischen Problemen eingesetzt. Hierunter fallen zum Beispiel Differentiationen, Integrationen oder Filteroperationen. Dazu werden Funktionen von einem Raum in einen anderen überführt, ohne dass dabei Informationen über diese Funktion verloren gehen. Sie spielen beispielsweise in der Signalverarbeitung zur Analyse des Informationsgehaltes eines Signals eine wichtige Rolle.

2 Diskrete Sinustransformation

Die diskrete Sinustransformation ist eine Fourier-ähnliche Transformation. Sie ist außerdem eng mit der diskreten Kosinustransformation (DCT) verwandt, basiert jedoch auf der ungeraden Sinusfunktion.
Sowohl die DCT als auch die DST wurden 1974 von Nasir Ahmed, T. Natarajan und K.R. Rao beschrieben.[1] Die DST vom Typ II (DST-II), welche besonders relevant für diese Arbeit ist, wurde 1978 von H.B. Kekra und J.K. Solanka entwickelt.[2]

2.1 Definition

Die DST dient dazu, eine reellwertige Eingabefolge aus beispielsweise dem Orts- oder Zeitraum, bestehend aus N Elementen $x[n]$, in eine entsprechende reellwertige Ausgabefolge im Impuls- beziehungsweise Frequenzraum (Spektralbereich), repräsentiert durch $X[n]$, zu überführen [3]:

$$x[n] = x_0, \ldots, x_{N-1} \Rightarrow X[n] = X_0, \ldots, X_{N-1}$$

Ein wichtiges Merkmal dieser Transformationen ist, dass sie reversibel sind. Das bedeutet, dass eine entsprechende Rücktransformation existiert, die in der Lage ist, die transformierten und möglicherweise bearbeiteten Daten wieder in den ursprünglichen Raum zu überführen.

Die DST nutzt zur Berechnung ausschließlich die Sinusfunktion und ermöglicht es, das ursprüngliche Signal in eine Summe von sinusförmigen Komponenten unterschiedlicher Frequenzen aufzuteilen. Dadurch ist es besonders nützlich für die Analyse oder Bearbeitung realer Daten, die eine ungerade Symmetrie aufweisen.[3][4]
Die aus der diskreten Sinustransformation (DST) gewonnenen Frequenzkoeffizienten zeigen dabei, wie stark die einzelnen sinusförmigen Anteile im ursprünglichen Signal vertreten sind.
Die DST besitzt folgende Merkmale. Sie ist:

Merkmale [3]:

- **reellwertig:**
  Im Gegensatz zur diskreten Fouriertransformation (DFT), arbeitet man bei der DST und DCT statt mit der Entwicklungsfunktion e^{ikx} mit den trigonometrischen Funktionen $sin(kx)$ und $cos(kx)$. Es wird also eine rein reelle Matrix genutzt und dadurch von Anfang an nur mit reellen Zahlen gerechnet. Dabei „extrahiert“ beziehungsweise „imitiert“ die DST den Imaginärteil der DFT. Diese Zerlegung ergibt sich aus der folgenden Relation [5]:

$$cos(x) = \mathfrak{Rm}(e^{ix}),\ sin(x) = \mathfrak{Im}(e^{ix})$$

- **diskret:**
  Die DST wird auf einen endlichen Bereich diskreter Daten angewendet. Ein Beispiel hierfür sind die Werte einer Funktion an einer diskreten Menge von Stützpunkten. Es handelt sich also immer um abzählbar viele isolierte Werte. Dies resultiert oft aus der Gegebenheit von diskreten Messwerten, die anschließend weiterverarbeitet werden müssen.

- **linear:**
  - homogen: $DST(ax) = a * DST(x)$
  - additiv: $DST(x + y) = DST(x) + DST(y)$

2.2 Verschiedene Formen

Es gibt insgesamt acht verschiedene Formen der DST. Die verschiedenen Formen von DST-I bis DST-IV entsprechen im Wesentlichen der reellwertigen, ungeraden DFT mit gerader Ordnung, wobei lediglich ein konstanter Faktor zu berücksichtigen ist.
Die verschiedenen Formen unterscheiden sich dadurch, wie ihre konkreten periodischen Fortsetzungen realisiert werden. Jede Grenze kann entweder gerade oder ungerade sein und sie kann symmetrisch um einen Datenpunkt oder den Punkt zwischen 2 Datenpunkten sein. Daraus ergeben sich $2 * 2 * 2 * 2 = 16$ Möglichkeiten. Die Hälfte dieser Möglichkeiten, d. h. diejenigen, bei denen die linke Grenze ungerade ist, entsprechen den 8 Arten der DST. Diese unterschiedlichen Randbedingungen wirken sich stark auf die Anwendungen der Transformation aus.[6] Die Varianten DST-V bis DST-VIII scheinen jedoch in der Praxis kaum Verwendung zu finden.

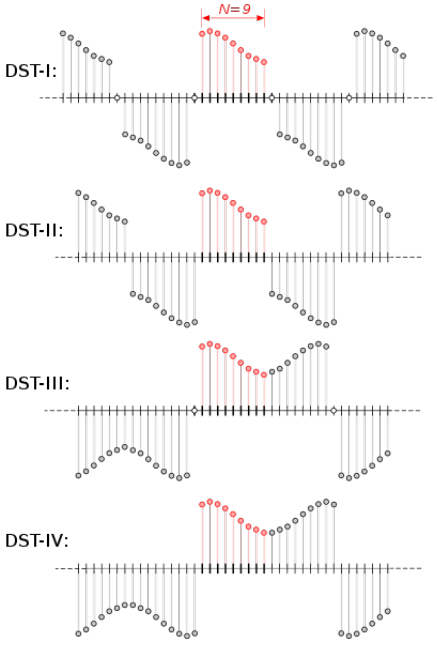


Abbildung 2.1: Unterschiedliche periodische Fortsetzungen bei DST-I bis DST-IV [3]

Beispielsweise ist für einen Satz diskreter Daten $u_n, n = 0, \ldots, N-1$ die DST-II [5]:

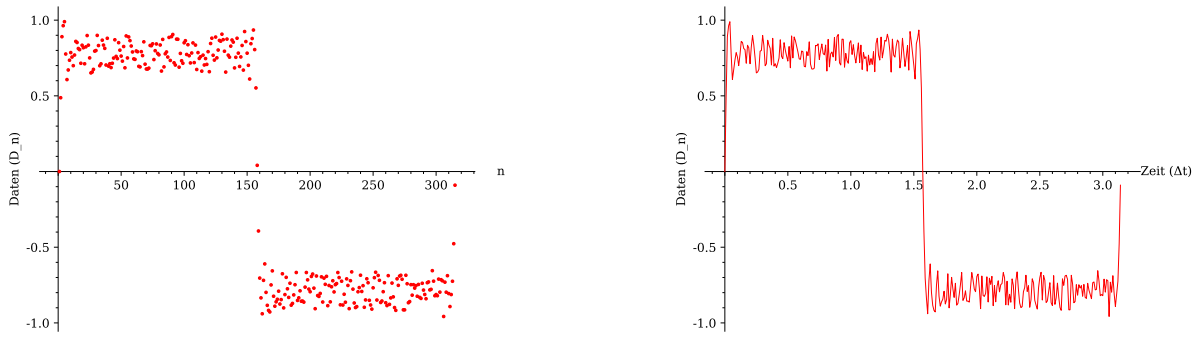
$$\omega_k = \mathfrak{S}_{II}(u)_k = \sum_{n=0}^{N-1} u_n \sin\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)(k+1)\right], \qquad k = 0, \ldots, N-1$$

2.3 Anwendungsbereiche

Die Fähigkeit der DST, ungerade Symmetrieeigenschaften zu erfassen und reellwertige Koeffizienten zu liefern, macht sie zu einem wertvollen Hilfsmittel in verschiedenen wissenschaftlichen und technischen Bereichen. Durch die Zerlegung eines Signals in sinusförmige Schwingungen können verschiedene Frequenzanteile des Signals extrahiert und analysiert werden. Somit könnte sie beispielsweise eingesetzt werden, um redundante Informationen in einem Signal zu reduzieren, indem nur die wichtigsten Frequenzanteile beibehalten werden, während unwichtige Frequenzen unterdrückt oder vernachlässigt werden. Dadurch kann beispielsweise auch ein Rauschen reduziert werden.
Häufig wird die DST zur Lösung von partiellen Differentialgleichungen mit Spektralmethoden eingesetzt. Solche Gleichungen dienen dazu, physikalische Vorgänge mathematisch zu modellieren. Hierbei erfolgt eine Umwandlung der physikalischen Darstellung eines Problems in den Spektralbereich. Die Unbekannten werden dabei von physikalischen Größen zu Spektralkoeffizienten transformiert.
Des Weiteren kommt die DST im Videokompressionsstandard „High Efficiency Video Coding“ (HEVC), auch bekannt als H.265, zum Einsatz.[7]
Zusammenfassend kann man also festhalten, dass die DST insbesondere in der Signalverarbeitung und der numerischen Analysis nützlich ist, um periodische oder periodisch fortgesetzte Datenreihen zu analysieren. Dadurch findet sie zum Beispiel Anwendung in der Bildverarbeitung, der Datenkompression und der Spektralanalyse.

3 Anwendung der DST

In diesem Kapitel wird die diskrete Sinustransformation an einem konkreten Beispiel angewendet und damit die Berechnung dieser veranschaulicht. Hierzu sei ein diskreter Datensatz $D = \{D_n\}_{n=1}^{N}$ in einer Datei **dstexample.dat** gegeben. Die Datenpunkte wurden mit einer Abtastrate von $\Delta t = 0.01$ erzeugt.



(a) Mit Δt abgetasteter Datensatz

(b) Als Funktion von n

Abbildung 3.1: Grafische Darstellung des Datensatzes $D = \{D_n\}_{n=1}^{N}$

3.1 Berechnung der DST

Anhand des vorliegenden Datensatzes wird nun die diskrete Sinustransformation $S_{T,k} = \mathfrak{S}(D)_k, k = 1, \ldots, N$ berechnet. Die dargestellten Daten zeigen eine gewisse Rechteckform, zusätzlich lässt sich bereits eine mögliche Periodizität erkennen. In diesem Fall scheint es sich um ein Signal bestehend aus einer Überlagerung von Sinusfunktionen zu handeln. Die zugrundeliegende Sinusfunktion ist asymmetrisch zur y-Achse und punktsymmetrisch bezüglich des Nullpunktes. Aufgrund dessen bietet sich hier die Verwendung der DST, genauer der Version II an.

$$\omega_k = \mathfrak{S}(D)_k = \frac{1}{\sqrt{N}} \sum_{n=1}^{N} D_n \sin\left[\frac{\pi}{N}\left(n - \frac{1}{2}\right)k\right], \qquad k = 1, \ldots, N \tag{3.1}$$

Der ursprüngliche Datensatz wird nun mithilfe eines Python-Programms dieser Transformation (3.1) unterzogen. Dadurch wird das komplexe Signal in seine grundlegenden Frequenzkomponenten zerlegt. Diese Frequenzkomponenten geben an, wie stark bestimmte Frequenzen im Signal vertreten sind. Durch die Verwendung der diskreten Sinustransformation bleiben alle Datenwerte reell. Folgende Grafik zeigt dies:

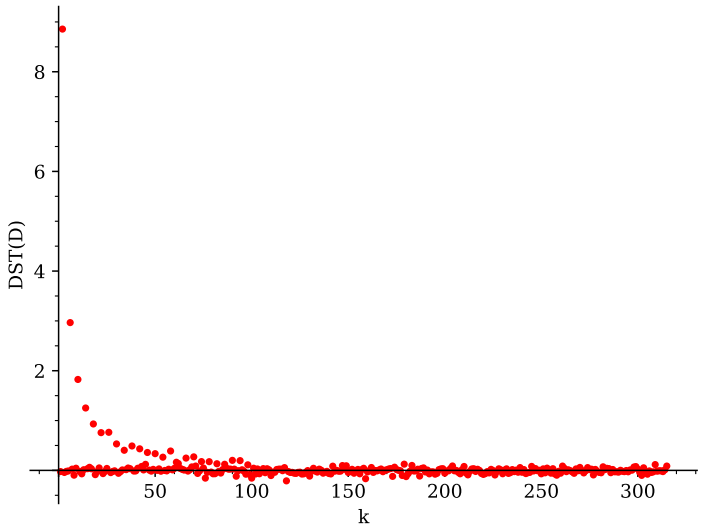


Abbildung 3.2: DST des Datensatzes, k $\hat{=}$ diskrete Frequenzzahl

```
def discrete_sine_transform(data):
    N = len(data)
    dst = np.zeros(N)
    for k in range(1, N + 1):
        sum_val = 0
        for n in range(1, N + 1):
            sum_val += data[n - 1] * np.sin((np.pi / N) * (n - 0.5) * k)
    dst[k - 1] = sum_val / np.sqrt(N)
    return dst
```

Listing 3.1: Python-Implementierung der DST-II

3.2 Filtern und Inverse

Es wurde die Liste $\mathfrak{S}(D)_k, k = 1, \ldots, N$ berechnet. Für die Filteroperationen wird die Liste $\mathfrak{S}(D)_k^{(m,n)}, k = 1, \ldots, N$ definiert, für welche gilt [5]:

$$\mathfrak{S}(D)_k^{(m_0,m_1)} = \begin{cases} 0 & k = 1, \ldots, m_0 - 1 \\ \mathfrak{S}(D)_k & k = m_0, \ldots, m_1 \\ 0 & k = m_1 + 1, \ldots, N \end{cases}$$

Das heißt, es werden in der vollständig diskreten Sinustransformierten $S_{T,k}$ alle Einträge $m_0 \leq m \leq m_1$ herausgefiltert. Das bedeutet, dass alle Werte außerhalb des definierten Bereiches auf Null gesetzt werden.

```
def filter_dst(dst, m0, m1):
    # dst-Array beginnend ab Index 0, aber eigentlich für k=1,...,N - Korrektur:
    start_idx = m0 - 1
    end_idx = m1 - 1
    filtered_dst = [data if start_idx <= i <= end_idx else 0 for i, data in
        enumerate(dst)]
    return filtered_dst
```

Listing 3.2: Implementierung der Filteroperation

Wie bereits erwähnt ist die Transformation reversibel. Es ergibt sich also folgende Rücktransformation:

$$D_n^{(m_0,m_1)} = \mathfrak{S}^{-1}\left(\mathfrak{S}(D)_k^{(m_0,m_1)}\right)_n$$

Die inverse diskrete Sinustransformation ($\mathfrak{S}^{-1}$) ergibt sich hierbei nicht „einfach“ durch Vorzeichenwechsel aus der Hintransformation, sondern [5]:

$$\mathfrak{S}^{-1}(\omega)_k = \frac{1}{\sqrt{N}}\left[2 * \sum_{n=1}^{N-1} \omega_n \sin\left(\frac{\pi}{N}\left(k - \frac{1}{2}\right)n\right) + (-1)^{(k-1)}\omega_N\right], \qquad k = 1, \ldots, N \quad (3.2)$$

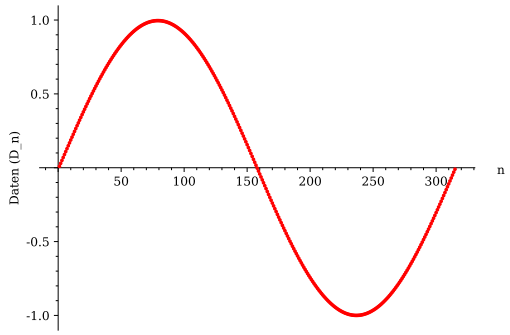
```
def inverse_discrete_sine_transform(dst_data):
    N = len(dst_data) # Off-by-one-Error beachten - dst_data beginnend ab Index 0
    data_reconstructed = np.zeros(N)

    for k in range(1, N + 1):
        sum_val = 0
        for n in range(1, N):
               sum_val += dst_data[n - 1] * np.sin((np.pi / N) * (k - 0.5) * n)
        sum_val = (2 * sum_val + (-1)**(k - 1) * dst_data[N - 1]) / np.sqrt(N)
        data_reconstructed[k - 1] = sum_val
    return data_reconstructed
```

Listing 3.3: Implementierung der inversen diskreten Sinustransformation

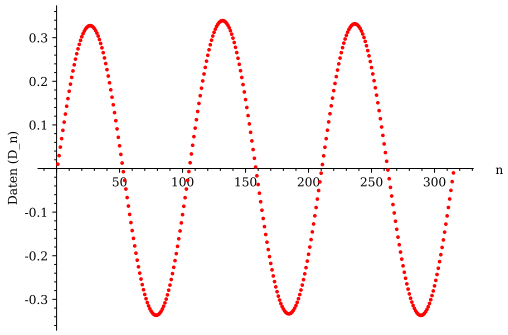
Diese Operationen werden nun für 4 verschiedene Filterbereiche auf die diskrete Sinustransformierte $S_{T,k}$ angewendet.

$$(m_0, m_1) = (1, 2), (3, 6), (7, 10), (1, N)$$

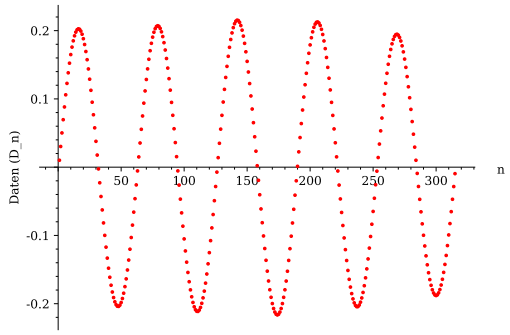
Die folgenden Abbildungen zeigen die bearbeiteten und rücktransformierten Daten im originalen t-Raum.



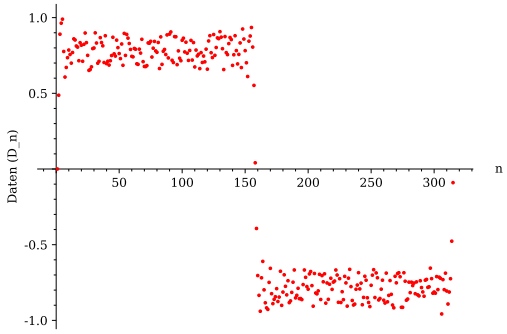
(a) $k = 1, 2$



(b) $k = 3, 4, 5, 6$



(c) $k = 7, 8, 9, 10$



(d) alle k

Abbildung 3.3: Die Rücktransformierte $D^{(m_0,m_1)}$ der DST mit (a) $m_0 = 1, m_1 = 2$, (b) $m_0 = 3, m_1 = 6$, (c) $m_0 = 7, m_1 = 10$, (d) $m_0 = 1, m_1 = N$

3.3 Charakterisierung der Resultate

In den Abbildungen (3.3) wird deutlich veranschaulicht, wie ein Großteil der Frequenzen herausgefiltert wurde, wodurch einzelne Frequenzbeträge des Gesamtsignals zu erkennen sind. Das sind Beispiele für sogenannte Frequenzbänder. Es handelt sich immer noch um eine Überlagerung einzelner sinusförmiger Anteile, allerdings sind hier deutlich weniger dieser Anteile zusammengefasst. In einem solchen Frequenzband werden eine bestimmte Anzahl von Frequenzen beziehungsweise ein Frequenzbereich des Signals hervorgehoben, während alle anderen Frequenzen außerhalb dieses Bereiches unterdrückt werden. Dieser resultierende Filter ist ein Bandbreitenfilter (oder auch Bandpass genannt). Mithilfe eines solchen Filters können bestimmte Frequenzen eines Signals selektiert werden. An diesen Abbildungen (3.3) kann man auch erkennen, wie groß der Einfluss der einzelnen Frequenzbereiche auf das Originalsignal ist beziehungsweise wie prägnant sie vertreten sind. Beispiele für solche Filter in der Praxis sind Hoch- oder Tiefpassfilter.
Abbildung (3.3(d)) zeigt das Signal, auf welches keine Filteroperationen angewendet wurden. Hier wird deutlich, dass das ursprüngliche Signal mit all seinen Frequenzanteilen

wieder hergestellt werden konnte. Im Gegensatz dazu werden in den anderen Abbildungen lediglich bestimmte Frequenzbereiche des Signals selektiert und dargestellt.
Zusätzlich ist an dieser Stelle anzumerken, dass die Amplituden der einzelnen Frequenzbänder nicht konstant sind. Um die Korrektheit dieser Ergebnisse zu gewährleisten, wurden die durchgeführten Berechnungen mehrfach mit unterschiedlichen eigenen Programmen und externen Bibliotheken wie *SciPy* geprüft. Verschiedene Faktoren wie numerische Genauigkeit, die Beschaffenheit der Ausgangsdaten und die Wechselwirkung der einzelnen Komponenten im Frequenzbereich könnten mögliche Gründe für diese Variationen sein.

Laufzeitkomplexität:
Die DST-Operationen, wie die Anwendung der Transformation und Rücktransformation, weisen eine Laufzeitkomplexität von $\mathcal{O}(n^2)$ auf. Dies resultiert aus der Notwendigkeit, aufgrund der Summe, den gesamten Datensatz für jeden individuellen Datenpunkt zu durchlaufen. Es existieren jedoch analog zur Schnellen Fourier-Transformation Algorithmen, wie der Cooley-Turkey-Algorithmus, die auch auf die DST angewendet werden können und die Laufzeit auf $\mathcal{O}(n \log n)$ reduzieren können. [8]

4 Zusammenfassung

In dieser Arbeit wurde das Thema diskrete Sinustransformation behandelt. Dazu wurden erst die Grundlagen des Algorithmus erklärt, bevor die Anwendung der DST anhand einer praktischen Berechnung dargestellt wurde. Im Rahmen dessen wurde ein Python-Programm entwickelt, welches einen gegebenen diskreten Datensatz mittels der DST transformiert, filtert und schließlich rücktransformiert. Alle dieser Schritte wurden dabei grafisch dargestellt. Im beigefügten digitalen Anhang finden sich sowohl ein Programm zur Erzeugung der grafischen Darstellungen als auch eines zur numerischen Datenberechnung.
Die vorliegende Arbeit hat sich auf die grundlegenden Aspekte der Algorithmusanwendung beschränkt. Es eröffnen sich jedoch zahlreiche Möglichkeiten für zukünftige Weiterentwicklungen. Ein Beispiel hierfür ist die Implementierung eines verbesserten Algorithmus, wie im Abschnitt zur Laufzeitkomplexität erwähnt wurde.

Abbildungsverzeichnis

Literaturverzeichnis

[1] Ahmed, Nasir ; Natarajan, T. ; Rao, K. R.: *Discrete Cosine Transform.* IEEE Transactions on Computers, January 1974

[2] Dhamija, Swati ; Jain, Priyanka: *Comparative Analysis for Discrete Sine Transform as a suitable method for noise estimation.* International Journal of Computer Science, September 2011

[3] *Diskrete Sinustransformation [Online].* – URL: `https://dewiki.de/Lexikon/Diskrete_Sinustransformation`, Stand 16. August 2023

[4] Britanak, Vladimir ; Yip, Patrick C. ; Rao, K. R.: *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations.* Academic Press, Dezember 2014. – ISBN 978–1493301034

[5] Perlt, Holger: *Vorlesungsskript Numerik*

[6] *Discrete sine transform [Online].* – URL: `https://en.wikipedia.org/wiki/Discrete_sine_transform`, Stand 16. August 2023

[7] Fiedler, Martin: *Videokompressionsverfahren - von MPEG-1 bis H.264 und H.265 [Online].* – URL: `https://keyj.emphy.de/files/projects/videocomp.pdf`, Stand 16. August 2023

[8] Püschel, Markus ; Moura, José M. F.: *Algebraic Signal Processing Theory: Cooley–Tukey Type Algorithms for DCTs and DSTs.* IEEE Transactions on Signal Processing, April 2008

Selbstständigkeitserklärung

Ich versichere, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder veröffentlicht, noch einer anderen Prüfungsbehörde vorgelegt.

Leipzig, den 24.09.2023

Nikolaidis, Simon

Ort, Datum