Termin: Montag, 9. Mai 2022

Abschlussprüfung Sommer 2022

6511

4 Entwicklung eines Softwaresystems

Schriftliche Aufgabenstellungen

Mathematisch-technischer Softwareentwickler
Mathematisch-technische Softwareentwicklerin

Aufgabenbogen

3 Phasen, davon

- 7 Stunden schriftliche Aufgabe
- 4 Tage Realisierung des Konzepts
- 30 Minuten Fachgespräch

100 Punkte

Hinweis:
Bei der Bearbeitung der Aufgaben ist von einem gewöhnlichen Geschäftsbetrieb auszugehen, der **nicht** durch die COVID-19-Pandemie beeinflusst bzw. durch entsprechende behördliche Verfügungen eingeschränkt ist.

Vorbemerkung

Dieser Aufgabensatz besteht aus einem Aufgabenbogen und 20 einzelnen Bearbeitungsbogen.

Füllen Sie bei allen Bearbeitungsbogen zuerst die Kopfleiste aus. Die Bearbeitungsbogen sind während der Bearbeitung in dem vorgesehenen Kästchen durchzunummerieren. Verwenden Sie die einzelnen Bogen nicht als Schreibunterlage und kontrollieren Sie, ob Ihre Eintragungen auf der Durchschrift deutlich erscheinen (auch in der Kopfleiste).

Die vorliegende bundeseinheitliche Prüfungsaufgabe wird in drei Phasen bearbeitet.

Phase I:
- Schriftliche Klausur unter Aufsicht des Prüfungsausschusses der IHK
- Sie soll in der Regel montags stattfinden, Dauer 7 Stunden
- Es sind keine Hilfsmittel zugelassen.
- Die Ergebnisse sind handschriftlich auf Papier zu erstellen.
- Das Original wird dem Prüfungsausschuss übergeben, eine Kopie behält der Prüfling zur weiteren Bearbeitung.

Phase II:
- Die Bearbeitung erfolgt am betrieblichen Arbeitsplatz.
- Sie findet von dienstags bis freitags statt.
- Verwendete Hilfsmittel und Quellen sind anzugeben.
- Die Ergebnisse sind auf Papier und elektronisch lesbar nach Vorgabe des Prüfungsausschusses abzugeben.
- Hinzuzufügen ist auch eine Eigenhändigkeitserklärung.

Phase III:
- Das Fachgespräch findet zeitnah als Einzelprüfung mit dem Prüfungsausschuss statt.
- Der Prüfling soll das Prüfungsprodukt, die Aufgabenanalyse und den Lösungsentwurf in maximal 10 Minuten vorstellen und begründen. Hilfsmittel wie Flip Chart, Folien o. Ä. können verwendet werden.
- Im anschließenden etwa 20-minütigen Gespräch sind die Ergebnisse zu verteidigen.

Gemeinsame Prüfungsaufgaben der Industrie- und Handelskammern. Dieser Aufgabensatz wurde von einem überregionalen Ausschuss, der entsprechend § 40 Berufsbildungsgesetz zusammengesetzt ist, beschlossen.
Die Vervielfältigung, Verbreitung und öffentliche Wiedergabe der Prüfungsaufgaben und Lösungen ist nicht gestattet. Zuwiderhandlungen werden zivil- und strafrechtlich (§§ 97 ff., 106 ff. UrhG) verfolgt. – © ZPA Nord-West 2022 – Alle Rechte vorbehalten!

Physikalischer Hintergrund

Die Autokorrelationsfunktion (AKF, engl.: ACF) einer Funktion $f(t)$ beschreibt die Ähnlichkeit zu sich selbst zu einem späteren Zeitpunkt $f(t+s)$. Das ist vor allem bei verrauschten Signalen wichtig, um die zugrundeliegende Funktion besser charakterisieren zu können, denn durch eine Untersuchung der Selbstähnlichkeit des Signals wird das zufällige Rauschen zu großen Teilen ignoriert.

Eine typische Anwendung der Autokorrelationsfunktion in der heutigen Technik ist der optische Autokorrelator. Er misst die AKF eines gepulsten Lasers mittels Spiegeln, Kristallen und optischen Detektoren. Solche Laser kommen in wissenschaftlichen Experimenten, der Materialbearbeitung und der Medizin zum Einsatz. Abbildung 1 zeigt den prinzipiellen Aufbau eines optischen Autokorrelators.

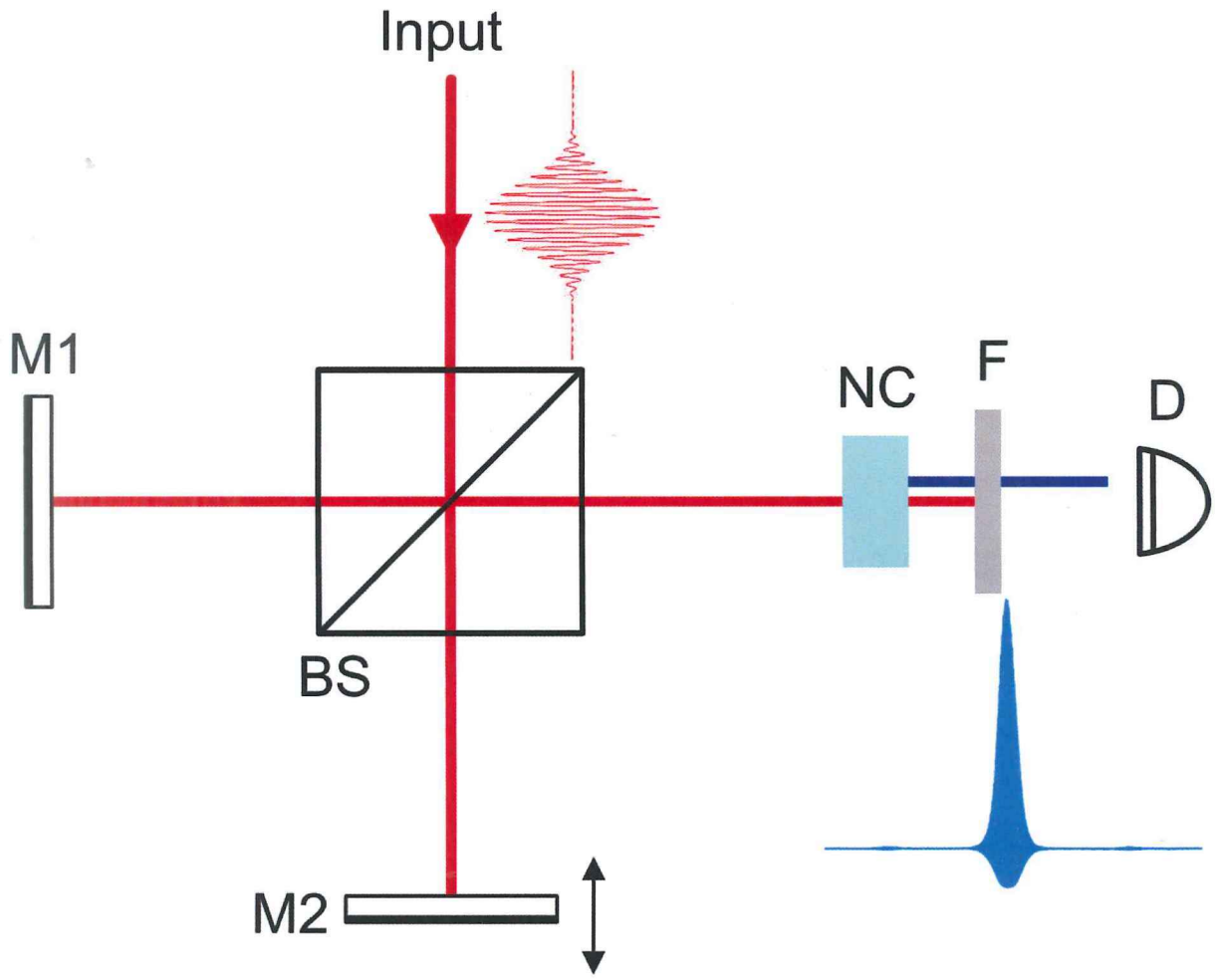

Abbildung 1: Aufbau eines Autokorrelators, Link: https://de.wikipedia.org/wiki/Autokorrelator

M1 und *M2* sind Spiegel, an denen das Licht vollständig reflektiert, wobei *M2* mithilfe eines Motors vor und zurück bewegt werden kann. *BS* ist ein halbdurchlässiger Spiegel, eine Hälfte des Lichts wird zu *M1* reflektiert und die andere Hälfte an M2 durchgelassen. Im Kristall *NC* findet durch physikalische Effekte eine Frequenzverdopplung (blauer Strahl) des wieder zusammengeführten Signals statt, welche durch den Filter *F* an den Detektor *D* weitergegeben wird. Durch Verschiebung des Spiegels *M2* wird die Laufzeit des Signals vom Eingang durch den Strahlteiler *BS* und zurück verändert. Dann kann im Detektor *D* die AKF des Eingangssignals gemessen werden, welche Aufschluss über die Breite des Laserpulses oder die Ursprungsfrequenz gibt.

Problembeschreibung

Ihre Firma, die MATSE AG, möchte einen optischen Autokorrelator bauen und Sie wurden damit beauftragt, einfache mathematische Methoden zur Signalauswertung bereitzustellen. Da bisher aber noch nicht einmal ein Prototyp existiert, müssen Sie dafür simulierte Daten benutzen, wie sie Ihnen später auch vom Gerät geliefert werden. Dazu ist es notwendig, dass Ihr Datenmodell eine einfache Austauschbarkeit der Signaldatenquelle vorsieht! Die Daten werden dabei von der Signalquelle (Detektor) in einem regelmäßigen Intervall geliefert. Wie lange aber die Verarbeitung und die Ausgabe der Daten dauert, ist von verschiedenen Faktoren abhängig und nicht vorhersehbar. Darum muss die Software in unabhängigen Threads oder Prozessen implementiert werden.

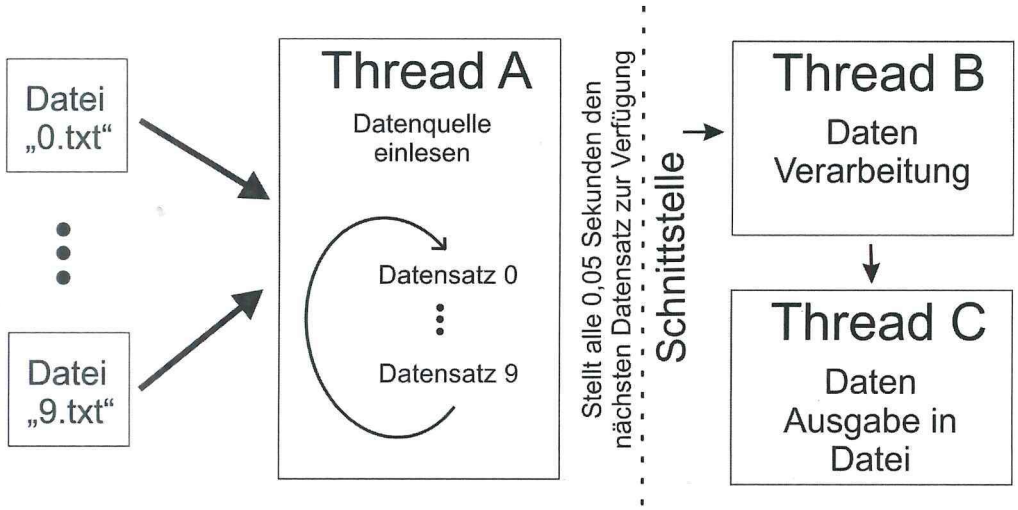

Abbildung 2: Konzept der unabhängigen Threads und ihrer Aufgaben

Simulierte Daten

Der Detektor im späteren Autokorrelator wird permanent und nebenläufig Daten liefern, ohne anzuhalten. Um dieses Verhalten zu simulieren, benötigen Sie einen Thread oder Prozess, der durchgehend Dateien ausliest und die dort enthaltenen Daten zur weiteren Verarbeitung zur Verfügung stellt. Die Daten sollen mit 20 Hz zur Verfügung gestellt werden, d. h. je eine Messung steht für 0,05 Sekunden zur Verfügung, bevor eine neue Messung die vorherige überschreibt.

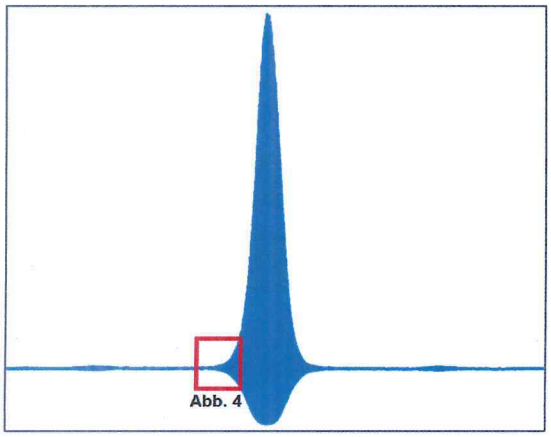

Abbildung 3: Beispiel einer Messung

Ihnen werden per Download zehn Dateien mit jeweils *N* Messdaten zur Verfügung gestellt. Diese Dateien sind aufsteigend von „0.txt“ bis „9.txt“ benannt und beinhalten jeweils eine komplette Messung.

Zuerst werden die Daten der ersten Messung aus der Datei „0.txt“ eingelesen, dann die Daten der zweiten Messung aus „1.txt“ usw. bis „9.txt“.

Nach Einlesen der Messung aus „9.txt“ beginnt das Einlesen wieder von vorne bei „0.txt“.

Das Format der Dateien sieht wie folgt aus:

```
# int   pos
20114 122185
20114 122179
20008 122178
...
...
```

Das „#“ steht zu Beginn einer Kommentarzeile, die ignoriert werden kann.

Die erste Spalte beinhaltet die Intensität des Messsignals aus dem Detektor D, also den y-Wert. Davon mit einem Tabulator getrennt steht in der zweiten Spalte die Position des Spiegels *M2*, also der $\tilde{x}$-Wert. Beides sind nicht-negative, ganze Zahlen.

Mathematische Methoden

In einem weiteren Thread oder Prozess sollen mathematische Methoden zur Auswertung der Daten einer Messung laufen. Diese Methoden beinhalten in dieser Reihenfolge:

1. Umrechnung und Normierung der Daten
Die x̃-Werte werden in Pikosekunden [ps] umgerechnet, die Formel lautet:

$$\hat{x}_k = \frac{\tilde{x}_k}{2^{18}-1} \cdot 266.3 - \ 132.3 \quad , \forall\, k \in [0, N-1]$$

Die y-Werte werden normiert, sprich alle y-Werte werden durch das Maximum aller y-Werte dividiert.

2. Glättung der Daten
Als nächstes müssen die Daten geglättet werden, denn durch Messungenauigkeiten sind diese nicht stetig, siehe Abbildung 4, da die x-Werte nicht monoton steigend sind. Dazu soll der *gleitende Mittelwert* berechnet werden. Die Formel hierfür lautet:

$$x_k = \frac{1}{n}\sum_{i=0}^{n-1} \hat{x}_{k-\tau+i} \qquad \text{mit} \quad \tau = \frac{n-1}{2}, \forall\, k \in [0, N-1]$$

$$n = \begin{cases} [0.002 \cdot N] - 1 & , \text{für } [0.002 \cdot N] \text{ gerade} \\ [0.002 \cdot N] & , \text{für } [0.002 \cdot N] \text{ ungerade} \end{cases}$$

n bezeichnet dabei die Größe des Mittelungsfensters, ist ungerade und entspricht 0,2 % der Anzahl der Messdaten N.

Treffen Sie eine geeignete Wahl für die Fälle $k < \tau$ und $k > N - 1 - \tau$, also den linken und rechten Rand der Messungen.

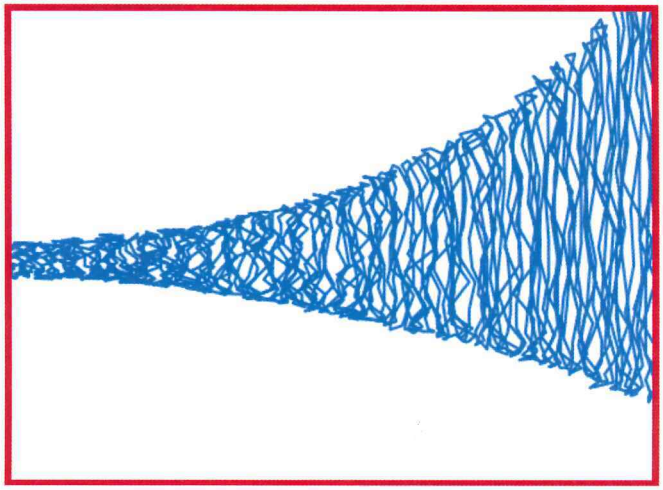

Abbildung 4: Ausschnitt aus den Sensordaten

3. Obere Einhüllende
Die obere Einhüllende ist die Funktion, die die AKF oben komplett „einschnürt", siehe Abbildung 5.

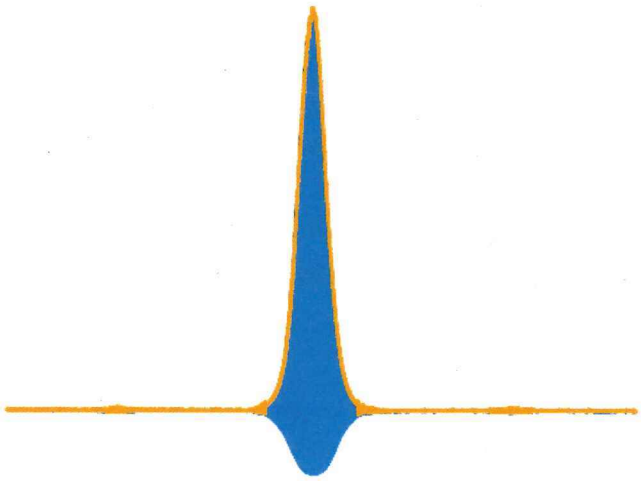

Abbildung 5: Obere Einhüllende

Die obere Einhüllende wird hier durch die Verbindung der lokalen Maxima approximiert. Eine solche Funktion zu bestimmen ist numerisch sehr aufwendig, weshalb eine einfachere Methode genutzt wird. Es reicht für unsere Zwecke aus „von links" jedem Positionswert den zuletzt höchsten Intensitätswert zuzuordnen. Wurde der höchste Intensitätswert der Messung erreicht, wird das gleiche Verfahren „von rechts" wiederholt.

4. Pulsbreite (FWHM – full width at half maximum)
Die Pulsbreite *b* beschreibt den Abstand der beiden Punkte *L* und *R* (siehe Abbildung 6). *L* bzw. *R* sind der erste Punkt links bzw. rechts vom Maximum, an dem die obere Einhüllende die halbe Höhe zwischen der Grundlinie und dem Maximum erreicht hat. Die Grundlinie ist hierbei nicht das Minimum der gesamten Messung, sondern die mittlere Höhe der äußersten linken ein Prozent der Intensitätswerte.

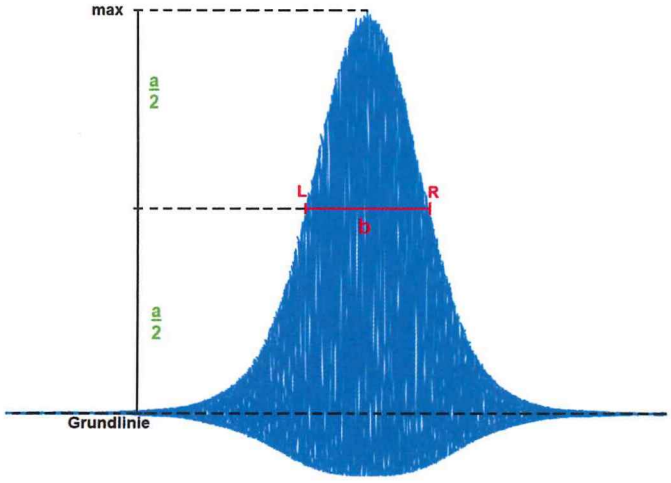

Abbildung 6: Pulsbreite

Ausgabe der Ergebnisse

Die fertigen Daten werden nach der vollständigen Berechnung in folgendem Format ausgeschrieben:

```
# FWHM = 3.7428899400660225e-06, 31602, 33656
# pos  int      env
0.0018260094247135369 0.16528505897771953    0.16528505897771953
0.0018260093170075734 0.1652113368283093     0.16528505897771953
0.001826009640125464  0.16444954128440367    0.16528505897771953
...
...
0.0019036737178079002 0.16514580602883355    0.16514580602883355
```

Zeilen, die mit „#" beginnen, bezeichnen erneut Kommentarzeilen. Die erste Zeile hat dabei eine besondere Bedeutung. Sie ist eine Kommentarzeile mit dem reellen Wert der Pulsbreite und den ganzzahligen Werten der Indizes der Punkte *L* und *R* im folgenden Format (siehe oben):

```
# FWHM = <fwhm : float>, <indexL : int>, <indexR : int>
```

Die nun folgenden Spalten sind mit Tabulatoren getrennt. Die erste Spalte beinhaltet das transformierte Positionssignal x, die zweite Spalte die normierte Intensität y und die dritte Spalte den Wert der oberen Einhüllenden.

Zu jeder Eingabedatei soll es exakt eine Ausgabedatei geben, diese trägt jeweils das Präfix „out". Die Ergebnisse der verarbeiteten Daten aus „0.txt" werden also in „out0.txt" gespeichert usw.

Simulationsende

Das Programm endet, wenn es zu jeder Eingabedatei eine Ausgabedatei gibt. Dies kann auch schon nach dem ersten Durchgang erreicht werden, je nachdem wie schnell die zur Verfügung gestellten Daten weiterverarbeitet werden. Gegebenenfalls müssen weitere Durchgänge erfolgen. Sollten bereits verarbeitete Dateien erneut eingelesen werden, müssen sie jedoch nicht erneut verarbeitet und ausgegeben werden!

Aufgabe

Implementieren Sie eine Simulation des Autokorrelators, in der:

- ein Detektor die Daten aus den gegebenen Dateien ausliest.
- die Daten nacheinander die beschriebenen mathematischen Methoden durchlaufen.
- die Ergebnisse im gegebenem Format ausgeschrieben werden.

Das Einlesen, Verarbeiten und Ausgeben der Daten muss nebenläufig realisiert werden. Dabei dürfen nur die Standardbibliotheken für Threads und Prozesse der von Ihnen gewählten Programmiersprache verwendet werden. Explizit von der Verwendung ausgeschlossen sind externe Bibliotheken wie z. B., aber nicht ausschließlich, OpenMP, MPI, OpenACC, o. Ä., welche eine Nebenläufigkeit bereits implementieren.

Hilfsmittel

Ihnen wird das Python-Programm „FileViewer.py" zur Visualisierung der Ausgabedateien zur Verfügung gestellt. Es benötigt mindestens Python 3.7.1 und matplotlib 3.0.2. Python-Installer für Ihr Betriebssystem können von https://www.python.org/downloads/ heruntergeladen werden. Wählen Sie während der Installation „Python zur PFAD-Umgebungsvariable hinzufügen" aus!

Nach der Installation führen Sie in der Konsole folgenden Befehl aus, um matplotlib zu installieren:

```
pip install matplotlib
```

In den ersten paar Zeilen von „FileViewer.py" finden Sie einstellbare Parameter wie

```
dir = <string>
filePrefix = <string>
fileSuffix = <string>
```

mit denen Sie Speicherort und die Namensgebung Ihrer Ausgabedateien anpassen können.

Das Programm wird von der Kommandozeile gestartet mit:

```
python FileViewer.py
```

Achten Sie bei der Ausführung auf relative und absolute Pfade in „dir = ..." .

Im Rahmen der schriftlichen Aufgabe sind am ersten Tag abzugeben:

- Aufgabenanalyse und Beschreibung der mathematischen Methoden
- Konzepterstellung zur Nebenläufigkeit von Einlesen, Verarbeiten und Ausgeben
- Entwurf der vier Algorithmen zur Berechnung der Ergebnisse
- Klassen, Methoden und Datenstrukturen in Form von UML-Diagrammen
- UML-Sequenzdiagramme für die wesentlichen Abläufe
- Detaillierte Beschreibung der wesentlichen Methoden in Form von Nassi-Shneiderman-Diagrammen oder Programmablaufplänen

Im Rahmen des Prüfprodukts sind in gedruckter und elektronischer Form abzugeben:

- Verbale Beschreibung der realisierten Nebenläufigkeit
- Verbale mathematische Beschreibung der vier realisierten Algorithmen zur Berechnung der Ergebnisse
- Entwicklerdokumentation und Benutzeranleitung
- Programmsystem (bestehend aus Klassen, Schnittstellen, Methoden) als Quellcode
- Zusammenfassung und Ausblick (z. B. Erweiterungsmöglichkeiten)

Im Rahmen des Prüfprodukts sind in elektronischer Form abzugeben:

- Programmsystem in ausführbarer Form für die nebenläufige Simulation eines Autokorrelators
- Ausgabedateien Ihrer Berechnung zu den einzelnen Messdatensätzen

Dem Prüfprodukt ist eine eigenhändig unterschriebene Eigenhändigkeitserklärung (Juristische Erklärung) folgenden Inhalts beizufügen:

„Ich erkläre verbindlich, dass das vorliegende Prüfprodukt von mir selbstständig erstellt wurde. Die als Arbeitshilfe genutzten Unterlagen sind in der Arbeit vollständig aufgeführt. Ich versichere, dass der vorgelegte Ausdruck mit dem Inhalt der von mir erstellten digitalen Version identisch ist. Weder ganz noch in Teilen wurde die Arbeit bereits als Prüfungsleistung vorgelegt. Mir ist bewusst, dass jedes Zuwiderhandeln als Täuschungsversuch zu gelten hat, der die Anerkennung des Prüfprodukts als Prüfungsleistung ausschließt."

Im Rahmen des auftragsbezogenen Fachgesprächs sind die Aufgabenanalyse und der Lösungsentwurf zu begründen und das Prüfungsprodukt zu erläutern.