**Zürcher Hochschule für Angewandte Wissenschaften**
**ZHAW**

School of Management and Law
Abteilung Banking, Finance, Insurance

Bachelor-Thesis

# Bootstrapping Value at Risk (VaR):

## Circular Block Bootstrap vs. Standard Bootstrap

Vorgelegt von:
Lionel Rémy Nyffeler
Matrikelnummer: 17-660-143

Eingereicht bei:
Armin Bänziger

Winterthur, 26. Mai 2021

# Management Summary

Bootstrapping kann als eine Variation des historischen Simulationsansatzes eingesetzt werden, welche darauf abzielt, das Konfidenzintervall des Value at Risks (VaR) zu berechnen. Die ursprüngliche Bootstrap-Methode, eingeführt von Efron im Jahre 1977, zieht jeweils zufällig einzelne Tagesrenditen aus einem Sample. Das zufällige Ziehen einzelner Tage führt jedoch insbesondere bei Zeitreihen dazu, dass die zeitliche Struktur der Renditen ignoriert wird. Aus diesem Grund wird in der Literatur der Block Bootstrap vorgeschlagen. Diese Arbeit befasst sich mit dem Circular Block Bootstrap (CBB) nach Politis und Romano. Dabei werden zufällig ganze Blöcke, aus dem Sample gezogen, um die Eigenschaften der zeitlichen Struktur innerhalb der Blöcke aufrechtzuerhalten. Circular bedeutet, dass das Sample-Ende mit dem Sample-Anfang verbunden wird, um das Ziehen von unvollständigen Blöcken zu verhindern.

Diese Arbeit untersucht, ob das Ziehen von Blöcken mit der CBB-Methode die Abhängigkeit innerhalb der Zeitreihe tatsächlich besser berücksichtigt und den VaR exakter schätzen kann. Somit wird in dieser Arbeit ein Vergleich zwischen dem CBB und der Bootstrap Historical Simulation (BHS) gemacht.

Um diese Fragestellung zu beantworten, wird der 10-Tages-VaR auf den beiden Konfidenzniveaus 95 % und 85 % berechnet. Dieser wird anhand eines Portfolios, bestehend aus fünf Aktien (Logitech, Novartis, Swiss Re, LafargeHolcim und Nestle) über die Zeitspanne vom 04.01.2010 – 16.03.2021 mit dem Rolling-Window-Prinzip angewendet. Dabei werden die beiden Bootstrap-Methoden anhand der Kombinationen, bestehend aus den verschiedenen Sample-Grössen von 250, 500, 750 und 1000 Tagen mit je 1000, 2000 und 3000 Simulationen getestet. Um die beiden Modelle zu vergleichen, wird mit dem Anteilswert-Test eruiert, ob die Modelle korrekt spezifiziert sind und das Risiko exakt einschätzen. Weiter muss sichergestellt werden, dass die Exceptions keine zeitliche Struktur mehr aufweisen. Dafür wird die Zufälligkeit der Exceptions mit dem Runs-Test und die Autokorrelation mit dem Ljung-Box-Test quantifiziert. Die Durchführung der Simulationen und des Backtestings wird in Python implementiert.

Die Resultate zeigen, dass die CBB-Methode das exaktere Modell ist und somit das Ziehen von Blöcken anstelle einzelner Tage zu einer Verbesserung der Schätzung des VaRs herbeiführt. Zusätzlich wird bewiesen, dass der CBB die zeitliche Struktur besser ausgleicht, indem die Exceptions zufällig auftreten und keine Autokorrelation aufweisen. Insbesondere bei den Sample-Grössen von 750 und 1000 Tagen kann die CBB-Methode das Risiko am besten einschätzen. Eine weitere Erkenntnis ist, dass die Anzahl Simulationen (1000, 2000 und 3000) bei beiden Methoden keinen signifikanten Einfluss auf die Ergebnisse hat. Davon kann abgeleitet werden, dass 2000 und 3000 Simulationen keinen grossen Mehrwert für die Resultate generieren.

# Inhaltsverzeichnis

# Tabellenverzeichnis



# Abbildungsverzeichnis

# 1 Einleitung

Value at Risk (VaR) wird als maximal erwarteter Verlust eines Portfolios bezeichnet, welcher über einem bestimmten Zeithorizont und einer definierten Wahrscheinlichkeit nicht überschritten wird (Jorion, 2010, S. 289). Der VaR ist eine der wichtigsten Risikokennzahlen, um das Marktrisiko zu quantifizieren. Diese Kennzahl versucht das totale Risiko eines Portfolios mit einer einzigen Zahl auszudrücken (Hull, 2012, S. 183).

Mit der steigenden Volatilität der Märkte in den 1980er Jahren und der steigenden Fremdfinanzierung der Finanzinstitute wuchs der Bedarf bei Banken, die Marktrisiken über verschiedene Anlagekategorien hinweg messen zu können (Holton, 2002, S. 6). Die technischen Entwicklungen, wie das schnellere Durchführen komplexer Berechnungen oder die Verbreitung von Finanzdaten, ermöglichten es, neue Modelle in der Praxis umzusetzen (Holton, 2002, S. 6). Im Jahr 1994 wurde zum ersten Mal das Konzept des VaRs durch die J.P. Morgan veröffentlicht, welches sich rasch in der Finanzindustrie etablierte (Hull, 2012, S. 183). Dieses Konzept wurde 1996 auch im Basler Ausschuss aufgenommen, mit dem Ziel, dass die Banken neben dem Kreditrisiko auch die Marktrisiken mit Eigenkapital unterlegen (Romero, Muela und Martin, 2013, S. 2). Nebst den regulatorischen Vorgaben wird der VaR vor allem intern bei Banken als Risikomanagementinstrument benutzt, um die Risiken des Handels zu kontrollieren (Whitehead, 2011, S. 2). Ein weiterer Vorteil des VaRs ist, dass diese Kennzahl sowohl auf Gesamtbankebene als auch auf den einzelnen Hierarchieebenen gerechnet werden kann (Jorion, 2010, S. 456).

Ursprünglich wurde von der J.P. Morgan 1994 der Varianz-Kovarianz-Ansatz angewendet, welcher ein parametrischer Ansatz ist. Dabei werden Annahmen bezüglich der Parameter getroffen, indem die Verteilung mit dem Mittelwert und der Varianz berechnet wird (Romero et al., 2013, S. 2). Dieser Ansatz geht von einer Normalverteilung der Renditen aus, wobei bewiesen wurde, dass Renditen nicht normalverteilt sind (Crouhy, Galai und Mark., 2014, S. 249). Da solche Annahmen nicht der Realität entsprechen, führt dies zu ungenauen und möglicherweise zu optimistischen Schätzungen. Höhere Tails in der Verteilung implizieren, dass aussergewöhnlich hohe Verluste häufiger vorkommen als in der Normalverteilung angenommen wurde (Crouhy et al., 2014, S. 249).

Im Laufe der Zeit wurden in der Finanz-Branche immer wieder neue Erkenntnisse und somit neue Ansätze und Methoden vorgeschlagen, um die Berechnung des VaR zu verbessern (Romero et al., 2013, S. 3). Weitere Alternativen, nebst dem Varianz-Kovarianz-Ansatz, sind die historische Simulation und die Monte-Carlo-Simulation. Der historische Simulationsansatz ist ein nicht-parametrischer Ansatz und eignet sich insbesondere, wenn keine Informationen zur Verteilung der Renditen vorhanden sind (Hull, 2012, S. 303). Im Gegensatz zum Varianz-Kovarianz-Ansatz

müssen die Parameter nicht geschätzt werden und können vom historischen Datenset abgelesen werden. Dabei sind die historischen Volatilitäten und Korrelationen bereits im Datenset reflektiert und bilden somit die Tails besser ab (Crouhy et al., 2014, S. 254). Diese nicht-parametrischen Ansätze basieren alle auf der zugrundeliegenden Annahme, dass die Zukunft genügend ähnlich wie die jüngste Vergangenheit sein wird. Somit wird davon ausgegangen, dass die historischen Daten genutzt werden können, um das Risiko der nahen Zukunft zu prognostizieren (Romero et al., 2013, S. 5). Eine Umfrage von McKinsey (2012, S. 4) zeigt, dass 85 % der Banken weltweit die historische Simulation als Ansatz für die Berechnung des VaR verwenden.

Die historische Simulation kann auch mittels Bootstrapping durchgeführt werden. Der Standard Bootstrap, eingeführt von Efron (1977), ist eine Resampling-Methode, welche zur Schätzung unbekannter Grössen in Verbindung mit statistischen Modellen gebraucht wird. Basierend auf einer Stichprobe werden wiederholt Statistiken berechnet, um die Parameter der Verteilung von Zufallsvariablen zu schätzen (Boos, 2003, S. 1). In dieser Arbeit wird die Bootstrap Historical Simulation (BHS), welche dem Standard Bootstrap entspricht, mit der Circular Block Bootstrap-Methode (CBB) von Politis und Romano (1992) verglichen. Bei der BHS werden nach Zufall einzelne Tage gezogen, wohingegen bei der Block Bootstrap-Methode ganze Blöcke von zusammenhängenden Tagen gezogen werden.

Das zufällige Ziehen von einzelnen Tagen bei der BHS ignoriert die Abhängigkeiten innerhalb der Zeitreihe (Rjiba, Tsagris und Mhalla, 2015, S. 4). Damit die Autokorrelation besser berücksichtigt wird, schlägt Hall (1985, S. 231-246) den Block Bootstrap vor, bei welchem ganze Blöcke gezogen werden.

Ein Problem beim Block Bootstrap ist, dass das Ziehen von Blöcken am Ende des Datensets zu unvollständigen Blöcken führt. Dafür schlagen Politis und Romano (1992) den Circular Block Bootstrap (CBB) vor, welche eine Variation des Block Bootstraps ist. Circular bedeutet, dass das Ende mit dem Anfang der Zeitreihe verbunden wird, um zusätzliche Blöcke zu erhalten und das Problem unvollständiger Blöcke zu beseitigen (Politis und Romano, 1992, S. 4).

## 1.1 Fragestellung

Diese Thesis beantwortet folgende Fragestellung: *Führt der Circular Block Bootstrap gegenüber dem Standard Bootstrap zu einer Verbesserung der Schätzung des VaR?*

In dieser Arbeit wird der Standard Bootstrap nach Efron (1979), nachfolgend Bootstrap Historical Simulation genannt (BHS), mit der CBB-Methode nach Politis und Romano (1992) verglichen. Mit diesen beiden Methoden wird der 10-Tages-VaR mit den Konfidenzniveaus 95 % und 85 % berechnet. Der VaR wird anhand eines Portfolios, bestehend aus fünf Wertschriften (Novartis, Logitech, Nestle, Lafarge und Swiss Re), in Python implementiert und ausgewertet. Um einen aussagekräftigen Vergleich zu machen, werden die beiden Bootstrap-Methoden anhand der Kombinationen, bestehend aus den verschiedenen Sample-Grössen von 250, 500, 750 und 1000 Tagen mit 1000, 2000 und 3000 Simulationen berechnet. Diese Arbeit wird die Resultate zusätzlich nach Anzahl Simulationen und Sample-Grössen analysieren.

Dowd (2007, S. 197) macht darauf aufmerksam, dass bei zu hohen Konfidenzniveaus weniger Exceptions und somit zu wenige Beobachtungen der Tail-Verteilung vorhanden sind. Dadurch besteht die Möglichkeit, dass die Resultate ungenau sind (Dowd, 2007, S. 197). Weiter erwähnt Dowd (2007, S. 197), dass die Resultate je nach Konfidenzniveau unterschiedlich sein können. Aus diesem Grund wird nebst dem Konfidenzniveau 95 % zusätzlich auf dem Konfidenzniveau von 85 % getestet.

Der Fokus dieser Arbeit liegt darauf, herauszufinden, ob das Ziehen von Blöcken, bestehend aus zehn Tagen, den VaR besser modelliert als wenn zehn zufällige einzelne Tage gezogen werden. Die CBB-Methode ist eine Variation des Block Bootstraps (Rjiba et al. 2015, S. 5). Diese Arbeit analysiert nicht den isolierten Einfluss des Circular-Effekts, sondern dient lediglich der Meidung des Ziehens unvollständiger Blöcke (Politis und Romano, 1992, S. 4). Weiter wird sich diese Arbeit nicht mit der Definition einer optimalen Block-Länge auseinandersetzen und beschränkt sich damit auf eine Standardblockgrösse von zehn Tagen.

Um die beiden Methoden miteinander zu vergleichen, werden mehrere Backtesting-Methoden angewendet. Zuerst wird die Exceptions Rate analysiert und mittels Anteilswert-Test evaluiert, ob die VaR-Modelle korrekt spezifiziert sind (Hull, 2012, S. 198). Die Exceptions Rate sagt jedoch nur aus, wie viele Exceptions generiert werden, und nicht, ob sie zufällig oder gruppiert vorkommen (Hull, 2012, S. 198). In dieser Arbeit ist zentral, dass überprüft wird, ob das Ziehen von Blöcken die zeitliche Struktur besser berücksichtigt und somit keine Zufälligkeit und Autokorrelation besitzt. Damit der Anteilswert-Test angewendet werden kann, muss sichergestellt werden, dass die Exceptions keine zeitliche Struktur mehr aufweisen. Dafür wird die Zufälligkeit

der Exceptions mit dem Runs-Test nach Wald und Wolfowitz (1943) und die Autokorrelation mit dem Ljung-Box-Test (Ljung und Box, 1978) quantifiziert.

## 1.2 Aufbau der Arbeit

Im ersten Kapitel wurde eine Einführung in das Thema gegeben und die Fragestellung erläutert, welche diese Arbeit untersuchen wird. Im zweiten Kapitel werden die theoretischen Grundlagen des VaRs und dessen verschiedene Berechnungsansätze aufgezeigt. Darauf folgt eine Einführung in die Bootstrap-Methoden, insbesondere wird auf die BHS- und die CBB-Methode eingegangen. Weiter behandelt dieses Kapitel die Theorie der verschiedenen Backtesting-Methoden. Im dritten Kapitel wird das methodische Vorgehen erläutert, wobei das gewählte Portfolio sowie das Datenset aufgezeigt werden. Im selben Kapitel erfolgt die Erklärung der Implementation und des Vorgehens bei der Umsetzung der Methoden. Im vierten Kapitel werden die Resultate des Backtestings präsentiert und die beiden Methoden miteinander verglichen. Das letzte Kapitel fasst die Resultate zusammen, geht auf die Beantwortung der Forschungsfrage ein und gibt einen Ausblick für weiterführende Arbeiten.

# 2 Theoretische Grundlagen

Im folgenden Kapitel werden der VaR und dessen wichtigsten Ansätze aufgezeigt, wobei insbesondere auf die historische Simulation eingegangen wird. Danach wird eine Einführung in die Bootstrap-Methoden gemacht und speziell die beiden ausgewählten Bootstrapping Methoden erläutert. Zum Schluss wird die theoretische Grundlage für das Backtesting gegeben, bei welcher die Exceptions Rate, der Anteilswert-Test, der Runs-Test und der Ljung-Box-Test erklärt werden.

## 2.1 Value at Risk

Der VaR bezeichnet den maximal erwarteten Verlust eines Portfolios, welcher einen bestimmten Zeithorizont und eine definierte Wahrscheinlichkeit (Konfidenzniveau α) nicht überschreitet (Jorion, 2010, S. 289). Als Beispiel eines 10-Tages-VaRs (95 %) von einem Portfolio mit dem aktuellen Marktwert von CHF 5 Millionen kann folgende Aussage formuliert werden: *«Wir sind zu 95 % sicher, dass wir über die nächsten zehn Tage nicht mehr als CHF 30'000 verlieren werden.»*

In diesem Beispiel beträgt der VaR CHF 30'000. Zur Verallgemeinerung wird der VaR in dieser Arbeit relativ und unabhängig eines Portfolio-Betrages betrachtet, indem alle fünf Aktien als gleichgewichtete Positionen geführt werden. Weitergehend werden die täglichen Log-Renditen durch $R_t = \log(\frac{p_t}{p_{t-1}})$ definiert, wobei $p_t$ den Preis des Portfolios zum Zeitpunkt $t$ darstellt (Rjiba et al., 2015, S. 3). Folglich wird der 1-Tages-VaR mit dem Konfidenzniveau $\alpha$ durch folgende Formel definiert:

$$VaR_t(\alpha) = F^{-1}(1-\alpha) \tag{1}$$

$F^{-1}$ entspricht der inversen kumulativen Verteilungsfunktion der Renditen $R_t$ (Hull, 2012, S. 191).

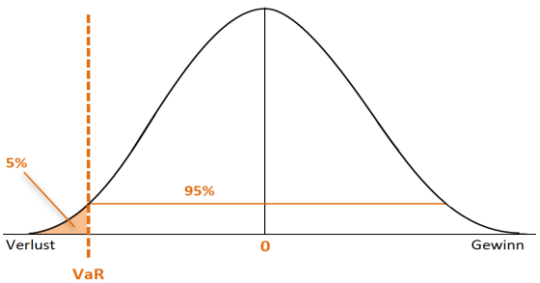



Abbildung 1: Grafisches Beispiel eines VaRs (95 %)

Die Abbildung 1 zeigt ein Beispiel einer Verteilung der täglichen Renditen. Dabei ist ersichtlich, dass der VaR das $(1-\alpha)$-Quantil der Verteilungsfunktion darstellt. Indes befinden sich 5 % der Verluste (orange Fläche) ausserhalb des Konfidenzniveaus $\alpha$ von 95 %. Am häufigsten wird ein

Konfidenzniveau zwischen 95 % und 99 % gewählt (Hendricks, 1996, S. 40). Gemäss Basel II (Basel Committee on Banking Supervision [BCBS], 1996, S. 3) müssen Banken den 10-Tages-VaR auf dem einseitigen Konfidenzniveau von 99 % berechnen, wobei das Datenset des historischen Beobachtungszeitraums mindestens einem Jahr entsprechen muss (Hull, 2012, S. 267). In dieser Arbeit wird der VaR auf den Konfidenzniveaus 95 % und 85 % getestet. Die Aussagekraft des VaRs stösst jedoch an seine Grenzen, da sie keine Aussage über den tatsächlichen Verlust macht, sondern nur aussagt, wie wahrscheinlich ein bestimmter Betrag in einer definierten Zeit überschritten wird (Crouhy et al., 2014, S. 239). Anders ausgedrückt ist der VaR ein Quantil, welches angibt, wie oft der tatsächliche Verlust grösser als der berechnete VaR ist (Jorion, 2010, S. 362).

## 2.2 Ansätze zur Berechnung des VaRs

Für die Berechnung des VaRs eines Portfolios wurden ursprünglich folgende Methoden entwickelt (Crouhy et al., 2014, S. 248):

- Varianz-Kovarianz-Ansatz (parametrisch)
- Historische Simulation (nicht-parametrisch)
- Monte-Carlo-Simulation (semi-parametrisch)

Der Varianz-Kovarianz-Ansatz ist ein bekannter parametrischer Ansatz und entspricht dem ursprünglich entwickelten Ansatz von der J.P. Morgan im Jahre 1996 (Romero et al., 2013, S. 2). Dieser Ansatz ist insofern parametrisch, dass Annahmen bezüglich der Parameter getroffen werden (Romero et al., 2013, S. 2). Ein Nachteil dieser Methode ist, dass von einer Normalverteilung der Renditen ausgegangen wird, wobei diese nicht der Realität entspricht (Crouhy et al., 2014, S. 248). Dies kann zu erheblichen Tails führen, wodurch die Marktrisiken stark unterschätzt werden. Ein weiterer Nachteil dieses Ansatzes ist, dass die Volatilität und die Korrelation der Renditen geschätzt werden müssen (Crouhy et al., 2014, S. 257).

Aufgrund dieser Nachteile wurden neue Ansätze entwickelt, wie beispielsweise der historische Simulations-Ansatz. Die historische Simulation ist ein nicht-parametrischer Ansatz, da keine Annahmen bezüglich der Parameter getroffen werden müssen (Romero et al., 2013, S. 4). Die Idee dieses Ansatzes liegt darin, dass die historischen Daten und damit die aktuelle empirische Verteilung der Renditen verwendet werden können, um den VaR zu prognostizieren (Romero et al., 2013, S. 5). Ein Vorteil dieser Methode ist, dass diese einfach zu implementieren ist und die Fat Tails in der Verteilung und andere extreme Events in der Berechnung berücksichtigt werden, solange diese auch im Datenset enthalten sind (Crouhy et al., 2014, S. 258). Folglich ist ein Nachteil, dass die Ergebnisse nur vergangene Beobachtungen enthalten können und keine Möglichkeit zur

Extrapolation haben. Wenn der beobachtete Zeitraum in der Vergangenheit besonders ruhig war, der VaR aktuell aber in einer volatilen Umgebung angewendet wird, kann dies zu einer erheblichen Unterschätzung des Risikos führen (Romero et al., 2013, S. 5).

Bei der Monte-Carlo-Simulation handelt es sich um einen semi-parametrischen Ansatz, bei welchem wiederholt zufällige Szenarien generiert werden. Jedes Szenario ist ein möglicher Portfoliowert zu einem bestimmten Zeitpunkt. Dies wird so häufig simuliert, bis diese zu einer definitiven Verteilung konvergiert (Crouhy et al., 2014, S. 254). Bei diesem semi-parametrischen Ansatz müssen zuerst alle relevanten Risikofaktoren spezifiziert werden, wie beispielsweise die Volatilität oder die Korrelation. Ein Vorteil ist, dass mit den Parametern Änderungen in der Marktstruktur simuliert werden können, wie zum Beispiel die Einführung des Euros im Jahr 1999 (Crouhy et al., 2014, S. 258).

Von diesen drei Ansätzen wird gemäss einer Umfrage von McKinsey (2012) die Monte-Carlo-Simulation als besten theoretischen Ansatz betrachtet. Bei Banken wird die Monte-Carlo-Simulation jedoch weniger häufig angewendet, da diese sehr berechnungs-intensiv ist und damit viel Rechenleistung benötigt (Crouhy et al., 2014, S. 257). In der Literatur gibt es noch viele weitere Varianten dieser Ansätze, wie beispielsweise die EVT (Extreme Value Theory) oder die Filtered Historical Simulation. Romero et al. (2013, S. 38) machen in ihrer Arbeit eine ausführliche Gegenüberstellung der Literatur. Darin ist ersichtlich, dass die EVT und die Filtered Historical Simulation den VaR am häufigsten als beste Methode definiert wurde. In dieser Gegenüberstellung sind jedoch die Bootstrap Methoden nicht berücksichtigt worden.

Für die Prognose des VaRs muss eine optimale Länge des Datensets definiert werden. Ist das Datenset kurz, passt sich der VaR schneller der aktuellen Marktsituation an, da immer nur die neusten Renditen berücksichtigt werden (Hendricks, 1996, S. 41). Somit könnte es vorkommen, dass in einem kurzen Datenset keine extreme Ereignisse enthalten sind, da diese bereits weiter in der Vergangenheit liegen. Bei einem langen Datenset hingegen wird zu viel berücksichtigt, wodurch der VaR nicht der aktuellen Marktsituation entspricht und immer konstant bleibt (Hendricks, 1996, S. 41). Eine Möglichkeit ist die unterschiedliche Gewichtung der Renditen im Datenset. Der EWMA (Exponentially Weighted Moving Average) gewichtet und berücksichtigt die aktuelleren Daten stärker als die älteren, da angenommen wird, dass die Renditen beispielsweise von letzter Woche wichtiger sind als diese von vor zwei Jahren (Hendricks, 1996, S. 42). Gemäss Basel II (BCBS, 1996, S. 3) müssen Banken ein Datenset von mindestens 250 Tagen nehmen (Hull, 2012, S. 267). Aufgrund der Unklarheit einer optimalen Sample-Grösse für die Berechnung des VaRs werden in dieser Arbeit die verschiedenen Sample-Grössen von 250, 500, 750 und 1000 Tage angewendet.

## 2.3 Bootstrap-Methoden

Bootstrapping wurde erstmals von Efron (1977) beschrieben und ist eine allgemeine Resampling-Methode, welche zur Schätzung unbekannter Grössen in Verbindung mit statistischen Modellen gebraucht wird. Basierend auf einer Stichprobe werden wiederholt Statistiken berechnet, um die Parameter der Verteilung von Zufallsvariablen zu schätzen (Boos, 2003, S. 1). Bootstrapping wird unter anderem auch in der Toxikologie und Chemometrie angewendet (Dixon, 2006, S. 1). Da das Bootstrapping ein nicht-parametrischer Ansatz ist, müssen auch hier keine Verteilungsannahmen getroffen werden. Dies eignet sich insbesondere in Zweifelsfällen bezüglich Annahmen der Normalverteilung des Modells. Der Vorteil von Bootstrapping ist, dass diese Methode oft eine genauere Approximation liefert als die asymptotische Theorie erster Ordnung (Härdle, Horowitz und Kreiss, 2003, S. 435). Die Resultate hängen jedoch davon ab, ob die Daten iid (independent and identically distributed) sind, oder ob es sich um eine Zeitreihe handelt. Wenn die Daten iid sind, kann der Bootstrap durch eine Zufallsstichprobe mit Zurücklegen oder durch die Stichprobe eines parametrischen Modells der Datenverteilung implementiert werden (Härdle et al., 2003, S. 435). Die Details dazu wurden von Beran und Ducharme (1991), Hall (1992), Tibshirani und Efron (1993), sowie von Davison und Hinkley (1997) diskutiert (Härdle et al., 2003, S. 435). Wenn es sich um eine Zeitreihe handelt, muss das Bootstrapping anders durchgeführt werden, damit die Abhängigkeitsstruktur angemessen berücksichtigt wird (Härdle et al., 2003, S. 435).

### 2.3.1 Bootstrap Historical Simulation (BHS)

Die Standard Bootstrap-Methode, kann als eine Variation des historischen Simulationsansatzes eingesetzt werden, welche darauf abzielt das Konfidenzintervall für den VaR zu berechnen (Hull, 2012, S. 313). Bei dieser Bootstrap-Methode werden aus einem Sample, bestehend aus historisch täglichen Renditen, zufällige Stichproben mit Zurücklegen gezogen (Dowd, 2007, S. 63). Bei einem 10-Tages-VaR wird eine Zufallsstichprobe im Umfang von zehn gezogen und eine Prognose der folgenden 10-Tages-Portfolio-Rendite gemacht. Dieser Prozess wird danach beispielsweise 1000 Mal wiederholt. Bei einem Konfidenzniveau von 95 % würde dies bedeuten, dass in 5 %, also in 50 von 1000 Fällen, der Verlust grösser ist als der vom VaR definierte Verlust (Hull, 2012, S. 313). Dutta und Bhattacharya (2008, S. 2) argumentieren, dass die BHS besser ist als die normale historische Simulation, da sie die wahren Verteilungseigenschaften beibehält und gleichzeitig mit der Knappheit von adäquaten Daten besser umgeht.

### 2.3.2 Circular Block Bootstrap (CBB)

Durch das Resampling einzelner Tage wird die Reihenfolge der Daten verändert, was dazu führt, dass wertvolle Informationen über die Grundgesamtheit verloren gehen (Jeong und Chung, 2001, S. 54). Da es sich bei Renditen um Zeitreihen handelt, welche voneinander abhängig sind, wird die Standard Bootstrap-Methode weniger genau (Jeong und Chung, 2001, S. 54). Um mit diesem Problem umzugehen, wurden von Hall (1985), Künsch (1989), sowie von Liu und Singh (1992) verschiedene Block Bootstrap-Varianten vorgeschlagen. Beim Block Bootstrap werden Blöcke anstatt einzelne Beobachtungen gezogen, um die Interdependenzstruktur zu erhalten (Jeong und Chung, 2001, S. 54). Bei der Block Bootstrap-Methode gibt es wiederum verschiedene Variationen:

- Nonoverlapping Block Bootstrap nach Carlstein (1986)
- Moving Block Bootstrap nach Künsch (1989) und Liu und Singh (1992)
- Circular Block Bootstrap nach Politis und Romano (1992)
- Stationary Bootstrap nach Politis und Romano (1994)

Diese Arbeit wird sich mit dem Circular Block Bootstrap (CBB) nach Politis und Romano (1992) auseinandersetzen. Der Circular-Effekt bedeutet, dass das Ende mit dem Anfang der Zeitreihe verbunden wird, um zusätzliche Blöcke zu erhalten und das Problem von unvollständigen Blöcken zu beseitigen (Politis und Romano 1992, S. 4).

Beim Moving Block Bootstrap nach Künsch (1989) und Liu und Singh (1992) geht es darum, dass die Blöcke überlappend sind, während der Nonoverlapping Block Bootstrap nach Carlstein (1986) das Sample in nicht überlappende Blöcke unterteilt wird. Beim Stationary Bootstrap nach Politis und Romano (1994) wird die Block-Länge nach Zufall generiert. Lahiri (1999, S. 395) behauptet, es sei besser, überlappende Blöcke und fixe Block-Längen zu benutzen und empfiehlt den Moving Block Bootstrap oder den Circular Block Bootstrap einzusetzen. MacKinnon (2006, S. 18) weist darauf hin, dass die Block-Länge nicht zu kurz sein darf, da sonst die Abhängigkeiten verloren gehen. Die Block-Länge soll jedoch auch nicht zu lange sein, da sonst der Bootstrap zu wenig zufällig ist.

Rjiba et al. (2015, S. 2) vergleichen in ihrer Arbeit die BHS, den CBB und den Filtered Historical Simulation. In der Arbeit von Rjiba et al. (2015, S. 1) wurde jedoch der VaR mit dem Konfidenzniveau von 99 % gewählt. Dabei wird argumentiert, dass sich der VaR bei der BHS- und der CBB-Methode zu langsam anpasst, wenn die Marktvolatilität steigt. Aus diesem Grund wird die Filtered Historical Simulation, nach Barone-Adesi, Giannopoulos und Vosper (1999), eingesetzt, welches sich als das genauere Modell erweist (Rjiba et al., 2015, S. 12). Der Vergleich der BHS und der CBB-Methode wird in der Arbeit von Rjiba et al. (2015, S. 3) jedoch nicht weiter vertieft.

## 2.4 Backtesting

Das Backtesting ist ein wichtiges Instrument zur Überprüfung der Wirksamkeit der Modelle (Jorion, 2010, S. 357). Nachfolgend werden mehrere Tests erläutert, welche die Resultate der beiden VaR-Methoden vergleichen und validieren sollen. Diese Tests werden zeigen, ob bei den berechneten Modellen möglicherweise eine Unter- oder Überschätzung der Risiken vorhanden ist. Indem der VaR rollierend über mehrere Jahre getestet wird, können dabei Aussagen gemacht werden, wie gut dieses Modell in der Vergangenheit funktioniert hätte (Hull, 2012, S. 196). Zuerst werden die Exceptions Rate und der Anteilswert-Test erläutert. Um die Zufälligkeit und Autokorrelation der Exceptions zu testen, werden nachfolgend der Runs-Test und der Ljung-Box-Test erklärt.

### 2.4.1 Exceptions Rate

Ein grundlegender Test beim VaR-Verfahren besteht darin, zu sehen, ob das angegebene Wahrscheinlichkeitsniveau $\alpha$ tatsächlich erreicht wird (Mehmke, Cremers, und Packham, 2012, S. 26). Eine Exception ist, wenn die tatsächliche 10-Tages-Portfolio-Rendite $PR_t$ der nachfolgenden Tage tiefer ist als der geschätzte $VaR_t(\alpha)$. Nach jeder Berechnung eines VaRs zum Zeitpunkt $t$, wird eine Indikatorvariable $I_t$ definiert, welche folgende zwei Zustände annehmen kann (Mehmke et al., 2012, S. 22):

$$I_t = \begin{cases} 1 & wenn\ PR_t < VaR_t(\alpha) \\ 0 & wenn\ PR_t \geq VaR_t(\alpha) \end{cases} \tag{2}$$

In dieser Arbeit wird der VaR über ein Datenset von mehr als elf Jahren mit dem Rolling-Window-Prinzip getestet, wobei sich das Fenster immer um zehn Tage verschiebt (siehe weitere Erläuterungen im Kapitel 3.2). Nach allen Durchführungen wird die Sequenz $I$ generiert, wobei 1 eine Exception und 0 keine Exception darstellt. Die nächste Abbildung zeigt ein Beispiel einer solchen Sequenz, in welcher die Exceptions ersichtlich sind.


Abbildung 2: Beispiel einer Sequenz $I$ mit 8 Exceptions (VaR 95 %)

Die Exceptions Rate sagt in Prozenten aus, wie häufig der berechnete VaR überschritten wurde (Mehmke et al., 2012, S. 23). Nachfolgend wird nun zwischen der erwarteten Exceptions Rate und der tatsächlichen Exceptions Rate unterschieden. Die erwartete Exceptions Rate $q$ wird durch

$1 - \alpha/100$ definiert, was bei einem Konfidenzniveau α von 95 %, eine erwartete Exceptions Rate $q$ von 5 % ergibt. Dabei wird die erwartete Anzahl Exceptions für das Konfidenzniveau $\alpha$ als $q \times n$ definiert, wobei $n$ der Anzahl Tests entspricht (Jorion, 2010, S. 371).

Die tatsächliche Exceptions Rate berechnet sich aus $m/n$, wobei $m$ die tatsächliche Anzahl Exceptions ist. Im Idealfall sollte die tatsächliche Exceptions Rate genau der erwarteten Exceptions Rate entsprechen. Eine höhere tatsächliche Exceptions Rate würde aussagen, dass das Modell das Risiko im Vergleich zur Realität unterschätzt (Hull, 2012, S. 197). Bei einem VaR mit einem Konfidenzintervall von 95 % wird erwartet, dass der prognostizierte VaR genau in 5 % der Fälle überschritten wird. Wenn aber der VaR in der Realität in 10 % der Fälle überschritten wird, würde dies bedeutet, dass das Risiko zu tief eingeschätzt wurde (Hull, 2012, S. 197). Umgekehrt würde eine tiefe tatsächliche Exceptions Rate bedeuten, dass das Risiko überschätzt wurde. Dies soll mit dem Anteilswert-Test überprüft werden, welches im nächsten Abschnitt erklärt wird.

### 2.4.2 Anteilswert-Test

Mit dem Anteilswert-Test wird kontrolliert, ob die Modelle korrekt spezifiziert sind (Jorion, 2010, S. 358). Damit wird die Wahrscheinlichkeit $p$ berechnet, welche aussagt, ob die tatsächliche Anzahl Exceptions $m$ der erwarteten Anzahl Exceptions entspricht, wenn die Erfolgswahrscheinlichkeit $q = 5$ % beträgt (Hull, 2012, S. 198). Dafür wird eine Nullhypothese $H_0$ formuliert, bei welcher angenommen wird, dass die Wahrscheinlichkeit, dass die Anzahl Exceptions $m$, der erwarteten Anzahl Exceptions entspricht:

$$VaR\ 95\ \%\text{:}\ \begin{cases} H_0\text{:}\ q = 5\ \% \\ H_1\text{:}\ q \neq 5\ \% \end{cases} \qquad VaR\ 85\ \%\text{:}\ \begin{cases} H_0\text{:}\ q = 15\ \% \\ H_1\text{:}\ q \neq 15\ \% \end{cases} \tag{3.1}$$

Diese Hypothese wird nun mit dem Anteilswert-Test, auch Binomialtest genannt, überprüft. Dabei wird angenommen, dass $m$ einer Binomialverteilung mit der folgenden Wahrscheinlichkeitsfunktion folgt/entspricht (Jorion, 2010, S. 359):

$$f(k = m) = \frac{n!}{k!\,(n-k)!} q^k (1-q)^{n-k} \tag{3.2}$$

Die Anzahl Exceptions $m$ wird in die Zufallsvariable $k$ eingesetzt, welche dem Resultat von $n$ - unabhängigen Bernoulli-Versuchen entspricht. Das Resultat des Anteilswert-Tests ist der P-Wert $p$, welcher die Wahrscheinlichkeit darstellt, dass die Nullhypothese irrtümlicherweise verworfen wird (Dowd, 2007, S. 181). Generell entspricht die Wahrscheinlichkeit $p$ dem kleinstmöglichen Signifikanzniveau, bei dem $H_0$ gerade noch verworfen werden könnte. Bei Hypothesen-Tests existieren zwei Fehlerarten (Jorion, 2010, S. 360):

- Fehler 1. Art (α-Fehler): Irrtümliche Ablehnung der Nullhypothese $H_0$
- Fehler 2. Art ($\beta$-Fehler): Eine falsche Nullhypothese $H_0$ wird nicht abgelehnt

Für den Test wird ein Signifikanzniveau von 5 % definiert, welches der Ablehnungsbereich in der Stichprobenverteilung für die Nullhypothese $H_0$ ist (Jorion, 2010, S. 360). Daraus wird folgende Entscheidungsregel definiert:

- Wenn $p \leq$ als 5 %: Hypothese verwerfen
- Wenn $p >$ als 5 %: Hypothese nicht verwerfen

Je höher also der P-Wert der Resultate ist, desto unwahrscheinlicher ist es, die Nullhypothese irrtümlicherweise zu verwerfen (Dowd, 2007, S. 181). Es könnte jedoch vorkommen, dass eine falsche Nullhypothese nicht verworfen wird, was den Fehler 2. Art ($\beta$-Fehler) darstellen würde (Jorion, 2010, S. 66).

Der Anteilswert-Test fokussiert sich nur auf die Anzahl Exceptions, jedoch nicht darauf, ob die Exceptions gruppiert auftreten, was häufig bei Volatilitäts-Clustern vorkommt (Jorion, 2010, S. 362). Insbesondere in Zeiten höherer Unsicherheiten kann es vorkommen, dass der VaR innerhalb einer kurzen Periode mehrmals überschritten wird (Jorion, 2010, S. 362). Da der CBB mit dem Ziehen von Blöcken versucht, diese zeitliche Struktur in der Berechnung des VaRs stärker zu berücksichtigen, sollten die Exceptions keine Zufälligkeit oder Autokorrelation aufweisen. Um auszuschliessen, dass diese zeitlichen Strukturen bestehen, wird die Zufälligkeit mit dem Runs-Test nach Wald und Wolfowitz (1943) und die Autokorrelation mit dem Ljung-Box-Test nach Ljung und Box (1978) quantifiziert, welche nachfolgend erklärt werden.

### 2.4.3 Runs-Test

Der Runs-Test ist ein nicht-parametrischer Test, welcher die Zufälligkeit einer Sequenz überprüft (Wald und Wolfowitz, 1943). Dafür wird die Nullhypothese definiert:

$$\begin{cases} H_0\text{: } Sequenz\ I\ ist\ zufällig \\ H_1\text{: } Sequenz\ I\ ist\ nicht\ zufällig \end{cases} \tag{4.1}$$

Diese Hypothese soll überprüfen, ob die Exceptions in der Sequenz $I$ zufällig vorkommen. Abgeleitet von der Sequenz $I$ werden nachfolgende Parameter definiert (Mogull, 1994, S. 297):

- $n$: Länge der Sequenz $(n_1 + n_2)$ (im Beispiel: 206)
- $n_1$: Anzahl 0-Werte (im Beispiel: 198)
- $n_2$: Anzahl 1-Werte (im Beispiel: 8)
- $r$: Anzahl Runs (im Beispiel: 15)

Der Erwartungswert $\mu_R$ ist:

$$\mu_R = \frac{2n_1n_2}{n} + 1 \tag{4.2}$$

und die Standardabweichung $\sigma_R$:

$$\sigma_R = \sqrt{\frac{2n_1n_2(2n_1n_2 - n_1 - n_2)}{n^2(n_1 + n_2 - 1)}} \tag{4.3}$$

Da das Sample gross ist, wird hier der Zentrale Grenzwertsatz angewendet. Dabei wird davon ausgegangen, dass die Verteilung eines grossen Samples die Normalverteilung approximiert. Mit folgender Definition wird der Z-Score berechnet (Mogull, 1994, S. 297):

$$z = \frac{r - \mu_R}{\sigma_R} \sim N(0,1) \tag{4.4}$$

Mit dem Z-Score wird anschliessend der P-Wert ermittelt:

$$p = 2P(z > |z|) \tag{4.5}$$

Aufgrund der berechneten P-Werte kann für die beschriebene Hypothese (4.1) folgende Vorgehensweise definiert werden:

- Wenn $p \leq$ als 5 %: $H_0$ verwerfen, die Sequenz ist nicht zufällig.
- Wenn $p >$ als 5 %: $H_0$ nicht verwerfen, es kann jedoch nicht gefolgert werden, dass die Reihenfolge der Daten nicht zufällig ist.

### 2.4.4 Ljung-Box-Test

Der Ljung-Box-Test ist ein statistischer Test, bei welchem die Autokorrelation über mehrere Lags getestet wird (Ljung und Box, 1978). Ein Lag entspricht einer zeitlichen Verschiebung. Bei der Anwendung wird die Anzahl Lags gewählt, in der das Modell nach Autokorrelation suchen soll. Dafür wird der Ljung-Box-Test auf die Sequenz $I$ der Exceptions angewendet (Burns, 2002a, S. 8). Auch hier wird eine Nullhypothese $H_0$ formuliert:

$$\begin{cases} H_0\text{: } Sequenz\ hat\ keine\ signifikante\ Autokorrelation \\ H_1\text{: } Sequenz\ hat\ eine\ signifikante\ Autokorrelation \end{cases} \tag{5.1}$$

Die Autokorrelation bei Lag $k$ der Sequenz $x_t$ (mit Mittelwert null) mit der Länge $n$ ist (Burns, 2002b, S. 4):

$$r_k = \frac{\sum_{t=k+1}^{n} x_t x_{t-k}}{\sum_{t=1}^{n} x_t^2} \tag{5.2}$$

Die Teststatistik $Q$ mit der Anzahl Lags $M$ wird definiert durch:

$$Q_M = n(n+2) \sum_{k=1}^{M} \frac{r_k^2}{n-k} \tag{5.3}$$

Für das Signifikanzniveau α, entspricht der kritische Wert für das Verwerfen der Nullhypothese $H_0$:

$$Q_M > x_{1-\alpha,h}^2 \tag{5.4}$$

wobei das Resultat $x_{1-\alpha,h}^2$ das 1-α Quantil der Chi-Quadrat-Verteilung mit $h$ Freiheitsgrade darstellt (Ljung und Box, 1978, S. 3). Dabei wird folgendermassen vorgegangen:

- Wenn $p \leq$ als 5 %: $H_0$ verwerfen, die Sequenz hat keine signifikante Autokorrelation.
- Wenn $p >$ als 5 %: $H_0$ nicht verwerfen, es kann jedoch nicht gefolgert werden, dass die Sequenz eine Autokorrelation hat.

# 3 Methodisches Vorgehen

## 3.1 Portfolio und Datenset

Für die Simulation wird ein Portfolio definiert, welches aus fünf Aktien besteht: Novartis, Logitech, LafargeHolcim, Swiss Re und Nestle. Dafür wurden die täglichen adjustierten Schlusskurse dieser fünf Aktien von Refinitiv herangezogen. Das Datenset umfasst Kurse vom 04.01.2010 – 16.03.2021 und beinhaltet insgesamt 2814 Datenpunkte. Die Entwicklung der Portfolio Renditen ist in der folgenden Abbildung ersichtlich. Dabei fallen insbesondere die Unsicherheiten an den Finanzmärkten im September 2011, die Aufhebung des Euro-Mindestkurses im Januar 2015 und die Corona-Krise anfangs März 2020 auf.

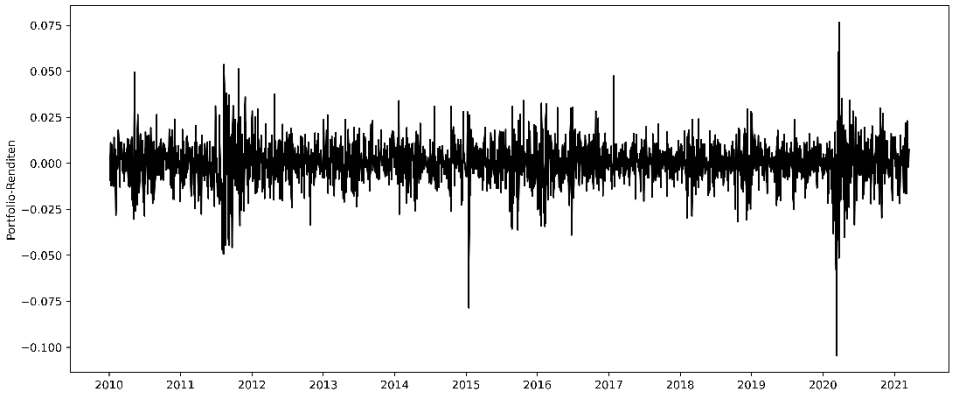



Abbildung 3: Entwicklung der täglichen Portfolio-Renditen vom 05.01.2010 – 16.03.2021

Die Abbildung 4 zeigt die Verteilung der Renditen in einem Histogramm.

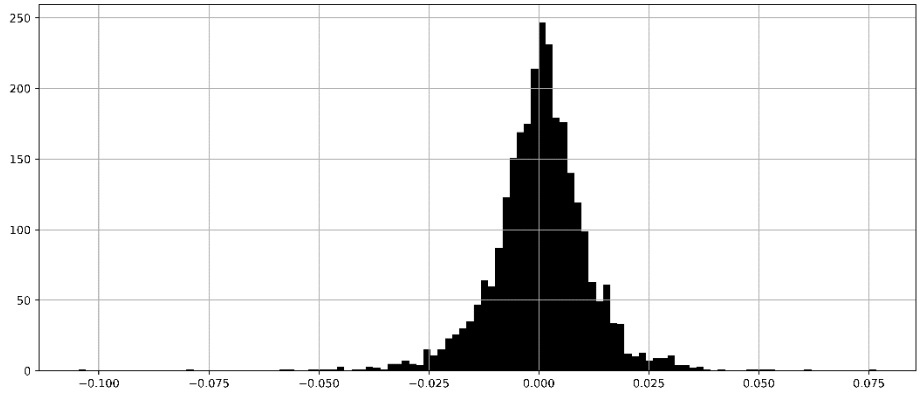



Abbildung 4: Verteilung der Portfolio-Renditen in einem Histogramm

Die folgenden Abbildungen zeigen die Entwicklung des Mittelwertes, der Standardabweichung und der Autokorrelation für das 750-Tages-Rolling-Window.

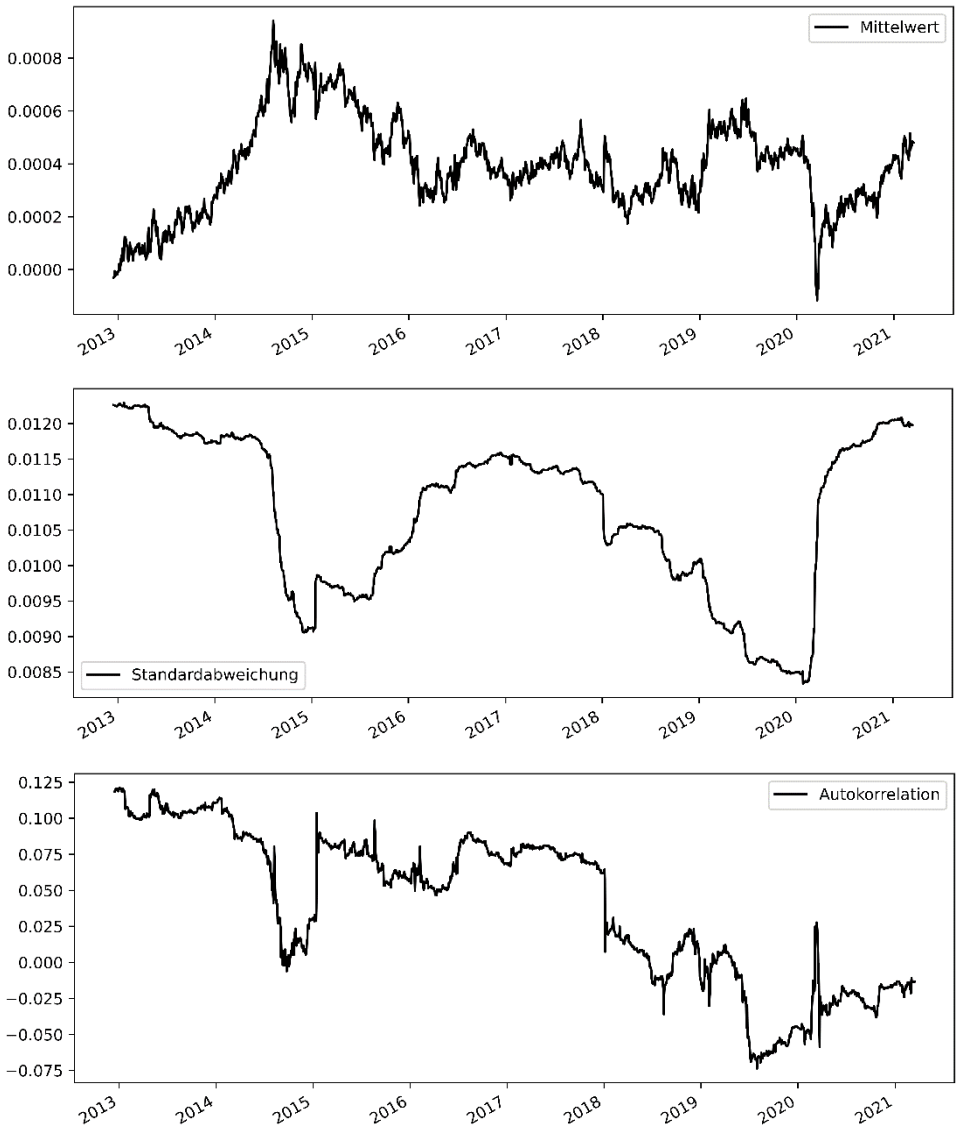



Abbildung 5: Mittelwert, Standardabweichung und Autokorrelation

In den oben aufgeführten Grafiken ist ersichtlich, dass sich der Mittelwert nahe bei null bewegt, während die Autokorrelation und die Standardabweichung eine höhere Fluktuation über die Zeit aufweisen.

| | | | |
|---|---|---|---|
| **Mittelwert:** | 0.000342 | **Standardabweichung:** | 0.011292 |
| **Minimum:** | -0.104574 | **Maximum:** | 0.076604 |
| **Autokorrelation:** | 0.049765 | **Jarque-Bera-Test:** | 4932 |
| **Skewness:** | -0.464371 | **Kurtosis:** | 6.434136 |

Tabelle 1: Zusammenfassung der Statistik der Portfolio-Renditen

Die Tabelle 1 zeigt, dass der Mittelwert nahe bei null ist und es grosse Ausreisser in beide Richtungen hat, wie dies das Minimum und das Maximum zeigen. Mit der Skewness (Schiefe) und

der Kurtosis (Wölbung) wird die Verteilung der Portfolio-Renditen beschrieben. Eine Normalverteilung hätte eine Schiefe von null und ist somit symmetrisch und eine Kurtosis von drei (Dowd, 2007, S. 19). Im Vergleich dazu weist die Verteilung der Portfolio-Renditen jedoch eine negative Skewness auf, was darauf hindeutet, dass extreme negative Renditen häufiger vorkommen als in der Normalverteilung erwartet wird (Dowd, 2007, S. 19). Weiter ist die Verteilung der Renditen leptokurtic (steilgipflig), da die Kurtosis grösser als drei ist. Dies bedeutet, dass extreme Ereignisse häufiger vorkommen, als unter einer Normalverteilung angenommen wird (Dowd, 2007, S. 19).

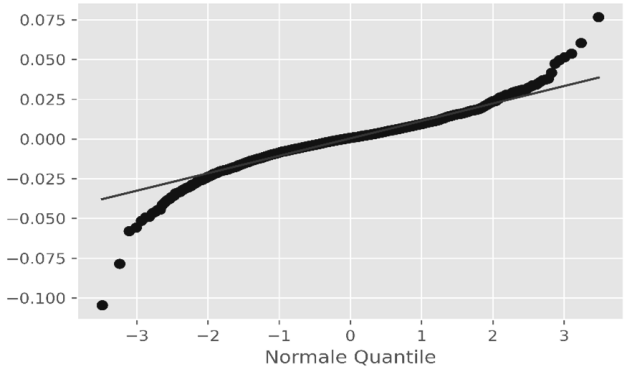



Abbildung 6: Q-Q Plot der täglichen Portfolio-Renditen

Im Q-Q Plot in der Abbildung 6 ist ersichtlich, dass bei den täglichen Portfolio-Renditen grössere Tails vorhanden sind und somit eine Abweichung zur Normalverteilung besteht. Dies unterstreicht auch das Resultat des Jarque-Bera-Tests in der Tabelle 1. Je weiter das Resultat des Jarque-Bera-Tests von Null entfernt ist, desto stärker weicht die Verteilung von der Normalverteilung ab (Jorion, 2010, S. 68). Dies zeigt, dass die Berechnung des VaRs mit dem Varianz-Kovarianz-Ansatz zu ungenauen Resultaten führt, da das Modell von einer Normalverteilung ausgeht.

## 3.2 Implementation

Die Implementation ist mit der Programmiersprache Python umgesetzt, wessen Code im Anhang 3 ersichtlich ist. Der Verständlichkeit halber wird in dieser Arbeit, sowie in den Kommentaren im Python-Code vom VaR mit dem Konfidenzniveau 95 %, von einer Bootstrap-Sample-Grösse $T = 750$ Tagen und einer Anzahl von 1000 Simulationen gesprochen. Die Tabelle 2 zeigt die Kombinationen der verschiedenen Sample-Grössen und Anzahl Simulationen, welche überprüft werden. Jede Kombination wird fünf Mal wiederholt, um sicherzustellen, dass die Resultate konstant sind. Alle Kombinationen der Tabelle 2 werden sowohl für das Konfidenzniveau 95 % als auch für das Konfidenzniveau 85 % gerechnet.

| Anzahl Simulationen | Sample-Grössen: 250 | 500 | 750 | 1000 |
|---|---|---|---|---|
| 1000 | S_1000_L_250_V1 | S_1000_L_500_V1 | S_1000_L_750_V1 | S_1000_L_1000_V1 |
| | S_1000_L_250_V2 | S_1000_L_500_V2 | S_1000_L_750_V2 | S_1000_L_1000_V2 |
| | S_1000_L_250_V3 | S_1000_L_500_V3 | S_1000_L_750_V3 | S_1000_L_1000_V3 |
| | S_1000_L_250_V4 | S_1000_L_500_V4 | S_1000_L_750_V4 | S_1000_L_1000_V4 |
| | S_1000_L_250_V5 | S_1000_L_500_V5 | S_1000_L_750_V5 | S_1000_L_1000_V5 |
| 2000 | S_2000_L_250_V1 | S_2000_L_500_V1 | S_2000_L_750_V1 | S_2000_L_1000_V1 |
| | S_2000_L_250_V2 | S_2000_L_500_V2 | S_2000_L_750_V2 | S_2000_L_1000_V2 |
| | S_2000_L_250_V3 | S_2000_L_500_V3 | S_2000_L_750_V3 | S_2000_L_1000_V3 |
| | S_2000_L_250_V4 | S_2000_L_500_V4 | S_2000_L_750_V4 | S_2000_L_1000_V4 |
| | S_2000_L_250_V5 | S_2000_L_500_V5 | S_2000_L_750_V5 | S_2000_L_1000_V5 |
| 3000 | S_3000_L_250_V1 | S_3000_L_500_V1 | S_3000_L_750_V1 | S_3000_L_1000_V1 |
| | S_3000_L_250_V2 | S_3000_L_500_V2 | S_3000_L_750_V2 | S_3000_L_1000_V2 |
| | S_3000_L_250_V3 | S_3000_L_500_V3 | S_3000_L_750_V3 | S_3000_L_1000_V3 |
| | S_3000_L_250_V4 | S_3000_L_500_V4 | S_3000_L_750_V4 | S_3000_L_1000_V4 |
| | S_3000_L_250_V5 | S_3000_L_500_V5 | S_3000_L_750_V5 | S_3000_L_1000_V5 |
| | 256 VaR-Tests | 231 VaR-Tests | 206 VaR-Tests | 181 VaR-Tests |

Tabelle 2: Kombinationen der Durchführungen

Aus dem Datenset werden zuerst die ersten 750 Tage $t_{0-750}$ als Sample für das Bootstrapping und die folgenden zehn Tage $t_{750-760}$ für das Backtesting gewählt. Die Log-Renditen der einzelnen Aktien werden zur 10-Tages-Log-Rendite zeitlich summiert und mit der nachfolgenden Formel in diskrete Renditen umgerechnet.

$$R_{Diskret} = e^{R_{Stetig}} - 1 \tag{6}$$

Die Portfolio-Rendite wird anhand des Mittelwerts der Renditen der einzelnen Aktien gerechnet. Um die Simulationen auf allen Sample-Grössen anwenden zu können und damit jede Simulation jeweils zehn Tage nach dem Sample für das Backtesting zur Verfügung steht, wird eine Korrektur des ursprünglichen Samples durchgeführt. Mit dieser Korrektur wird die Länge des Datensets definiert. Bei jedem Durchgang wird zuerst das Bootstrap-Sample festgelegt, welches beim ersten Durchgang den Renditen mit Index $t_{0-750}$ entspricht. Anhand dieses Samples wird das Bootstrapping durchgeführt und der VaR berechnet. Die darauffolgenden zehn Tage $t_{750-760}$ werden für das Backtesting verwendet, welche im Anschluss mit dem vorherig berechneten VaR verglichen werden. Anhand der Formel (2) wird eruiert, ob es sich um eine Exception handelt.

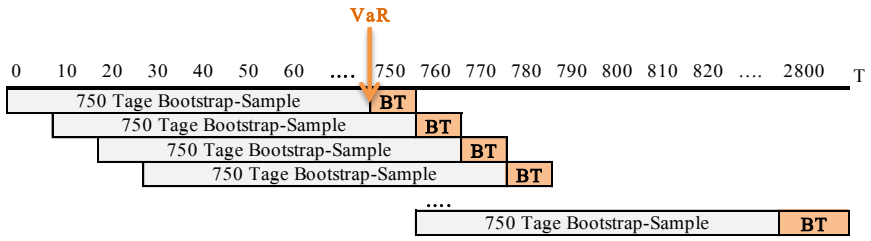



Abbildung 7: Beispiel des Rolling-Window-Prinzips (BT = Backtesting)

Dieser ganze Prozess wird jeweils mit dem Rolling-Window-Prinzip über das ganze Datenset, welches 2814 Renditen von über elf Jahren beinhaltet, durchgeführt (Zivot und Wang, 2001, S. 308). Das heisst, im zweiten Durchgang wird erneut das Bootstrap-Sample definiert, wobei sich das Fenster um zehn Tage verschiebt. Dafür werden im zweiten Durchgang die Renditen mit

Index $t_{10-760}$ als Bootstrap-Sample genommen und $t_{760-770}$ für das Backtesting (Zivot und Wang, 2001, S. 308).

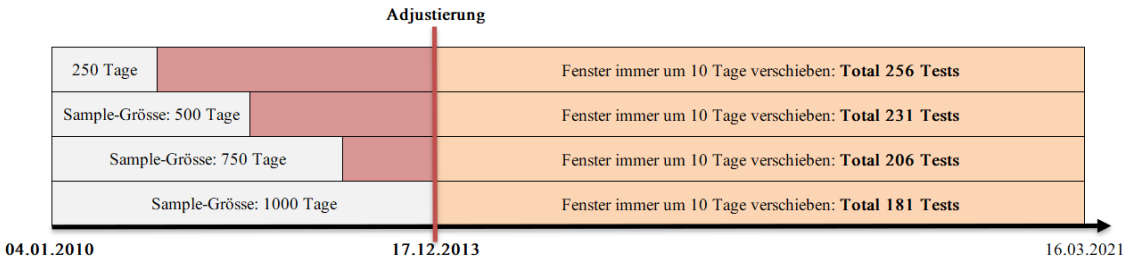



Abbildung 8: Illustration der Adjustierung

Die Abbildung 8 zeigt, dass bei den verschiedenen Bootstrap-Sample-Grössen eine andere Anzahl Tests durchgeführt wird. Dies bedeutet, dass der Beobachtungszeitrum über die verschiedenen Sample-Grössen hinweg nicht deckungsgleich ist, was zu leichten Unterschieden führen kann, insbesondere im Vergleich der verschiedenen Sample-Grössen. In dieser Arbeit wird diese Adjustierung ausschliesslich beim Vergleich der verschiedenen Sample-Grössen angewendet, da alle Sample-Grössen über den gleichen Beobachtungszeitraum hinweg verglichen werden sollen. Somit werden nur die Resultate der berechneten VaRs ab dem 17.12.2013 berücksichtigt und nicht bereits vom 04.01.2010.

### 3.2.1 BHS-Simulation

Bei der BHS werden vom Bootstrap-Sample zufällig zehn einzelne Tage mit Zurücklegen gezogen und die Renditen aller fünf Aktien gewählt. Diese Log-Renditen werden zeitlich summiert und in eine diskrete 10-Tages-Portfolio-Rendite umgerechnet. Dieser Prozess wird danach 1000 Mal wiederholt, was eine Tabelle, bestehend aus 1000 Portfolio-Renditen, ergibt. Die 51 tiefste Portfolio-Rendite entspricht demnach dem VaR 95 %.

### 3.2.2 CBB-Simulation

Im gleichen Schritt wie die Berechnung des BHS-VaRs, wird auch der VaR des CBBs berechnet. Dabei wird jeweils eine Zufallszahl zwischen null und 750 gezogen und davon ein Block von zehn aufeinanderfolgenden Tagen mitgenommen. Das Circular-Regelwerk kommt nur dann zum Einsatz, wenn der Block über das Bootstrap-Sample gehen würde. Zum Beispiel wenn die Zufallszahl $t_{747}$ entspricht und das Sample bei $t_{750}$ endet, würde der Block nur aus drei Tagen bestehen. Um dieses Problem zu beheben, wird das Ende des Samples mit dem Anfang des Samples verbunden. Es werden die restlichen sieben Tage mit dem Index von $t_{0-7}$ herangezogen. Der

komplette Block besteht in diesem Fall aus $t_{747-750}$ und $t_{0-7}$. Wie bei der BHS-Methode werden diese zeitlich summiert und in eine diskrete 10-Tages-Portfolio-Rendite umgerechnet. Auch hier wird der Prozess 1000 Mal wiederholt und die 51 tiefste Rendite als VaR definiert.

### 3.2.3 Backtesting

Um die beiden Bootstrap-Methoden miteinander zu vergleichen, werden die Resultate mit diversen Backtesting-Methoden evaluiert. Bei jedem Durchgang wird eruiert, ob es sich um eine Exception handelt. Die nachfolgende Abbildung zeigt die berechneten VaRs der beiden Bootstrap-Methoden und die tatsächlichen Portfolio-Renditen, welche mit dem Rolling-Window in 206 Durchgängen (bei einer Sample-Grösse von 750) über das Datenset hinweg berechnet werden. An den Stellen, wo die tatsächliche 10-Tages-Portfolio-Rendite die 10-Tages-VaR unterschreitet, handelt es sich um Exceptions.

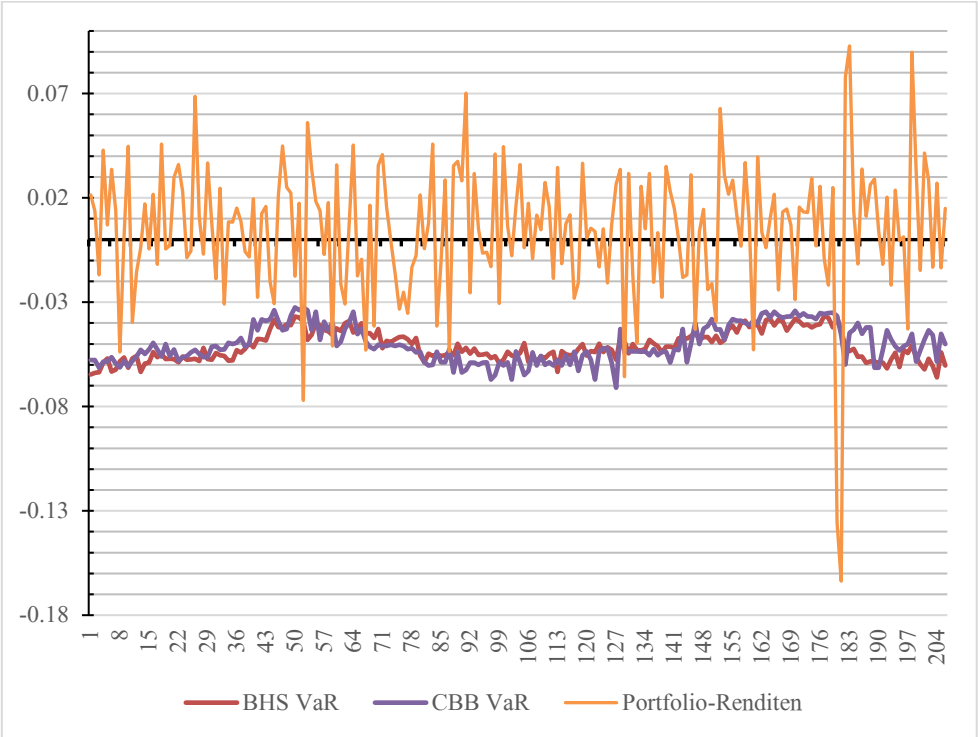



Abbildung 9: Beispiel der VaR 95 % im Vergleich zu den 10-Tages-Portfolio-Renditen

Mit der erhaltenen Exceptions Rate wird anschliessend der Anteilswert-Test durchgeführt. Dies wurde in Python mit der Funktion *scipy.stats.binom_test* implementiert, wobei zweiseitig getestet wurde. Zusätzlich wird im Parameter der Python-Funktion keine Korrektur von 0.5 gemacht, da das Sample grösser als 50 ist. Der Runs-Test und der Ljung-Box-Test werden mit den Funktionen *runstest_1samp* und *stats.acorr_ljungbox* berechnet. Der Code zum Vorgehen beim Backtesting ist ebenfalls im Anhang 3 ersichtlich.

# 4 Resultate

Nach der Durchführung der Berechnungen, welche im vorherigen Kapitel beschrieben wurden, werden nun die Resultate analysiert und ausgewertet. Im Anhang 1 werden die Resultate zusätzlich nach der Anzahl Simulationen aufgezeigt. Im Anhang 2 befinden sich die Resultate der CBB-Methode, welche die Resultate gruppiert nach den verschiedenen Sample-Grössen und in Abhängigkeit mit den Anzahl Simulationen aufzeigen. Wichtig hervorzuheben ist, dass eine Adjustierung des Beobachtungszeitraums ausschliesslich bei den Resultaten nach Sample-Grössen und im Anhang 2 durchgeführt wird, damit die einzelnen Sample-Grössen nur über den gleichen Zeitraum hinweg miteinander verglichen werden (siehe Abbildung 8).

## 4.1 Exceptions Rate und Anteilswert-Test

Die tatsächliche Exceptions Rate zeigt, wie gut das jeweilige Modell das Risiko einschätzt und sollte idealerweise die erwartete Exceptions Rate approximieren. In der nächsten Abbildung ist ersichtlich, dass die CBB-Methode für beide Konfidenzniveaus näher bei der erwarteten Exceptions Rate (5 % und 15 %) liegt als die BHS-Methode.

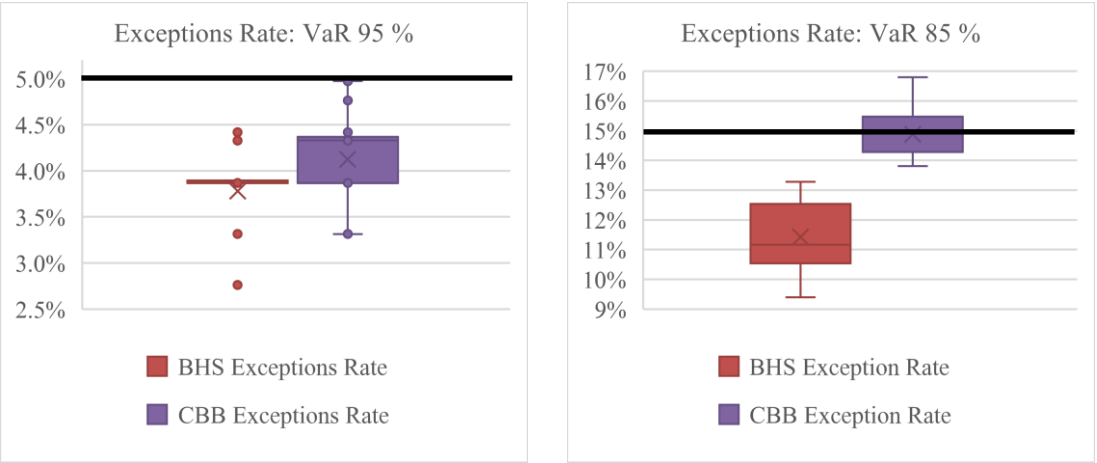
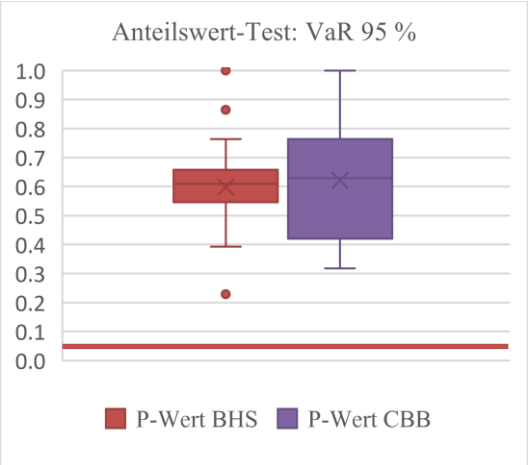



Abbildung 10: Boxplots der Verteilung der Exceptions Rates

Zusätzlich wird mit dem Anteilswert-Test die Hypothese (3.1) getestet, ob die Anzahl Exceptions den erwarteten Exceptions entspricht. Die Nullhypothese $H_0$ des Anteilswert-Tests wird bei der BHS-Methode beim VaR 85 % ein Mal verworfen.



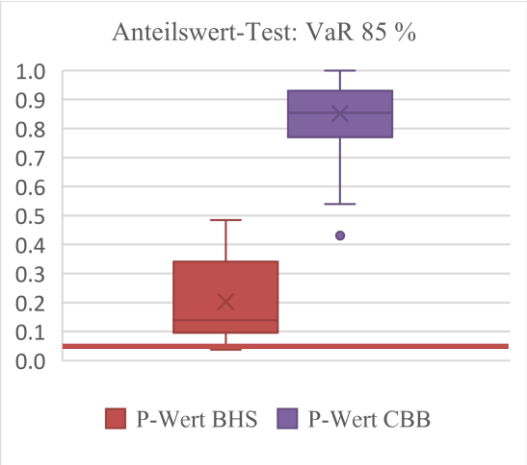
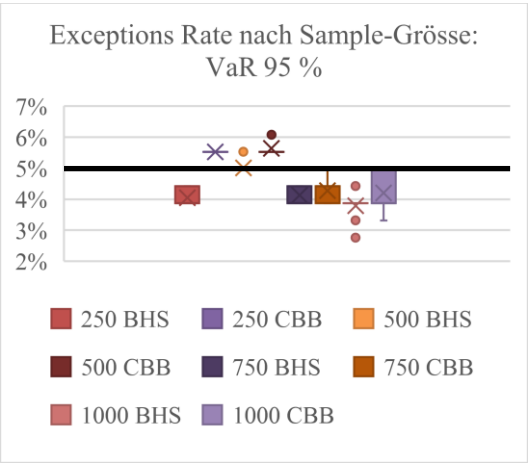
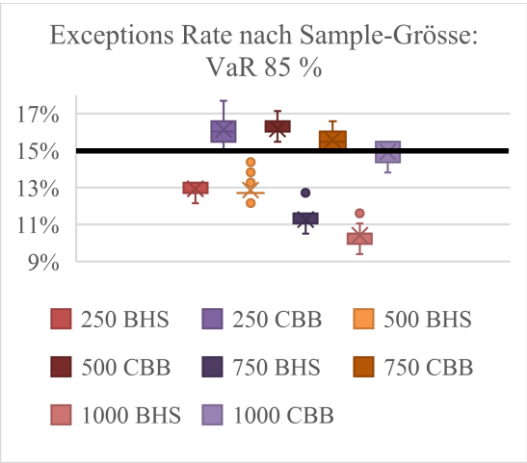
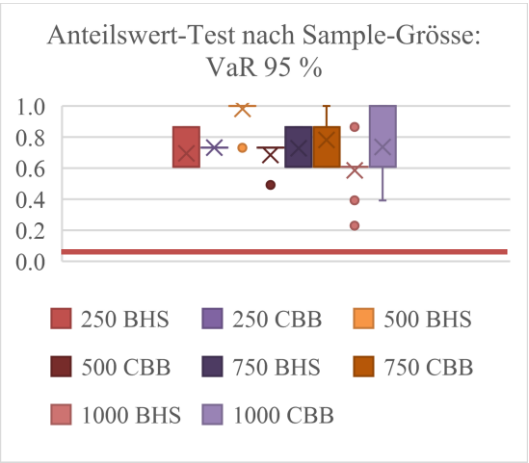




Abbildung 11: Boxplots der P-Werte des Anteilswert-Tests

In der Abbildung 11 handelt es sich um die Kombination *S_1000_L_1000_V3*, welche einen P-Wert von 0.0366 aufweist, da die Exceptions Rate bei 9.392 % liegt und damit das Risiko um mehr als 5 % unterschätzt. In der vorherigen Abbildung ist zu sehen, dass die CBB-Methode mehrheitlich höhere P-Werte hat und somit die Wahrscheinlichkeit, die Nullhypothese $H_0$ irrtümlich zu verwerfen, tiefer ist.



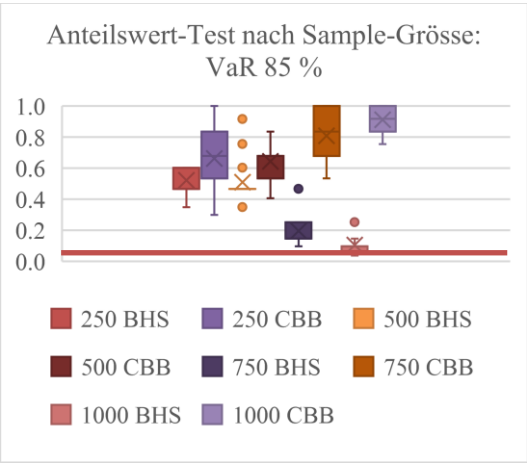




Abbildung 12: Boxplots der Exceptions Rates und Anteilswert-Tests nach Sample-Grösse

Bei den Resultaten nach Sample-Grösse wird festgestellt, dass die CBB-Methode, bis auf eine Ausnahme, über alle Sample-Grössen höhere P-Werte aufweist als die BHS-Methode. Dabei handelt es sich um den VaR 85 % mit der Sample-Grösse von 500 Tagen. Dies sagt aus, dass die CBB-Methode das Risiko genauer einschätzt als die BHS-Methode. Eine weitere Erkenntnis ist, dass über beide Konfidenzniveaus hinweg die Sample-Grössen von 750 und 1000 Tagen bei der CBB-Methode das Risiko am genausten einschätzen können.

## 4.2 Runs-Test

Beim Runs-Test wird die Zufälligkeit der Exceptions geprüft. In der nachfolgenden Abbildung ist ersichtlich, dass die Nullhypothese $H_0$ (4.1) beim Runs-Test für die BHS beim VaR 95 % ein Mal verworfen wird, da der P-Wert unter 0.05 liegt. Dies ereignet sich bei der Kombination *S_1000_L_1000_V5*, wobei der P-Wert 0.012 ist. Die CBB-Methode hingegen hat eine deutlich höhere Wahrscheinlichkeit, die Nullhypothese nicht zu verwerfen und bestätigt somit die Aussage, dass die Exceptions zufällig auftreten.

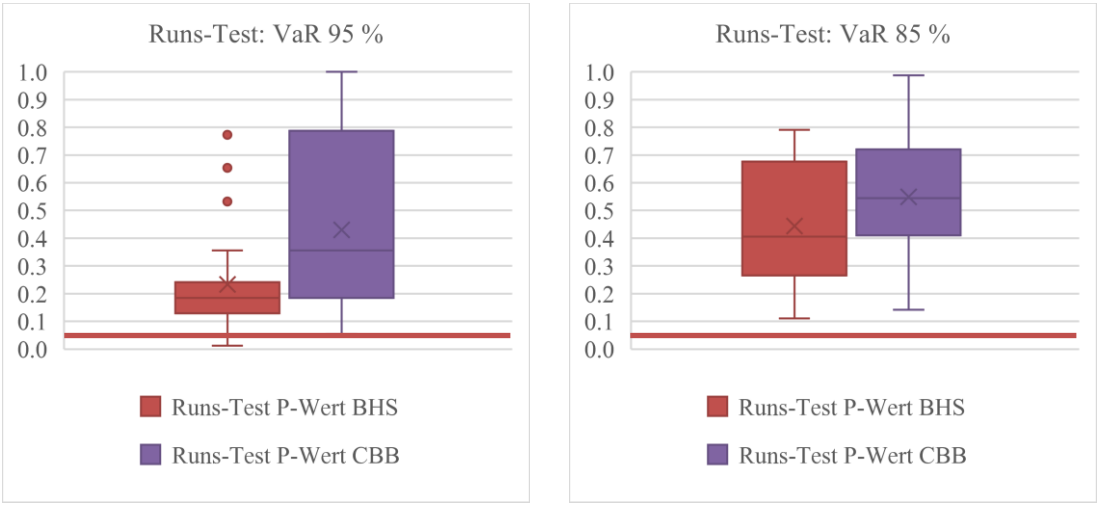
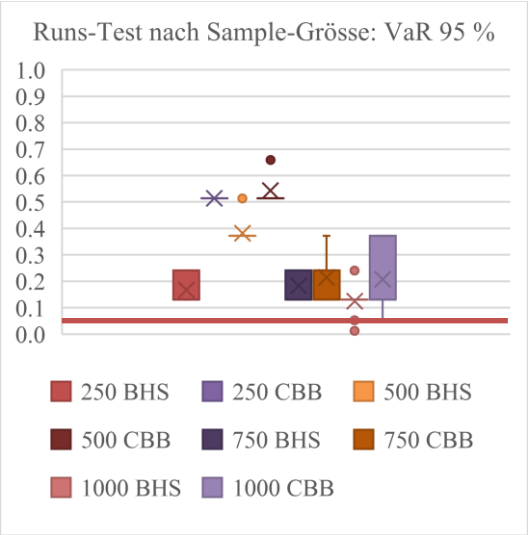



Abbildung 13: Boxplots der P-Werte aus dem Runs-Test

Die Resultate, gruppiert nach Sample-Grösse, zeigen, dass die BHS-Methode einzig beim VaR 85 % mit der Sample-Grösse von 250 Tagen höhere P-Werte aufweist als die CBB-Methode. Zu berücksichtigen ist, dass die CBB-Methode beim VaR 85 % bis zu 4 % mehr Exceptions generiert als die BHS-Methode. Der Median der Exceptions Rates der CBB-Methode liegt bei 15 % und jener der BHS-Methode bei 11 %. Ausserdem beweist die CBB-Methode, dass auch die zusätzlichen Exceptions zufällig verteilt sind.



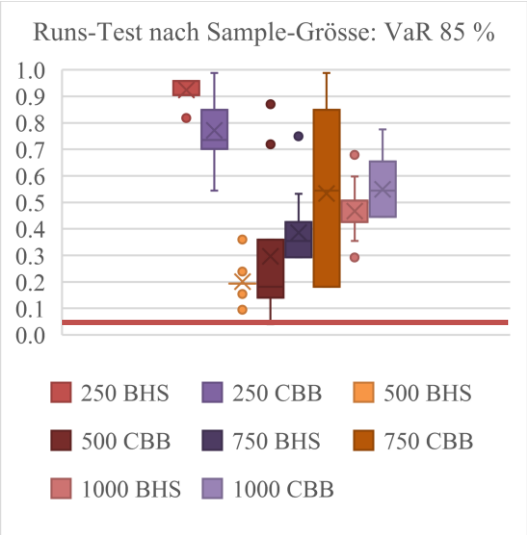
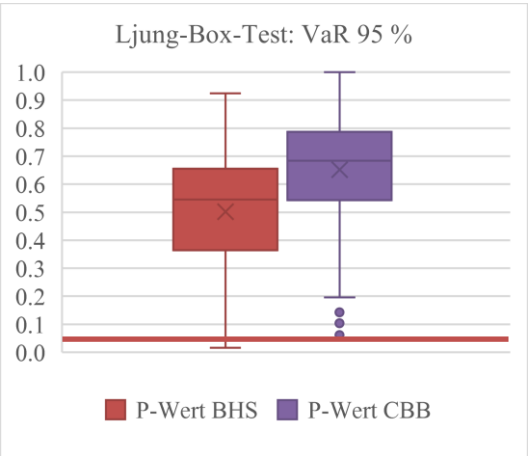




Abbildung 14: Boxplots der P-Werte aus dem Runs-Test nach Sample-Grösse

## 4.3 Ljung-Box-Test

Beim Ljung-Box-Test wird die Autokorrelation bis zu zehn Lags getestet. Nachfolgend ist beim VaR 95 % ersichtlich, dass die BHS die Nullhypothese $H_0$ (5.1) 16 Mal verwirft und die CBB-Methode einen höheren P-Wert hat. Beim VaR 85 % hingegen sind beide Methoden etwa gleich verteilt. Dennoch wird beim VaR 85 % die BHS-Methode vier Mal verworfen, während die CBB-Methode nur ein Mal verworfen wird. Bei beiden Konfidenzniveaus sind Ausreisser zu sehen.



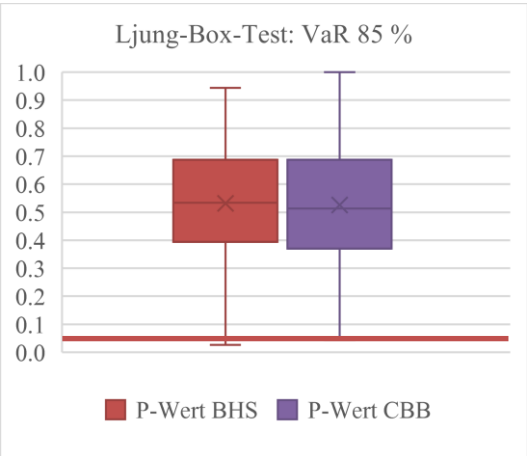




Abbildung 15: Boxplots der P-Werte aus dem Ljung-Box-Test

In der Tabelle 3 sind die Fälle abgebildet, bei denen die Nullhypothese $H_0$ verworfen werden. Diese ereignen sich hauptsächlich beim VaR 95 % bei einer Sample-Grösse von 250 Tagen. Dabei handelt es sich um die P-Werte der Lags von drei bis und mit fünf. Es gibt einen einzelnen Fall bei einer Sample-Grösse von 1000 Tagen, wo die BHS-Methode die Hypothese verwirft. Beim

VaR 85 % wird die Nullhypothese beim Lag zehn bei der Sample-Grösse von 750 Tagen drei Mal verworfen.

**VaR 95 %**

| Durchgang | Sample-Grösse | Lag | Q_BHS | P-Wert BHS | H0 BHS | Q_CBB | P-Wert CBB | H0 CBB |
|---|---|---|---|---|---|---|---|---|
| S_1000_L_250_V2 | 250 | 3 | 9.70 | 0.0213 | Verwerfen | 1.13 | 0.7690 | Nicht verwerfen |
| S_1000_L_250_V2 | 250 | 4 | 10.61 | 0.0314 | Verwerfen | 1.13 | 0.8889 | Nicht verwerfen |
| S_1000_L_250_V2 | 250 | 5 | 11.52 | 0.0420 | Verwerfen | 2.35 | 0.7990 | Nicht verwerfen |
| S_2000_L_250_V2 | 250 | 3 | 9.70 | 0.0213 | Verwerfen | 1.13 | 0.7690 | Nicht verwerfen |
| S_2000_L_250_V2 | 250 | 4 | 10.61 | 0.0314 | Verwerfen | 1.13 | 0.8889 | Nicht verwerfen |
| S_2000_L_250_V2 | 250 | 5 | 11.52 | 0.0420 | Verwerfen | 2.35 | 0.7990 | Nicht verwerfen |
| S_2000_L_250_V3 | 250 | 3 | 9.70 | 0.0213 | Verwerfen | 1.13 | 0.7690 | Nicht verwerfen |
| S_2000_L_250_V3 | 250 | 4 | 10.61 | 0.0314 | Verwerfen | 1.13 | 0.8889 | Nicht verwerfen |
| S_2000_L_250_V3 | 250 | 5 | 11.52 | 0.0420 | Verwerfen | 2.35 | 0.7990 | Nicht verwerfen |
| S_3000_L_250_V3 | 250 | 3 | 9.70 | 0.0213 | Verwerfen | 2.66 | 0.4477 | Nicht verwerfen |
| S_3000_L_250_V3 | 250 | 4 | 10.61 | 0.0314 | Verwerfen | 2.67 | 0.6142 | Nicht verwerfen |
| S_3000_L_250_V3 | 250 | 5 | 11.52 | 0.0420 | Verwerfen | 3.73 | 0.5887 | Nicht verwerfen |
| S_3000_L_250_V4 | 250 | 3 | 9.70 | 0.0213 | Verwerfen | 2.66 | 0.4477 | Nicht verwerfen |
| S_3000_L_250_V4 | 250 | 4 | 10.61 | 0.0314 | Verwerfen | 2.67 | 0.6142 | Nicht verwerfen |
| S_3000_L_250_V4 | 250 | 5 | 11.52 | 0.0420 | Verwerfen | 3.73 | 0.5887 | Nicht verwerfen |
| S_1000_L_1000_V5 | 1000 | 1 | 5.77 | 0.0163 | Verwerfen | 0.76 | 0.3830 | Nicht verwerfen |

**VaR 85 %**

| Durchgang | Sample-Grösse | Lag | Q_BHS | P-Wert BHS | H0 BHS | Q_CBB | P-Wert CBB | H0 CBB |
|---|---|---|---|---|---|---|---|---|
| S_1000_L_750_V1 | 750 | 10 | 18.56 | 0.0462 | Verwerfen | 9.19 | 0.5137 | Nicht verwerfen |
| S_1000_L_750_V3 | 750 | 10 | 18.56 | 0.0462 | Verwerfen | 11.61 | 0.3118 | Nicht verwerfen |
| S_2000_L_750_V3 | 750 | 10 | 20.28 | 0.0267 | Verwerfen | 12.08 | 0.2799 | Nicht verwerfen |
| S_3000_L_1000_V2 | 1000 | 10 | 19.34 | 0.0362 | Verwerfen | 9.42 | 0.4927 | Nicht verwerfen |
| S_1000_L_500_V2 | 500 | 10 | 12.58 | 0.2481 | Nicht verwerfen | 18.63 | 0.0453 | Verwerfen |

Tabelle 3: Verwerfung der Nullhypothese im Ljung-Box-Test

Die CBB-Methode verwirft die Nullhypothese nur ein Mal beim VaR 85 % bei der Sample-Grösse von 500 Tagen.

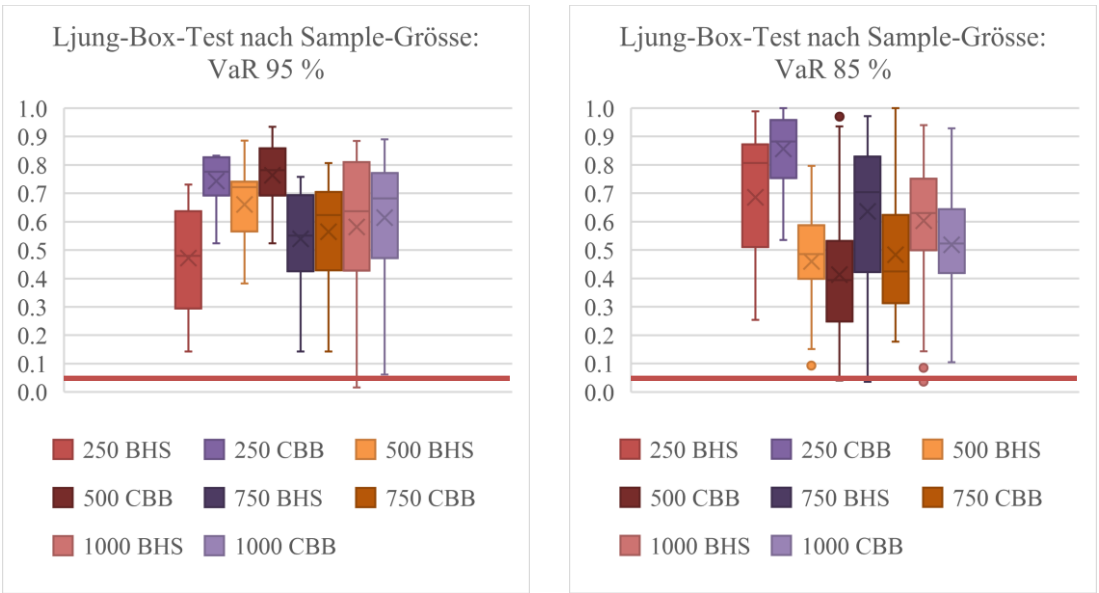



Abbildung 16: Boxplots der P-Werte aus dem Ljung-Box-Test nach Sample-Grösse

Zudem weist die CBB-Methode im Ljung-Box-Test beim VaR 95 % über alle Sample-Grössen hinweg höhere P-Werte auf. Beim VaR 85 % hingegen hat die CBB-Methode nur bei der Sample-Grösse von 250 Tagen höhere P-Werte. Dennoch kann gesagt werden, dass die Exceptions bei

der CBB-Methode keine Autokorrelation aufweisen und im Vergleich zur BHS-Methode das bessere Modell ist. Insbesondere beim Anteilswert-Test erweisen sich die Sample-Grössen von 750 und 1000 Tagen als effektiver. Bei denen wird die Nullhypothese nie verworfen. Sie weisen aber nicht immer die höchsten P-Werte auf.

Als Fazit kann gesagt werden, dass bei der CBB-Methode die nötigen Annahmen für den Anteilswert-Test erfüllt sind, da die Nullhypothesen des Ljung-Box-Tests und Runs-Tests nicht verworfen werden und somit die Exceptions zufällig und keine Autokorrelation aufweisen. Mit dieser Voraussetzung können die Resultate des Anteilswert-Tests interpretiert werden und schlussfolgern, dass die CBB-Methode das exaktere Modell ist. Insbesondere bei den Sample-Grössen von 750 und 1000 Tagen liefert die CBB-Methode die genausten Schätzungen. Im Anhang 2 werden die Resultate für die CBB-Methode ausführlicher aufgezeigt.

# 5 Schlussfolgerung

In dieser Arbeit wurde untersucht, ob der Circular Block Bootstrap zu einer besseren Schätzung des 10-Tages-VaRs als der Standard Bootstrap führt. Dabei wurde überprüft, ob das Ziehen von Blöcken, anstelle einzelner Tage, die Interdependenzstruktur des Samples stärker berücksichtigt und somit den VaR genauer prognostizieren kann. Diese Fragestellung wurde anhand eines Portfolios von fünf Aktien (Logitech, Novartis, Swiss Re, LafargeHolcim und Nestle) über die Zeitspanne vom 04.01.2010 – 16.03.2021 getestet. Dabei wurden die beiden Bootstrap-Methoden anhand der verschiedenen Sample-Grössen von 250, 500, 750 und 1000 Tagen mit je 1000, 2000 und 3000 Simulationen berechnet. Da das Konfidenzniveau 95 % nur 5 % Exceptions generiert, welche auf Zufälligkeit und Autokorrelation getestet werden können, wurden die beiden Bootstrap-Methoden zusätzlich mit dem Konfidenzniveau 85% verglichen.

Die Resultate zeigen, dass die CBB-Methode über die beiden Konfidenzniveaus 95 % und 85 % den VaR genauer einschätzt als die BHS-Methode. Zusätzlich konnte bewiesen werden, dass der Circular Block Bootstrap das Auftreten von Volatilitäts-Clustern besser ausgleicht, indem die Exceptions zufällig vorkommen und keine Autokorrelation aufweisen. Die CBB-Methode verwirft im Runs-Test die Nullhypothese nie und zeigt zudem höhere P-Werte auf als die BHS-Methode. Auch beim Ljung-Box-Test sprechen die Resultate für die CBB-Methode, da bei der BHS-Methode die Nullhypothese häufiger verworfen wird.

Eine weitere Schlussfolgerung ist, dass die CBB-Methode das Risiko bei den Sample-Grössen von 750 und 1000 Tagen am besten einschätzt. Ausserdem kann gesagt werden, dass die nötigen Annahmen für die Interpretation erfüllt sind, da auch bei diesen Sample-Grössen die Nullhypothesen des Runs-Tests und des Ljung-Box-Tests nicht verworfen werden. Die genauen Resultate für die CBB-Methode sind im Anhang 2 zu finden.

Weiter wurde festgestellt, dass die Anzahl Simulationen (1000, 2000 und 3000) bei beiden Methoden keinen grossen Einfluss auf die Resultate haben (siehe Details dazu im Anhang 1). Dies beweist, dass 1000 Simulationen beim Bootstrapping ausreichen, wobei 2000 und 3000 Simulationen keinen grossen Mehrwert für die Resultate generieren. Ausserdem benötigt die Berechnung mit 2000 oder 3000 Simulationen mehr Rechenleistung des Computers und dauert daher deutlich länger. Um alle Kombinationen für ein Konfidenzniveau zu rechnen, welche in Tabelle 2 aufgeführt sind, benötigt der Computer circa zwölf Stunden. Im Anhang 2 wird zudem aufgezeigt, dass ein grösseres Bootstrap-Sample nicht unbedingt mehr Simulationen benötigt.

Für weitere Arbeiten könnte die CBB-Methode mit dem Filtered Historical Bootstrap nach Barone-Adesi et al. (1999) verglichen werden. Der Filtered Historical Bootstrap ist ein semi-parametrischer Ansatz, welcher versucht, die Volatilität besser zu erfassen, indem GARCH-Modelle (Generalized Autoregressive Conditional Heteroscedasticity) eingesetzt werden (Romero et al., 2013, S. 29). Eine weitere Möglichkeit für zukünftige Arbeiten wäre, herauszufinden, ob eine unterschiedliche Gewichtung der Renditen im Sample eine Verbesserung der Schätzung des VaRs herbeiführt. Dabei könnten die neueren Renditen höher gewichtet werden als die älteren. Dies würde dazu führen, dass im Bootstrapping die Wahrscheinlichkeit des Ziehens angepasst wird und die aktuelleren Renditen häufiger gezogen werden als die älteren. Auch interessant wäre die Untersuchung des Circular Block Bootstraps, bei dem die Renditen zuerst mit der aktuellen Volatilität aktualisiert werden (siehe dazu Hull und White, 1998). Dies würde sich in Bezug auf die Genauigkeit der Berechnung des VaRs als gewinnbringen erweisen.

# Literaturverzeichnis


Barone-Adesi, G., Giannopoulos, K., & Vosper, L. (1999). VaR without correlations for portfolios of derivative securities. *Journal of Futures Markets*, 19(5), S. 583-602.

Basel Committee on Banking Supervision (1996). *Überblick über die Änderungen der Eigenkapitalvereinbarung zu Einbeziehung der Marktrisiken*. Abgerufen von https://www.bis.org/publ/bcbs23de.

Beran, R., & Ducharme, G. R. (1991). *Asymptotic theory for bootstrap methods in statistics*. Montréal: Les Publications CRM.

Boos, D. (2003). Introduction to the Bootstrap World, *Institute of Mathematical Statistics*, 18(2), S. 168-174.

Burns, P. (2002a). *The Quality of Value at Risk Via Univariate GARCH*. Abgerufen von SSRN: https://ssrn.com/abstract=443540.

Burns, P. (2002b). *Robustness of the Ljung-Box Test and its Rank Equivalent*. Abgerufen von SSRN: https://ssrn.com/abstract=443560.

Carlstein, E. (1986). The use of subseries methods for estimating the variance of a general statistic from a stationary time series. *Annual Statistics,* 14, S. 1171-1179.

Crouhy, M., Galai, D., & Mark, R. (2014). *The essentials of risk management*, Second Edition. New York: McGraw-Hill.

Davison, A. C., & Hinkley, D. V. (1997). *Bootstrap methods and their application.* Cambridge: Cambridge university press.

Dixon, P. M. (2006). Bootstrap resampling. *Encyclopedia of environmetrics*, Chichester: John Wiley & Sons.

Dowd, K. (2007). *Measuring market risk*. Second Edition. Chichester: John Wiley & Sons.

Dutta, D., & Bhattacharya, B. (2008). A Bootstrapped Historical Simulation Value at Risk Approach to S&P CNX Nifty. *In The National Conference on Money and Banking*, IGIDR, Mumbai.

Efron, B. (1979). "Bootstrap methods: Another look at the jackknife". *The Annals of Statistics*. 7 (1), S. 1-26.

Hall, P. (1985). Resampling a coverage pattern. *Stochastic processes and their applications.* 20(2), S. 231-246.

Hall, P. (1992). *The Bootstrap and Edgeworth Expansion*. First Edition. New York: Springer-Verlag.

Härdle, W., Horowitz, J., & Kreiss, J. P. (2003). Bootstrap methods for time series. *International Statistical Review*, 71(2), S. 435-459.

Hendricks, D., (1996). Evaluation of Value-at-Risk Models Using Historical Data. *Federal Reserve Bank of New York Economic Police Review*, 2(1), S. 39-70.

Holton, G. A. (2002). *History of Value-at-Risk: 1922-1998*. Working Paper. Boston: Contingency Analysis.

Hull, J., & White, A. (1998). Incorporating volatility updating into the historical simulation method for value-at-risk. *Journal of risk,* 1(1), S. 5-19.

Hull, J. (2012). *Risk management and financial institutions*. Third Edition. New Jersey: John Wiley & Sons.

Jeong, J., & Chung, S. (2001). Bootstrap tests for autocorrelation. *Computational statistics & data analysis*, 38(1), S. 49-69.

Jorion, P. (2010). *Financial Risk Manager Handbook: FRM Part I/Part II*. Sixth Edition. New Jersey: John Wiley & Sons.

Künsch, H. R. (1989). The jackknife and the bootstrap for general stationary observations. *The annals of Statistics*, S. 1217-1241.

Lahiri, S. N. (1999). Theoretical comparisons of block bootstrap methods. *Annals of Statistics*, S. 386-404.

Liu, R. Y., & Singh, K. (1992). Moving blocks jackknife and bootstrap capture weak dependence. *Exploring the limits of bootstrap*, S. 225-248.

Ljung, G. M., & Box, G. E. (1978). On a measure of lack of fit in time series models. *Biometrika*, 65(2), S. 297-303.

MacKinnon, J. G. (2006). Bootstrap methods in econometrics. *Economic Record*, 82, S. 2-18.

McKinsey & Company (2012). *Managing market risk: Today and Tomorrow.* Working Papers on Risk Nr. 32. New York: McKinsey & Company.

Mehmke, F., Cremers, H., & Packham, N. (2012). *Validierung von Konzepten zur Messung des Marktrisikos: Insbesondere des Value at Risk und des Expected Shortfall.* Working Paper Nr. 192. Frankfurt: School-Working Paper Series.

Mogull, R. G. (1994). Teacher's Corner: The One-Sample Runs Test: A Category of Exception. *Journal of Educational Statistics*, 19(3), S. 296-303.

Politis, D. N., & Romano, J. P. (1992). A circular block-resampling procedure for stationary data. *Exploring the limits of bootstrap*, 2635270.

Politis, D. N., & Romano, J. P. (1994). The stationary bootstrap. *Journal of the American Statistical association*, 89(428), S. 1303-1313.

Rjiba, M., Tsagris, M., & Mhalla, H. (2015). Bootstrap for Value at Risk Prediction. *International Journal of Empirical Finance*, 4(6), S. 362-371.

Romero, P. A., Muela, S. B., & Martin, C. L. (2013). A comprehensive review of value at risk methodologies. *Documentos de Trabajo FUNCAS*, (711).

Tibshirani, R. J., & Efron, B. (1993). An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57, S. 1-436.

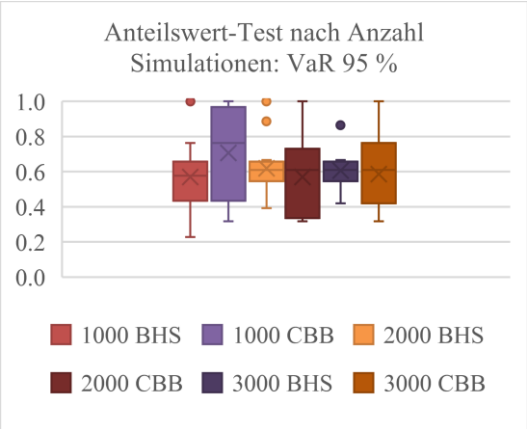
Wald, A., & Wolfowitz, J. (1943). An exact test for randomness in the non-parametric case based on serial correlation. *The Annuals of Mathematical Statistics*, 14(4), S. 378-388.

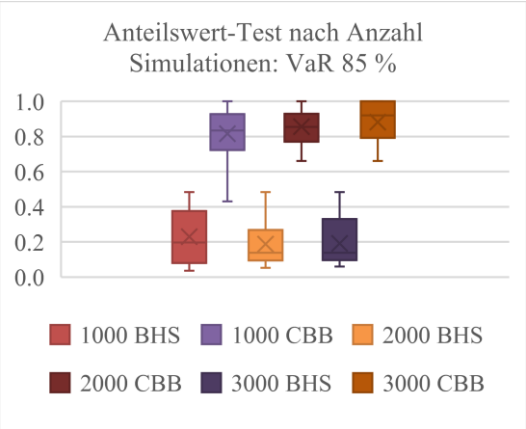
Whitehead, C. K. (2010). Destructive coordination. *Cornell Law Faculty Publications.* Paper Nr. 183, S. 96-323.

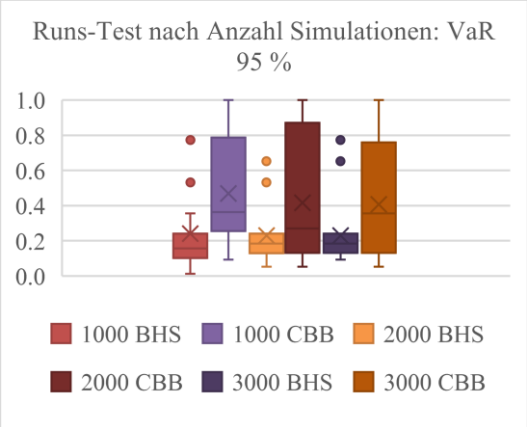
Zivot, E., & Wang, J. (2001). *Modelling Financial Time Series with S-PLUS*. Unpublished Working Paper.

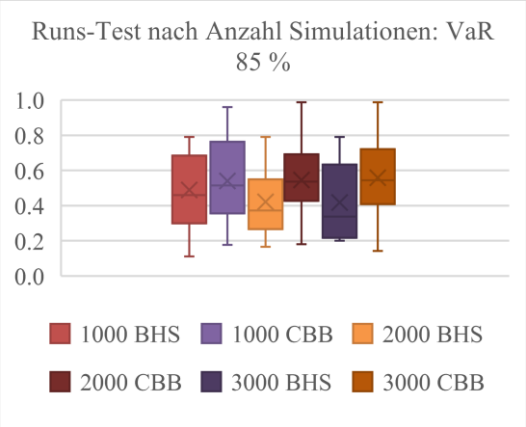
# Anhang 1: Resultate bezüglich der Anzahl Simulationen

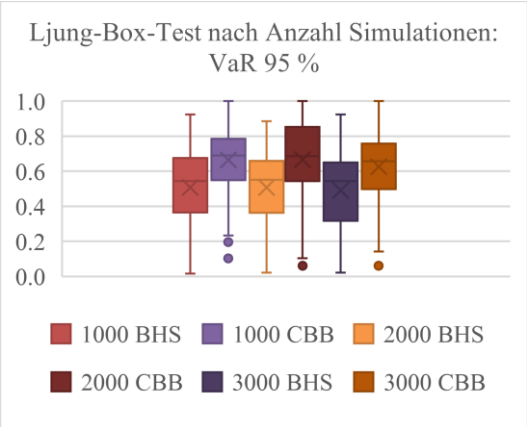
Die folgenden Abbildungen zeigen, dass die Anzahl Simulationen in dieser Grössenordnung keinen grossen Einfluss auf die Resultate haben.. Das Berechnen des VaRs mit 2000 oder 3000 Simulationen setzt die Rechenleistung des Computers suboptimal ein. Somit genügen 1000 Simulationen.

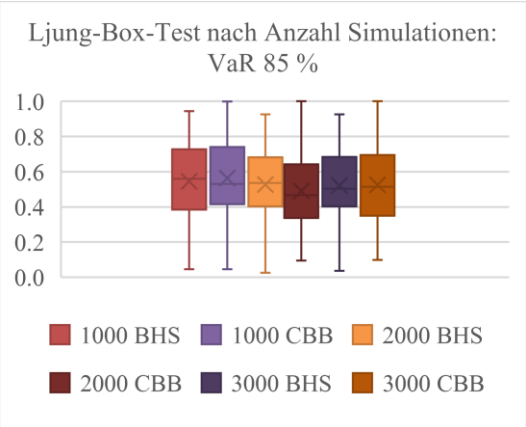


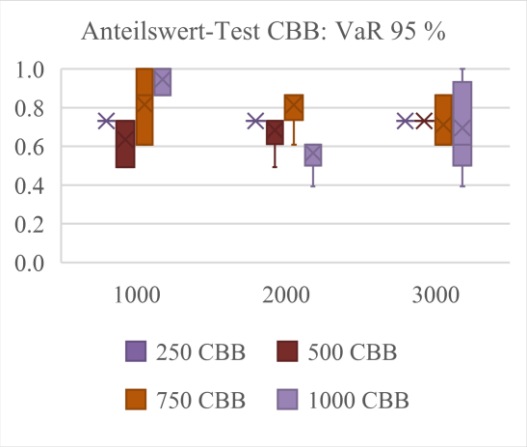



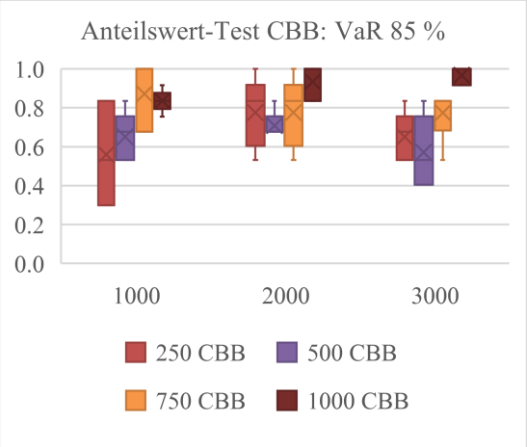



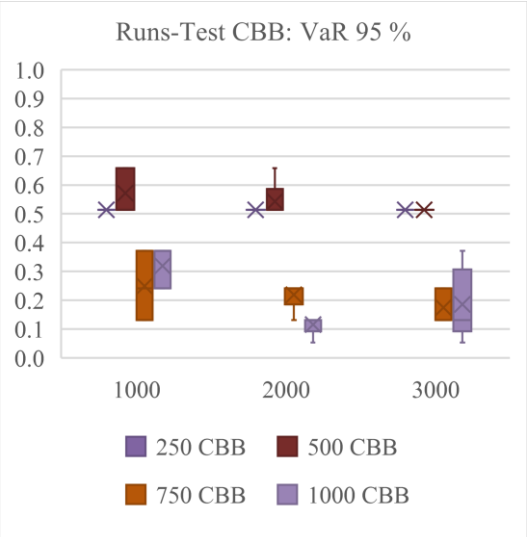
# Anhang 2: Resultate der CBB-Methode

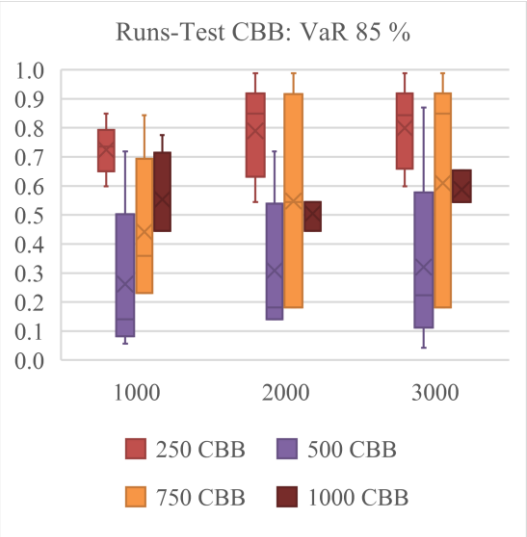
In diesem Abschnitt wird die CBB-Methode genauer analysiert, um herauszufinden, welche Sample-Grösse oder Anzahl Simulationen genauere Ergebnisse erzielt. Gleichzeitig wird untersucht, ob sich die Resultate, gruppiert nach Anzahl Simulationen, mit den Resultaten der Sample-Grössen verändern. Unklar ist, ob eine 1000-fache Ziehung aus einem Sample von 250 Tagen andere Resultate generiert als die 1000-fache Ziehung aus einem Sample von 1000 Tagen. Bei diesen Zahlen wird die gleiche Adjustierung durchgeführt, welche auch für die verschiedenen Sample-Grössen verwendet wird.



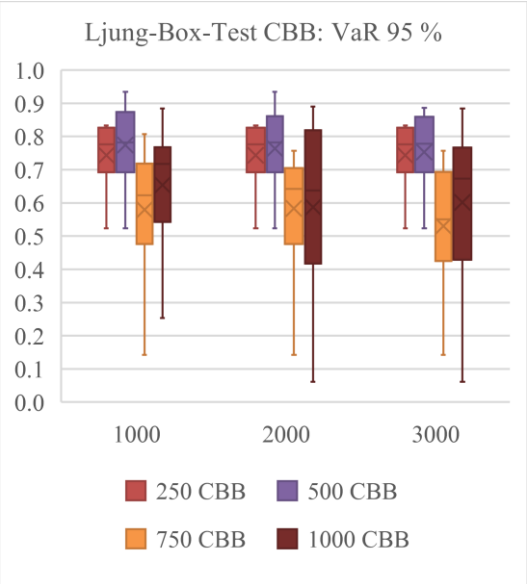



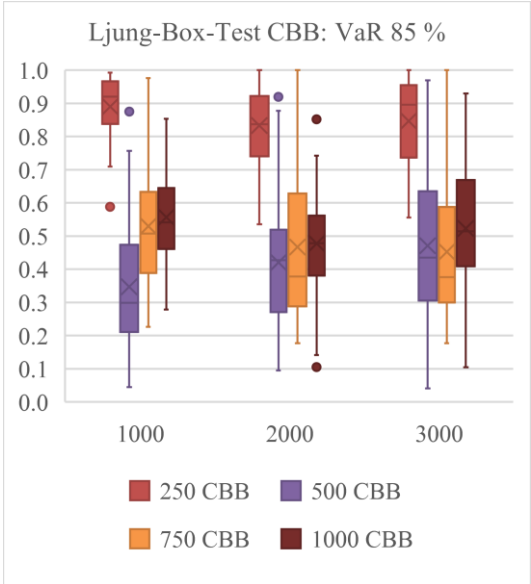

Die Resultate des Anteilswert-Tests zeigen, dass die Anzahl Simulationen bei den kleineren Sample-Grössen keine starken Abweichungen aufweisen. Mit einem grösseren Sample verändern sich die P-Werte, wobei kein richtiges Muster erkennbar ist. Auch hier ist ersichtlich, dass die Sample-Grössen von 750 und 1000 Tagen bei der CBB-Methode über beide Konfidenzniveaus hinweg besser abschneiden und dass 1000 Simulationen ausreichen, um eine gute Schätzung zu erhalten.



Auch die P-Werte des Runs-Tests zeigen, dass die Anzahl Simulationen keinen starken Einfluss auf die Resultate hat. Hier ist ersichtlich, dass die beiden kleineren Sample-Grössen höhere P-Werte liefern als die Sample-Grössen von 750 und 1000 Tagen.




Auch beim Ljung-Box-Test hat die Anzahl Simulationen keinen signifikanten Einfluss in Bezug auf die Sample-Grösse. Dies bedeutet, dass in diesem Fall ein grösseres Sample keine grössere Anzahl Simulationen benötigt.

Die Resultate zeigen, dass eine 1000-fache Ziehung aus einem Sample von 250 Tagen ähnliche Resultate generiert wie bei der 1000-fachen Ziehung aus einem Sample von 1000 Tagen.

Beim Ljung-Box-Test ist ersichtlich, dass die Sample-Grösse von 250 Tagen die höchsten P-Werte aufweist. Die Sample-Grössen von 750 und 1000 Tagen zeigen eher tiefe P-Werte auf, werden jedoch nicht verworfen.

# Anhang 3: Python-Code

Die gelben Markierungen im Code stellen die Adjustierung dar, welche für die Auswertung nach Sample-Grösse zusätzlich angewendet werden.

### Python-Code: Import der Daten

```
%autosave 0
import numpy as np
import math as Math
import pandas as pd
import matplotlib.pyplot as plt
import random
import pylab
import statistics
import statsmodels.api as sm
from statsmodels.sandbox.stats.runs import runstest_1samp
from scipy.stats.distributions import binom
from scipy import stats as stats
%matplotlib inline

#---------------------------------------------------------------------
#File mit täglichen Kursen einlesen und Log-Renditen berechnen
#---------------------------------------------------------------------
Kurse = pd.read_csv(r'C:\Users\lione\OneDrive\ZHAW\Bachelor-Arbeit\Kurse.csv',
                        sep=';',
                        index_col=0,
                        parse_dates=True).sort_index()
Returns = np.log(Kurse/Kurse.shift(1)).dropna()

#---------------------------------------------------------------------
#Portfolio-Renditen für Grafiken und Statistiken berechnen:
#Log-Renditen der einzelnen Aktien in diskrete Renditen umrechnen
#Durschnitt der diskreten Renditen ergeben die Portfolio-Renditen
#---------------------------------------------------------------------
PF_Returns =  Math.e**(Returns)-1
PF_Returns = PF_Returns.mean(axis=1)
```

### Python-Code: Bootstrapping

```
#---------------------------------------------------------------------
#Parameter definieren:
#---------------------------------------------------------------------
Anzahl_Simulationen = 1000 #Hier wurde 1000, 2000, 3000 Simulationen getestet
Sample_length = 750 #Länge des Samples für Bootstrapping: 250, 500, 750, 1000
alpha = 0.05 #Konfidenzniveau definieren 0.05 und 0.15
Position_VaR = int(Anzahl_Simulationen * alpha)
#Da unterschiedliche Anzahl Simulationen, dynamisch 5%-Position definieren
Anzahl_Wiederholungen = 5
#5 Mal durchführen, um Konsistenz der Berechnung zu überprüfen


#---------------------------------------------------------------------
#Sample-Grösse definieren:
#Automatische Adjustierung der Sample-Grösse:
#---------------------------------------------------------------------
Returns2 = Returns.reset_index()
Sample for Bootstrap = Returns2[['Novartis', 'LafargeHolcim',
                                 'Logitech', 'Nestle', 'Swiss_Re']]

sample_length_adjustierung = (round((len(Sample_for_Bootstrap)-Sam-
ple_length)/(10)))*10 + Sample_length
Returns_for_sample = Returns[:sample_length_adjustierung].reset_index()
```

```
Sample_for_Bootstrap = Returns_for_sample[['Novartis', 'LafargeHolcim',
                                           'Logitech', 'Nestle', 'Swiss_Re']]

#---------------------------------------------------------------------
#Loop 1: Alles 5 Mal ausführen und als Excel und Grafik speichern:
#---------------------------------------------------------------------
for y in range(0,Anzahl_Wiederholungen):
    BHS = [] #Tabelle mit allen berechneten VaRs der BHS-Methode
    CBB = [] #Tabelle mit allen berechneten VaRs der CBB-Methode
    Reality = [] #Tabelle für die jeweiligen 10 Tage,
                 #welche für das Backtesting benötigt werden

    #Dynamisch die Dateinamen für das Excel und die Grafik generieren:
    excel_name = "Simulationen/VaR_Simulation_"+str(alpha)
                 +"_Sim_"+str(Anzahl_Simulationen)
                 +"_length_"+str(Sample_length)+"_V"+str(y+1)+ ".xlsx"
    Bild_name = "Simulationen/VaR_Simulation_"+str(alpha)
                 +"_Sim_"+str(Anzahl_Simulationen)
                 +"_length_"+str(Sample_length)+"_V"+str(y+1)+".png"

    #---------------------------------------------------------------------
    #Loop 2: Rolling-Window-Methode:
    #750 Tage für die Durchführung des Bootstrappings & 10 Tage für Backtesting:
    #---------------------------------------------------------------------
    for x in range(0, len(Sample_for_Bootstrap)-Sample_length, 10):
        Samp = Sample_for_Bootstrap[x:x+Sample_length]

        #Die nächsten 10 Tage für das Backtesting wählen,
        #in diskrete Portfolio-Renditen umrechnen und speichern:
        T = Sample_for_Bootstrap.iloc[x+Sample_length:x+Sample_length+10]
        Real_Sample = T.sum()
        diskrete_real = Math.e**(Real_Sample)-1
        diskrete_real = diskrete_real.mean()
        Reality.append(diskrete_real)

        #---------------------------------------------------------------------
        #Loop 3:
            #BHS & CBB mit n Simulationen & zeitliche Addition der Log-Returns
            #umrechnen, in diskrete Returns und Berechnung des Mittelwerts
            #für die Portfolio-Rendite:
        #---------------------------------------------------------------------

        #---------------------------------------------------------------------
        #Bootstrap Historical Simulation (BHS):
        #10 Tage nach Zufall mit Zurücklegen ziehen und 1000 Mal wiederholen:
        #---------------------------------------------------------------------
        BHS_VaR = []
        for x in range(0,Anzahl_Simulationen):
            Stichproben = Samp.sample(n=10, replace=True).sum()
            df2 = Math.e**(Stichproben)-1
            x = df2.mean()
            BHS_VaR.append([x])

        #---------------------------------------------------------------------
        #Circular Block Bootstrap (CBB):
        #10-Tages-Blöcke mit Circular-Effekt ziehen und 1000 Mal wiederholen:
        #---------------------------------------------------------------------
        CBB_VaR = []
        for x in range(0,Anzahl_Simulationen):
            Random_X_from = np.random.randint(0,Sample_length+1)
            Random_X_to = Random_X_from + 10
            Stichproben_CBB = Samp[Random_X_from:Random_X_to]

            #---------------------------------------------------------------
            #Circular-Effekt:
            #Das Ende des Samples mit dem Anfang verbinden
            #---------------------------------------------------------------
            #Überprüfen, ob Block über den Rand geht.
            if Random_X_to > Sample_length:
                Random_X_to_Circle = (Random_X_to-Sample_length)
                #Anzahl fehlende Tage berechnen
                Rest_Circular = Samp[:Random_X_to_Circle]
                #Fehlende Tage vom Anfang des Samples holen
                Stichproben_CBB = pd.concat([Stichproben_CBB, Rest_Circular])
                #Beide Listen zusammenfügen

              Stichproben_CBB.sum()
```

```
100.                Stichproben_CBB = Stichproben_CBB.sum()
101.                df_CBB = Math.e**(Stichproben_CBB)-1
102.                #Log-Renditen in diskrete Renditen umrechnen
103.                x_CBB = df_CBB.mean()    #Portfolio-Rendite rechnen
104.                CBB_VaR.append([x_CBB])    #Tabelle mit 1000 Portfolio-Renditen
105.
106.            #-----------------------------------------------------------------
107.            #Alle 1000 simulierten Portfolio-Renditen aufsteigend sortieren
108.            #und die 51 schlechteste Rendite als VaR definieren:
109.            #-----------------------------------------------------------------
110.            CBB_VaR.sort()
111.            PF_Returns_Simulation_CBB = pd.DataFrame(CBB_VaR)
112.            VaR_CBB = PF_Returns_Simulation_CBB.iloc[Position_VaR]
113.            CBB.append(VaR_CBB)
114.
115.            BHS_VaR.sort()
116.            PF_Returns_Simulation_BHS = pd.DataFrame(BHS_VaR)
117.            a = PF_Returns_Simulation_BHS.iloc[Position_VaR]
118.            BHS.append(a)
119.
120.        #---------------------------------------------------------------------
121.        #Die berechneten VaRs der beiden Methoden in neuer Tabelle speichern
122.        #und die tatsächliche 10-Tages-Rendite ergänzen:
123.        #---------------------------------------------------------------------
124.        Simulationen = pd.DataFrame(BHS)
125.        Simulationen = Simulationen.rename(columns={0:'BHS_VaR'})
126.        Simulationen['CBB_VaR'] = pd.DataFrame(CBB)
127.        Simulationen = Simulationen.reset_index()
128.        Simulationen.drop('index', axis=1, inplace=True)
129.        Simulationen = pd.concat([Simulationen, pd.DataFrame(Reality)], axis=1)
130.        Simulationen = Simulationen.rename(columns={0:'Reality'})
131.
132.        #---------------------------------------------------------------------
133.        #Grafik speichern:
134.        #---------------------------------------------------------------------
135.        Simulationen.plot(figsize=(30,15));
136.        plt.savefig(Bild_name, dpi=400)
137.
138.        #---------------------------------------------------------------------
139.        #Anzahl Exceptions und Exceptions Rate berechnen:
140.        #---------------------------------------------------------------------
141.        Simulationen['BHS_Exceptions'] = Simulationen.BHS_VaR > Simulationen.Rea-
    lity
142.        Simulationen['Anzahl_BHS_Exceptions'] = (Simulationen.BHS_VaR > Simula-
    tionen['Reality']).sum()
143.        Simulationen['BHS_Exception_Rate'] = Simulationen.Anzahl_BHS_Excepti-
    ons/(len(Simulationen))
144.
145.        Simulationen['CBB_Exceptions'] = Simulationen.CBB_VaR > Simulationen.Rea-
    lity
146.        Simulationen['Anzahl_CBB_Exceptions'] = (Simulationen.CBB_VaR > Simula-
    tionen['Reality']).sum()
147.        Simulationen['CBB_Exception_Rate'] = Simulationen.Anzahl_CBB_Excepti-
    ons/(len(Simulationen))
148.
149.        Simulationen.to_excel(excel_name)
150.        Simulationen
```

## Python-Code: Anteilswert-Test

```
binom_95_BHS = []
binom_95_CBB = []
binom_85_BHS = []
binom_85_CBB = []

#-------------------------------------------------------------------------------
#Die beiden generierten DataFrames mit allen Resultaten importieren
#Anteilswert-Test durchführen und P-Werte speichern
#-------------------------------------------------------------------------------
for y in range(0,len(Results_95)):
    ErfolgsWK = 0.05  #Beim VaR 95% ist die Erfolgswahrscheinlichkeit = 5%
    x_BHS = Results_95.iloc[y].BHS_Exceptions
    x_CBB = Results_95.iloc[y].CBB_Exceptions
    n = Results_95.iloc[y].Anzahl_Tests
    b_95_BHS = stats.binom_test(x_BHS, n, ErfolgsWK, alternative='two-sided')
    b_95_CBB = stats.binom_test(x_CBB, n, ErfolgsWK, alternative='two-sided')
    print(x_BHS)
    print(n)
    print(b_95_BHS)
    binom_95_BHS.append(b_95_BHS)
    binom_95_CBB.append(b_95_CBB)

for y in range(0,len(Results_85)):
    ErfolgsWK = 0.15  #Beim VaR 85% ist die Erfolgswahrscheinlichkeit = 15%
    x_BHS = Results_85.iloc[y].BHS_Exceptions
    x_CBB = Results_85.iloc[y].CBB_Exceptions
    n = Results_85.iloc[y].Anzahl_Tests
    b_85_BHS = stats.binom_test(x_BHS, n, ErfolgsWK, alternative='two-sided')
    b_85_CBB = stats.binom_test(x_CBB, n, ErfolgsWK, alternative='two-sided')

    binom_85_BHS.append(b_85_BHS)
    binom_85_CBB.append(b_85_CBB)

binom_95_BHS = pd.DataFrame(binom_95_BHS)
binom_95_BHS = binom_95_BHS.rename(columns={0:"BHS_95"})
binom_95_CBB = pd.DataFrame(binom_95_CBB)
binom_95_CBB = binom_95_CBB.rename(columns={0:"CBB_95"})
binom_85_BHS = pd.DataFrame(binom_85_BHS)
binom_85_BHS = binom_85_BHS.rename(columns={0:"BHS_85"})
binom_85_CBB = pd.DataFrame(binom_85_CBB)
binom_85_CBB = binom_85_CBB.rename(columns={0:"CBB_85"})

Binomial_results = pd.concat([binom_95_BHS,binom_95_CBB], axis=1)
Binomial_results = pd.concat([Binomial_results,binom_85_BHS], axis=1)
Binomial_results = pd.concat([Binomial_results,binom_85_CBB], axis=1)
Binomial_results.to_excel('Binomial_results.xlsx')
Binomial_results
```

## Python-Code: Runs-Test

```
#-------------------------------------------------------------------------------
#Abgelegte Excels für das Backtesting einlesen:
#-------------------------------------------------------------------------------
Sim = [1000, 2000, 3000]
LL = [250, 500, 750, 1000]
Anzahl_Wiederholungen = 5
path = r'C:\Users\lione\OneDrive\ZHAW\Bachelor-Arbeit\Simulationen'


#-------------------------------------------------------------------------------
#Parameter eingeben:
#-------------------------------------------------------------------------------
Hypo_alpha = 0.05  #Konfidenzniveau für Hypothesen-Test eingeben

#-------------------------------------------------------------------------------
#Abgelegte Excels für das Backtesting mittels Schlaufen einlesen:
#-------------------------------------------------------------------------------
```

```
Results = pd.DataFrame()

for S in Sim:
    for L in LL:
        for y in range(0,Anzahl_Wiederholungen):

            #-----------------------------------------------------------------
            #Alle Excel-Files importieren und als DataFrames speichern:
            #-----------------------------------------------------------------
            excel_name = "\VaR_Simulation_Sim_"+str(S)+"_length_"+str(L)+"_V"+str(y+1)+".xlsx" #VaR 95%
            import_name = path + excel_name
            name = "S_"+str(S)+"_L_"+str(L)+"_V"+str(y+1)
            if L == 1000:
                correction = 0
            elif L == 750:
                correction = 25
            elif L == 500:
                correction = 50
            elif L == 250:
                correction = 75

            vars()[name] = pd.read_excel(import_name, sep=';',
                                         index_col=0).sort_index()
            vars()[name] = vars()[name][correction:].reset_index()

            #-----------------------------------------------------------------
            #Runs-Test:
            #Für jedes File den Runs-Test durchführen
            #und den Z-Score und den P-Wert speichern:
            #-----------------------------------------------------------------
            vars()[name]['RT_Z_Score_BHS'] = runstest_1samp(vars()[name].BHS_Exceptions, correction=False)[0]
            vars()[name]['RT_P_Value_BHS'] = runstest_1samp(vars()[name].BHS_Exceptions, correction=False)[1]
            vars()[name]['RT_reject_H0_BHS'] = vars()[name]['RT_P_Value_BHS'] <= Hypo_alpha
            vars()[name].Anzahl_BHS_Exceptions = vars()[name].BHS_Exceptions.sum()

            vars()[name]['RT_Z_Score_CBB'] = runstest_1samp(vars()[name].CBB_Exceptions, correction=False)[0]
            vars()[name]['RT_P_Value_CBB'] = runstest_1samp(vars()[name].CBB_Exceptions, correction=False)[1]
            vars()[name]['RT_reject_H0_CBB'] = vars()[name]['RT_P_Value_CBB'] <= Hypo_alpha
            vars()[name].Anzahl_CBB_Exceptions = vars()[name].CBB_Exceptions.sum()
            vars()[name]['Run'] = name
            vars()[name]['Anzahl_Tests'] = vars()[name]['Run'].count()

            #-----------------------------------------------------------------
            #Erste Zeile jedes DataFrames wählen
            #und in eine neue Tabelle mit allen Resultaten speichern:
            #-----------------------------------------------------------------
            First_col = pd.DataFrame(vars()[name].iloc[0])
            First_col = First_col.T
            Results = pd.concat([Results, First_col])

#---------------------------------------------------------------------------
#Tabelle mit den Resultaten optimieren und als Excel exportieren:
#---------------------------------------------------------------------------
Results = Results.set_index('Run')
Results = Results.drop(['BHS_VaR','CBB_VaR','Reality','BHS_Exceptions', 'CBB_Exceptions'], axis=1)
Results = Results.rename(columns=
                         {'Anzahl_BHS_Exceptions' : 'BHS_Exceptions'
                         ,'Anzahl_CBB_Exceptions' : 'CBB_Exceptions'})
Results.BHS_Exception_Rate =  Results.BHS_Exceptions / Results.Anzahl_Tests
Results.CBB_Exception_Rate =  Results.CBB_Exceptions / Results.Anzahl_Tests
Results.to_excel('Results_95.xlsx')
Results_95 = Results
Results_95
```

## Python-Code: Ljung-Box-Test

```
#----------------------------------------------------------------------
#Abgelegte Excels für das Backtesting einlesen:
#----------------------------------------------------------------------
Sim = [1000, 2000, 3000]
LL = [250, 500, 750, 1000]
Anzahl_Wiederholungen = 5
path = r'C:\Users\lione\OneDrive\ZHAW\Bachelor-Arbeit\Simulationen'

#----------------------------------------------------------------------
#Parameter eingeben:
#----------------------------------------------------------------------
Hypo_alpha = 0.05  #Konfidenzniveau für Hypothesen-Test
number_of_lags = 10 #Anzahl Lags eingeben

#----------------------------------------------------------------------
#Abgelegte Excels für das Backtesting mittels Schlaufen einlesen:
#----------------------------------------------------------------------
LB_Results = pd.DataFrame()

for S in Sim:
    for L in LL:
        for y in range(0,Anzahl_Wiederholungen):

            #----------------------------------------------------------------------
            #Alle Excel-Files importieren und als DataFrames speichern:
            #----------------------------------------------------------------------

            excel_name = "\VaR_Simula-
tion_Sim_"+str(S)+"_length_"+str(L)+"_V"+str(y+1)+".xlsx"
            import_name = path + excel_name
            name = "S_"+str(S)+"_L_"+str(L)+"_V"+str(y+1)
            if L == 1000:
                correction = 0
            elif L == 750:
                correction = 25
            elif L == 500:
                correction = 50
            elif L == 250:
                correction = 75

            vars()[name] = pd.read_excel(import_name, sep=';',
                                         index_col=0).sort_index()
            vars()[name] = vars()[name][correction:].reset_index()
            #----------------------------------------------------------------------
            #Ljung-Box-Test für jedes File durchführen und Resultate speichern:
            #----------------------------------------------------------------------
            LB_R = pd.DataFrame()
            LB_BHS = sm.stats.acorr_ljungbox(vars()[name].BHS_Exceptions,
lags=number_of_lags)
            LB_BHS = pd.DataFrame(LB_BHS)
            LB_BHS = LB_BHS.T
            LB_BHS = LB_BHS.rename(columns={0: 'Q_BHS', 1:'P_Value_BHS'})
            LB_BHS['LBT_Reject_H0_BHS'] = LB_BHS.P_Value_BHS < Hypo_alpha

            LB_CBB = sm.stats.acorr_ljungbox(vars()[name].CBB_Exceptions,
lags=number_of_lags)
            LB_CBB = pd.DataFrame(LB_CBB)
            LB_CBB = LB_CBB.T
            LB_CBB = LB_CBB.rename(columns={0: 'Q_CBB', 1:'P_Value_CBB'})
            LB_CBB['LBT_Reject_H0_CBB'] = LB_CBB.P_Value_CBB < Hypo_alpha
            #Spalten von BHS und CBB zusammenfügen:
            LB_R = pd.concat([LB_BHS,LB_CBB] , axis=1)
            LB_R['Lag'] = range(1,11)
            LB_R['Run'] = name
            #Resultate jedes Durchganges untereinander auflisten:
            LB_Results = pd.concat([LB_Results, LB_R])

LB_Results = LB_Results.set_index('Run')
LB_Results.to_excel('Ljung_Box_Results_95.xlsx')
LB_Results
```