Einfluss von Abhängigkeitsstrukturen auf Clustering Performance nach Dimensionsreduktion

Bachelorarbeit

Ludwig-Maximilians-Universität München
Fakultät für Mathematik, Informatik und Statistik

Fabian Stermann

betreut von
Prof. Dr. Thomas Augustin,
Cornelia Fütterer

München, den 11.05.2021

Zusammenfassung

Diese Arbeit befasst sich mit dem Einfluss unterschiedlicher Korrelationsszenarien auf die Performance von Clusteringverfahren in Kombination mit Methoden zur Dimensionsreduktion. Hierfür werden hochdimensionale Simulationsdaten generiert die verschiedene Szenarien an Abhängigkeit, Heterogenität und Variablenanzahl repräsentieren. Speziell wird sich hierbei an Einzelzell-RNA-Sequenzierungsdaten orientiert, die anhand einer Zero-Inflated Negativ Binomial Verteilung modelliert werden. Die Ergebnisse dieser Arbeit können einen Hinweis darauf geben, welche Methoden am besten zur Identifikation von Gruppen in diesen speziellen Datensätzen unter vorliegender Korrelationsstruktur geeignet sind.

Inhaltsverzeichnis

1 Einführung

Die Einzelzellanalyse ist eine Methodik in der biologischen Forschung, die erst seit Kurzem die Möglichkeit bietet die Funktionsweise und Zusammensetzung einzelner Zellen zu untersuchen [1]. Diese kann dazu verwendet werden unbekannte oder seltene Zelltypen zu identifizieren, Gene zu beobachten die in bestimmten Zelltypen unterschiedlich ausgedrückt werden oder Gewebestrukturen auf vorhandene Zellen zu untersuchen [2]. Hierbei kann es von Nutzen sein verschiedene Gruppen von Zelltypen zu identifizieren, die beispielsweise im Fall von Behandlung oder Krankheit in den einzelnen Genen unterschiedlich ausgedrückt werden. Speziell in dieser Arbeit soll sich an Daten orientiert werden, die mit einer RNA-Sequenzierung (scRNA-seq) erfasst wurden.

Um Einzelzelldatensätze in Gruppen aufzuteilen werden verschiedene Verfahren verwendet. Zwei dieser Methoden sind die Dimensionsreduktion (DR) und Clusteranalyse (CA) [3]. Die Kombination aus DR und CA ist eine verbreitete Vorgehensweise in der Analyse von hochdimensionalen Datensätzen, da hier die reine Clusteranalyse dem sogenannten Fluch der Dimensionalität unterliegt [4]. Viele distanzbasierte Clusteringverfahren scheitern in hoher Dimension, da sich die Distanzen zunehmend ähnlicher sind. Daher kann es sinnvoll sein, den Ausgangsdatensatz in eine niedrige Dimension zu projizieren und anschließend eine CA durchzuführen.

Die Wahl der richtigen Kombination aus DR und CA hat einen entscheidenden Einfluss auf die Ergebnisse. Eine zentrale Frage, ob die Abhängigkeit zwischen Genen eine Rolle in der Identifikation von Gruppen in Einzelzelldatensätzen spielt, soll Thema dieser Arbeit sein. Eine wünschenswerte Erkenntnis wäre, anhand der vorliegenden Abhängigkeit in den Daten die richtige Wahl der Verfahren treffen zu können. Hierfür soll der Zusammenhang anhand von Simulationsdatensätzen untersucht werden, die verschieden Szenarien an Heterogenität, Abhängigkeit und Variablenanzahl repräsentieren.

In Kapitel 2 sollen zunächst die verwendeten Methoden beschrieben und deren Algorithmen erläutert werden. Hierfür werden in 2.1 drei gängige Methoden zu DR aufgeführt, UMAP, t-SNE und die Hauptkomponentenanalyse. In 2.2 sind der k-Means Algorithmus, eine hierarchische Clusteringmethode und das Modellbasierte Clustering, sowie die Methode zur Evaluierung der Ergebnisse beschrieben.
In Kapitel 3 wird die gewählte Methode zur Simulation der Daten erläutert. Dafür soll zunächst die gewählte Verteilung und Abhängigkeitsstruktur beschrieben werden. Ein kurzer Überblick der Umsetzung in `R` soll die im Laufe der Arbeit aufgetretenen Hindernisse und ihre Lösungen darstellen.

Anschließend werden in Kapitel 4 die Ergebnisse aufgegliedert nach den DR Methoden präsentiert und untereinander verglichen.
Zuletzt wird ein Fazit gezogen und ein Ausblick auf mögliche Weiterführungen dieser Arbeit gegeben.

Angehängt sind in A Algorithmen, die in dieser Arbeit referenziert werden. Im Anhang B befinden sich weitere Abbildungen der Clusteringergebnisse. Anhang C gibt einen Überblick über die Struktur des elektronischen Anhangs.

2 Methoden

Die Kombination aus Dimensionsreduktion und Clusteranalyse ist oft eine gute Methode, um Gruppen in hochdimensionalen Datensätzen zu erkennen. Wie die einzelnen Methoden die in dieser Arbeit verwendet werden funktionieren und welche Kriterien sie berücksichtigen wird im Folgenden behandelt.

2.1 Dimensionsreduktion

Eine Reduktion der Dimension versucht einen p-dimensionalen Datensatz $\boldsymbol{X} = (\boldsymbol{X}_1, \ldots, \boldsymbol{X}_p)$ $= \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ in eine niedrigere Dimension $q < p$ zu überführen, ohne dabei relevante Informationen über die Datenstruktur zu verlieren.
Gerade im Bereich von hochdimensionalen genetischen Daten ist es nicht unüblich, eine Dimensionsreduktion durchzuführen, um Strukturen zu erkennen sowie die Datensätze zu visualisieren und besser verarbeiten zu können. Insgesamt sollen hier drei verschiedene Methoden betrachtet werden. Ziel dieses Abschnitts ist es einen Überblick über die gängigen Methoden zu geben und den Aufbau dieser darzustellen.

In einigen Methoden wird die euklidische Distanz zur Abstandsbestimmung verwendet. Diese ist definiert als

$$d(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}||_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}. \tag{2.1}$$

2.1.1 UMAP

UMAP (Uniform Manifold Approximation and Projection) ist eine erstmals 2018 eingeführte Methode von McInnes et al. [5] zur Dimensionsreduktion. Zwei Vorteile gegenüber bisher üblichen Methoden heben diese hervor: Zum einen wird ein hoher Anteil der globalen Datenstruktur in vergleichbar niedriger Rechenzeit erhalten, zum anderen können auch große Datenmengen leichter verarbeitet werden.

Der Algorithmus besteht grundlegend aus zwei Schritten.

Konstruktion des Graphen
Im ersten Schritt soll ein k-Nachbarn Graph erzeugt werden, in dem jeder Punkt zu seinen k-nächsten Nachbarn verbunden wird. Sei $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ der ursprüngliche Datensatz mit Maß $d : \boldsymbol{X} \times \boldsymbol{X} \to \mathbb{R}_0^+$. Für alle $\boldsymbol{x}_i$ werden die k-nächsten Nachbarn $\{\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_k}\}$

bezüglich des Maßes d gebildet.
Für jedes $\boldsymbol{x}_i$ wird nun

$$\rho_i = min\{d(\boldsymbol{x}_i, \boldsymbol{x}_{i_j}) \mid 1 \leq j \leq k,\ d(\boldsymbol{x}_i, \boldsymbol{x}_{i_j}) > 0\} \tag{2.2}$$

sowie σ_i festgelegt, sodass

$$\sum_{j=1}^{k} \exp\left(\frac{-max(0,\ d(\boldsymbol{x}_i, \boldsymbol{x}_{i_j})) - \rho_i}{\sigma_i}\right) = log_2(k). \tag{2.3}$$

Damit kann ein gewichteter gerichteter Graph $\bar{\boldsymbol{G}} = (\boldsymbol{V}, \boldsymbol{E}, w)$ definiert werden. Das bedeutet, dass jede Kante im Graphen ein bestimmtes Gewicht sowie eine Richtung besitzt. Eine Kante kann also von $\boldsymbol{x}_i \rightarrow \boldsymbol{x}_j$ und/oder $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_j$ existieren. Die Knoten $\boldsymbol{V}$ sind die Punkte $\boldsymbol{X}$, die gerichteten Kanten sind $\boldsymbol{E} = \{(\boldsymbol{x}_i, \boldsymbol{x}_{i_j}) \mid 1 \leq j \leq k, 1 \leq i \leq n\}$ und die Gewichtsfunktion

$$w((\boldsymbol{x}_i, \boldsymbol{x}_{i_j})) = exp\left(\frac{-max(0,\ d(\boldsymbol{x}_i,\ \boldsymbol{x}_{i_j})) - \rho_i}{\sigma_i}\right). \tag{2.4}$$

Hierbei entspricht das Gewicht einer Kante in etwa der Wahrscheinlichkeit, dass diese Kante existiert.
Sei $\boldsymbol{A}$ nun die gewichtete Adjazenzmatrix von $\bar{\boldsymbol{G}}$, die die Verbindungen der Punkte repräsentiert, und

$$\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{A}^{\boldsymbol{T}} - \boldsymbol{A} \circ \boldsymbol{A}^{\boldsymbol{T}} \tag{2.5}$$

wobei $\circ$ das paarweise Produkt zweier Matrizen beschreibt. $\boldsymbol{A}_{ij}$ kann als Wahrscheinlichkeit, dass die gerichtete Kante von $\boldsymbol{x}_i$ nach $\boldsymbol{x}_j$ existiert aufgefasst werden. $\boldsymbol{B}_{ij}$ entspricht dann der Wahrscheinlichkeit, dass es mindestens eine der beiden Kanten ($\boldsymbol{x}_i \rightarrow \boldsymbol{x}_j$ oder $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_j$) gibt. Der Graph $\boldsymbol{G}$ ist folglich ein ungerichteter gewichteter Graph mit Adjazenzmatrix $\boldsymbol{B}$. Dieser sollte die Struktur des hochdimensionalen Datensatzes so gut wie möglich beschreiben.

Graphstruktur in niedriger Dimension
UMAP verwendet eine kräfte-basierte gerichtete Graphstruktur, bei der anziehende Kräfte auf die Kanten und abstoßenden Kräfte auf die Knoten angewendet werden. Indem der

Algorithmus iterativ unterschiedliche Kräfte auf die Kanten und Knoten verteilt, wird die Struktur optimiert.
Die anziehende Kraft zwischen zwei Knoten i und j an Koordinaten $\boldsymbol{y}_i$ und $\boldsymbol{y}_j$ wird durch

$$\frac{-2ab||\boldsymbol{y}_i - \boldsymbol{y}_j||_2^{2(b-1)}}{1 + ||\boldsymbol{y}_i - \boldsymbol{y}_j||_2^2} w((\boldsymbol{x}_i, \boldsymbol{x}_j))(\boldsymbol{y}_i - \boldsymbol{y}_j) \tag{2.6}$$

beschrieben, mit Hyperparametern a und b.
Die abstoßende Kraft ist gegeben durch

$$\frac{2b}{(\epsilon + ||\boldsymbol{y}_i - \boldsymbol{y}_j||_2^2)(1 + a||\boldsymbol{y}_i - \boldsymbol{y}_j||_2^{2b})}(1 - w((\boldsymbol{x}_i, \boldsymbol{x}_j)))(\boldsymbol{y}_i - \boldsymbol{y}_j) \tag{2.7}$$

mit $\epsilon > 0$, um Division durch Null zu vermeiden.

Mithilfe dieser Kräfte wird die Kreuzentropie[6] zwischen dem gewichteten Graphen $\boldsymbol{G}$ und einem Graphen $\boldsymbol{H}$ in niedriger Dimension mit Punkten $\{\boldsymbol{y}_i, \ldots, \boldsymbol{y}_n\}$ minimiert. Ziel ist somit, Punkte $\boldsymbol{y}_i$ zu finden deren gewichteter Graph die Struktur von $\boldsymbol{G}$ annähernd approximiert. Da $\boldsymbol{G}$ die Topologie im hochdimensionalen Raum erfasst, repräsentiert der ähnliche Graph $\boldsymbol{H}$ diese so gut wie möglich in niedriger Dimension.

UMAP ist ursprünglich von den Autoren in `Python` [7] implementiert. Eine Version in `R` ist das Paket `uwot` [8].

2.1.2 t-SNE

Die Methode t-SNE von Van der Maaten et al. [9] ist eine Modifikation des Stochastic Neighbor Embedding, indem eine Student-t-Verteilung statt der ursprünglichen Normalverteilung zur Berechnung verwendet wird.

Zunächst werden die Distanzen im hochdimensionalen Raum in bedingte Wahrscheinlichkeiten $p_{j|i}$ umgewandelt.

$$p_{j|i} = \frac{exp(-||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||\boldsymbol{x}_i - \boldsymbol{x}_k||_2^2/2\sigma_i^2)} \tag{2.8}$$

Diese drücken die Ähnlichkeit von Punkt $\boldsymbol{x}_j$ zu $\boldsymbol{x}_i$ aus. Genauer gesagt ist es die Wahrscheinlichkeit, dass $\boldsymbol{x}_i$ $\boldsymbol{x}_j$ als Nachbarn wählen würde, läge die Wahrscheinlichkeitsdichte in

Form einer Normalverteilung um den Punkt $\boldsymbol{x}_i$. Die paarweisen gemeinsamen Wahrscheinlichkeiten werden dann als $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2n}$ definiert, wodurch Ausreißer ebenfalls berücksichtigt werden sollen.
Die entsprechenden Datenpunkte niederer Dimension werden dann als $\boldsymbol{y}_i, \boldsymbol{y}_j$ und mit einer ähnlichen Wahrscheinlichkeit q_{ij} beschrieben. Im t-distributed SNE werden die q_{ij} mithilfe einer Student t-Verteilung mit einem Freiheitsgrad definiert.

$$q_{ij} = \frac{(1 + ||\boldsymbol{y}_i - \boldsymbol{y}_j||_2^2)^{-1}}{\sum_{k \neq l}(1 + ||\boldsymbol{y}_k - \boldsymbol{y}_l||_2^2)^{-1}} \tag{2.9}$$

Je geringer die Differenz zwischen p_{ij} und q_{ij}, desto besser stellen die Punkte $\boldsymbol{y}_i$ und $\boldsymbol{y}_j$ den hochdimensionalen Datensatz dar. Diese wird mit der Kullback-Leibler Divergenz [10] als Kostenfunktion C minimiert.

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} log\left(\frac{p_{ij}}{q_{ij}}\right) \tag{2.10}$$

Hierbei sind P und Q die gemeinsamen Wahrscheinlichkeitsverteilungen im jeweiligen Raum. P_i sei nun die auf $\boldsymbol{x}_i$ bedingte Wahrscheinlichkeitsverteilungen über alle anderen Datenpunkte. Die Wahl einer geeigneten Varianz σ_i in Gleichung (2.8), welche wiederum P_i bestimmt, wird über die Perplexität gesteuert. Diese ist definiert als

$$Perp(P_i) = 2^{H(P_i)} \quad \text{mit} \quad H(P_i) = -\sum_j p_{j|i} log_2 p_{j|i} \tag{2.11}$$

und kontrolliert die Anzahl an nächsten Nachbarn, die jeder Punkt anzieht. Typischerweise liegt dieser Wert zwischen 5 und 50.

Implementiert ist die Methode im `R`-Paket `Rtsne` als Funktion `tsne()`. Standardmäßig wird im ersten Schritt eine PCA durchgeführt, um den hochdimensionalen Datensatz auf 30-50 Dimensionen zu reduzieren. Um den Algorithmus unabhängig von der Hauptkomponentenanalyse betrachtet zu können, wird dieser Schritt in der Arbeit nicht durchgeführt.

2.1.3 Hauptkomponentenanalyse

Die Hauptkomponentenanalyse (Princial Component Analysis, o.a. PCA) ist eine erstmals 1901 von Karl Pearson [11] eingeführte Methode zur Dimensionsreduktion. Hierbei werden lineare Kombinationen der Variablen, sogenannte Hauptkomponenten gebildet, die jeweils

einen Anteil der Varianz im Datensatz erklären [12]. Die größte Varianz in den Daten wird von der ersten Hauptkomponente erklärt, mit absteigender Varianz in den restlichen Hauptkomponenten. Da die Methode bereits in der Veranstaltung *Multivariate Verfahren* behandelt wurde, wird hier auf eine ausführliche Erläuterung verzichtet.

Gegeben sei der Datensatz $\boldsymbol{X} \in \mathbb{R}^{n \times p}$. $\boldsymbol{X} = \boldsymbol{P}\boldsymbol{\Delta}\boldsymbol{Q}^{\boldsymbol{T}}$ ist die Singulärwertzerlegung [12] von $\boldsymbol{X}$, mit

$\boldsymbol{\Delta} \in \mathbb{R}^{n \times p}$ Diagonalmatrix mit Singulärwerten von $\boldsymbol{X}$

$\boldsymbol{P} \in \mathbb{R}^{n \times n}$ Matrix mit Links-Singulärvektoren

$\boldsymbol{Q} \in \mathbb{R}^{p \times p}$ Matrix mit Rechts-Singulärvektoren.

Dann enthält die Matrix

$$\boldsymbol{F} = \boldsymbol{P}\boldsymbol{\Delta} = \boldsymbol{P}\boldsymbol{\Delta}\boldsymbol{Q}^{\boldsymbol{T}}\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{Q} \tag{2.12}$$

die geordneten Faktorwerte, bzw. die einzelnen Hauptkomponenten in ihren Spalten. Typischerweise werden die ersten beiden Komponenten ausgewählt, die dann im zweidimensionalen Raum dargestellt werden können.

Die PCA ist als Funktion `prcomp()` standardmäßig in `R` enthalten.

2.2 Clustering

Die Clusteranalyse beinhaltet Verfahren zur Identifikation von Gruppen in Datensätzen. Diese können durch unterschiedliche Methoden die auf Distanzen, Verteilungen oder ähnlichen Kriterien beruhen definiert sein. Um die verschiedenen Arten der Methoden zu berücksichtigen, wird jeweils ein partitionierendes, hierarchisches und modellbasiertes Clusteringverfahren betrachtet.

2.2.1 Partitionierendes Clustering

Die partitionierenden Verfahren teilen die Daten in eine vorgegebene Anzahl G an Clustern ein. Eine simple und weit bekannte Methode hierfür ist der k-Means Algorithmus. Es gibt mehrere Umsetzungen und Definitionen, in dieser Arbeit wird die Methode von Hartigan und Wong [13] verwendet. Auch wenn es unterschiedliche Arten des Algorithmus gibt, kann

die Methode generell in drei Schritte aufgeteilt werden [14].

Initialisierung $t = 1$; Wähle G zufällige Mittelwerte $\{\boldsymbol{m}_k^{(t)}\}_{k=1,\ldots,G}$.

Zuordnung Jeder Datenpunkt $\boldsymbol{x}_i$ wird dem Mittelwert $\boldsymbol{m}_k^{(t)}$ zugeteilt, zu dem er die niedrigste quadrierte Distanz d (2.1) aufweist. Daraus ergeben sich die Cluster $\{C_k\}_{k=1,\ldots,G}$.

$$C_k = \{\boldsymbol{x}_i \mid d(\boldsymbol{m}_k^{(t)}, \boldsymbol{x}_i)^2 \leq d(\boldsymbol{m}_j^{(t)}, \boldsymbol{x}_i)^2 \ \forall \ j = 1, \ldots, G\} \tag{2.13}$$

Aktualisierung Berechne die neuen Cluster-Mittelwerte aus den jeweils zugeordneten Beobachtungen, mit $|\cdot|$ als deren Anzahl in einem Cluster.

$$\boldsymbol{m}_k^{(t+1)} = \frac{1}{|C_k|} \sum_{\boldsymbol{x}_i \in C_k} \boldsymbol{x}_i \tag{2.14}$$

Die letzten beiden Schritte werden so oft wiederholt, bis sich die Zuordnung nicht mehr verändert. Ziel ist es, die Summe der quadrierten Distanzen

$$\sum_{k=1}^{G} \sum_{\boldsymbol{x}_i \in C_k} d(\boldsymbol{m}_k, \boldsymbol{x}_i)^2 \tag{2.15}$$

zu minimieren.

Implementiert ist der Algorithmus (von Hartigan und Wong [13]) standardmäßig in `R` [15] als Funktion `kmeans()`. Hier kann zusätzlich die Anzahl `nstart` an zufälligen Initialisierungen definiert werden, was unter Umständen bessere Partitionen liefern kann.

2.2.2 Hierarchisches Clustering

Unter den Begriff hierarchisches Clustering fallen mehrere distanzbasierte Methoden der Clusteranalyse. In den hier betrachteten agglomerativen Verfahren bildet zunächst jede Beobachtung ein einzelnes Cluster. Diese werden dann iterativ zusammengefasst und letztendlich zu einem großen Cluster kombiniert.

Initialisierung Jedes Cluster enthält genau eine Beobachtung $C_i = \{\boldsymbol{x}_i\}$. Die Distanzen der Cluster zueinander werden berechnet.

Zuordnung Die beiden Cluster mit der größten Ähnlichkeit bzw. geringsten Distanz zueinander werden zu einem Cluster zusammengefasst.

Aktualisierung Die Ähnlichkeiten der neuen Cluster müssen jetzt erneut berechnet werden. Seien nun C_i und C_j zwei Cluster die zu einem gemeinsamen Cluster $C_i \cup C_j$ agglomeriert werden sollen, und sei C_k ein weiteres Cluster. Die Ähnlichkeit des neuen Clusters $C_i \cup C_j$ relativ zu C_k ist mit der Lance-Williams Formel [16] und Distanzfunktion D (2.18) definiert als

$$D(C_i \cup C_j, C_k) = \alpha_1 d(C_i, C_k) \ + \ \alpha_2 D(C_j, C_k) \ + \ \beta D(C_i, C_j) \ + \ \gamma |D(C_i, C_k) - D(C_j, C_k)|. \tag{2.16}$$

Für die unterschiedlichen hierarchisches Verfahren definieren die Parameter $\alpha_1, \alpha_2, \beta$ und γ sowie die Distanzfunktion D die Ähnlichkeit und wie die Cluster miteinander verbunden sind. Mögliche Verfahren sind beispielsweise das Single, Average, Complete oder Centroid Linkage.
In dieser Arbeit wird Ward's Kriterium (Ward2) [17] und die euklidische Distanz verwendet. Hierfür seien $I = |C_i|, J = |C_j|, K = |C_k|$. Mit den Parametern

$$\alpha_1 = \frac{I+K}{I+J+K} \qquad \alpha_2 = \frac{J+K}{I+J+K} \qquad \beta = -\frac{K}{I+J+K} \qquad \gamma = 0$$

ergibt sich dann die Lance-Williams Formel zu

$$D(C_i \cup C_j, C_k) = \frac{I+K}{I+J+K} D(C_i, C_k) \ + \ \frac{J+K}{I+J+K} D(C_j, C_k) \ - \ \frac{K}{I+J+K} D(C_i, C_j). \tag{2.17}$$

In Ward2 wird die Distanzfunktion D quadriert und ist hier folglich mit (2.1)

$$D(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} (x_i - y_i)^2. \tag{2.18}$$

Die letzten beiden Schritte werden so oft wiederholt, bis nur noch ein Cluster übrig ist.

In `R` ist das Verfahren mit der Funktion `hclust()` standardmäßig implementiert. Die Ward2 Methode wird mit `method = "ward.D2"` spezifiziert. Damit wir letztendlich zwei Cluster erhalten, wird die Hierarchie mit der Funktion `cutree()` auf `k=2` Cluster reduziert.

2.2.3 Modellbasiertes Clustering

Anders als bei den bereits vorgestellten Verfahren verwendet das modellbasierte Clustering keine Distanzen zur Berechnung der Cluster. Hierbei wird angenommen, dass jedes Cluster eine eigene Verteilung besitzt, meist ist das eine multivariate Normalverteilung mit unbekannten Parametern [18].

Sei $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ der Zufallsvektor. Die Verteilung jeder Beobachtung wird durch eine Mischverteilung mit G Komponenten (Clustern) ausgedrückt [19].

$$f(\boldsymbol{x}_i; \boldsymbol{\Psi}) = \sum_{k=1}^{G} \pi_k f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k) \tag{2.19}$$

mit

$\boldsymbol{\Psi}$ $= \{\pi_1, \ldots, \pi_G, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_G\}$ Parameter der Mischverteilung

$f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k)$ Dichtefunktion der k-ten Komponente mit Parametervektor $\boldsymbol{\theta}_k$

$(\pi_1, ..., \pi_G)$ Gewichte der Mischverteilung mit $\pi_k > 0,\ \sum_{k=1}^{G} \pi_k = 1$

Die Parameter $\boldsymbol{\Psi}$ müssen in den meisten Fällen geschätzt werden. Dies wird durch die Maximierung der log-Likelihood von (2.19) mit dem EM Algorithmus [20] durchgeführt.

$$\ell(\boldsymbol{\Psi}; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) = \sum_{i=1}^{n} log(f(\boldsymbol{x}_i; \boldsymbol{\Psi})) \tag{2.20}$$

Das `R`-Paket `mclust` [19] verwendet ein Gaußsches Mischmodell (GMM), welches für jede Komponente eine multivariate Normalverteilung $f_k(\boldsymbol{x}_i; \boldsymbol{\theta}_k) \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ annimmt. Im Paket können die Merkmale Volumen, Form und Richtung der Cluster jeweils als gleich (`E`) oder variierend (`V`) definiert werden. Wird nichts vorgegeben, wählt die Funktion `Mclust()` anhand des BIC das beste Modell aus.

2.2.4 Evaluation

Um Clustering-Resultate mit den wahren Clustern oder einer anderen Methode zu vergleichen kann der Adjusted Rand Index [21] verwendet werden. Seien hierfür $A = \{A_1, ..., A_r\}$

und $B = \{B_1, ..., B_s\}$ zwei Partitionen des Datensatzes. Dann ist die Kontingenztabelle der Cluster mit $n_{ij} = |A_i \cap B_j|$

	B_1	B_2	$\dots$	B_s	Σ
A_1	n_{11}	n_{12}	$\dots$	n_{1s}	a_1
A_2	n_{21}	n_{22}	$\dots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
A_r	n_{r1}	n_{r2}	$\dots$	n_{rs}	a_r
Σ	b_1	b_2	$\dots$	b_s	n

Tabelle 1: Kontingenztabelle der Übereinstimmung zweier Clustering Partitionen

Der Adjusted Rand Index $ARI \in [0, 1]$ ist definiert als

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left(\sum_i \binom{a_j}{2} \sum_j \binom{b_j}{2}\right) / \binom{n}{2}}{\frac{1}{2}\left(\sum_i \binom{a_j}{2} + \sum_j \binom{b_j}{2}\right) - \left(\sum_i \binom{a_j}{2} \sum_j \binom{b_j}{2}\right) / \binom{n}{2}} \tag{2.21}$$

wobei 0 für sich in allen Beobachtungen unterscheidende Cluster und 1 für eine exakte Übereinstimmung der Cluster steht. Das `R`-Paket `Mclust` [19] beinhaltet die Funktion `adjustedRandIndex()`.

3 Simulation der Daten

Um verschiedene Abhängigkeitsstrukturen in den Daten untersuchen und vergleichen zu können, müssen mehrere Datensätze simuliert werden. Dabei wird sich an einem realen Datensatz orientiert, aus dem die Verteilungen der Gene geschätzt werden. In den folgenden Abschnitten sollen zunächst die Verteilungen definiert, die Wahl der Abhängigkeitsstruktur, die Methode zur Simulation und die Umsetzung in `R` erläutert werden.

3.1 Zero-Inflated Negativ Binomial Verteilung

Für die Modellierung von Gendaten sind mehrere diskrete Verteilungen möglich. Wagner et al. [22] schlagen eine Negativ Binomial Verteilung mit einem zusätzlichen Überschuss an Beobachtungen die den Wert 0 annehmen vor. Diese soll hier zunächst definiert werden.

Im Folgenden sind die Parameter (mit entsprechenden Argumenten der Funktion `dzinegbin()` aus dem `R`-Paket `VGAM` [23] in Klammern):

- μ Erwartungswert der Negativ Binomial Verteilung (`munb`)
- ϕ Dispersionsparameter (`size`)
- π Wahrscheinlichkeit zusätzlicher Nullen (`pstr0`)

Im klassischen Fall wird die Negativ Binomial Verteilung mit der Anzahl n an erfolgreichen Versuchen und der Wahrscheinlichkeit p des Eintretens eines Erfolges definiert. Über eine Mischverteilung aus einer Poissonverteilung mit gammaverteiltem λ kann diese jedoch auch angegeben werden als [24]

$$f_{NB}(x) = f(x\,;\,\mu,\phi) = \frac{\Gamma(x+\phi)}{\Gamma(\phi)x!} \cdot \frac{\mu^x \phi^\phi}{(\mu+\phi)^{x+\phi}}. \tag{3.1}$$

Das erlaubt die Stetigkeit des Parameters ϕ, mit dem eine Überdispersion der Daten modelliert werden kann. $\Gamma(\cdot)$ ist die Gammafunktion.

Die Dichtefunktion der Zero-Inflated Negativ Binomial Verteilung kann dann mit zusätzlicher Wahrscheinlichkeit π für $x = 0$ definiert werden.

$$f_{ZINB}(x) = \begin{cases} \pi + (1-\pi) f_{NB}(x) & x = 0 \\ (1-\pi) f_{NB}(x) & x \in \mathbb{N} \end{cases} \tag{3.2}$$

Um die zwei verschiedenen Gruppen in den Daten zu simulieren, werden zwei Intervalle für jeden Parameter definiert, aus denen pro Variable gezogen wird. Dabei wurden die Parameterintervalle von Fütterer et al. [25] verwendet, die über 7225 Gene aus dem Datensatz von Kolodziejczyk et al. [26] geschätzt wurden. Diese sollen unterschiedliche Ausmaße an Heterogenität mithilfe von drei verschiedenen Szenarien darstellen. Szenario 01 beschreibt dabei eine homogene Situation der Gruppen mit hoher Überlappung der Intervalle, während die Parameter in Szenario 03 aus sehr unterschiedlichen Intervallen gezogen werden.

Szenario	Gruppe	μ	ϕ	π
01	1	[45, 293]	[0.27, 0.47]	$[5.30 \cdot 10^{-7}, 0.01]$
	2	[12, 112]	[0.27, 0.47]	$[4.85 \cdot 10^{-7}, 2.11 \cdot 10^{-5}]$
02	1	[27, 397]	[0.24, 0.55]	$[3.65 \cdot 10^{-7}, 0.04]$
	2	[6, 171]	[0.23, 0.55]	$[3.26 \cdot 10^{-7}, 2.91 \cdot 10^{-2}]$
03	1	[19, 576]	[0.18, 0.78]	$[2.28 \cdot 10^{-7}, 0.08]$
	2	[2, 217]	[0.17, 0.82]	$[2.18 \cdot 10^{-7}, 6.11 \cdot 10^{-2}]$

Tabelle 2: ZINB-Parameter der beiden Gruppen für alle Szenarien

Für beide Gruppen wurde jeweils ein Datensatz mit $n_1, n_2 = 250$ Beobachtungen generiert, welche dann zu einem Datensatz mit $n = 500$ Beobachtungen zusammengefügt worden sind. Für jedes Szenario wurden Datensätze mit $p = 250, 500, 1000, 2000$ Variablen generiert, sodass die Relationen $n > p,\ n = p,\ n < p$ und $n << p$ betrachtet werden können.

3.2 Abhängigkeitsstruktur

Als Maß der Abhängigkeit im Datensatz soll hier der Korrelationskoeffizient verwendet werden. Für zwei Zufallsvariablen $\boldsymbol{X}_i$ und $\boldsymbol{X}_j$ ist der Koeffizient gegeben als

$$Korr(\boldsymbol{X}_i, \boldsymbol{X}_j) = \frac{Cov(\boldsymbol{X}_i, \boldsymbol{X}_j)}{\sqrt{Var(\boldsymbol{X}_i)} \cdot \sqrt{Var(\boldsymbol{X}_j)}} = \frac{\sigma_{X_i,X_j}}{\sigma_{X_i}\sigma_{X_j}} =: \rho_{X_i,X_j} = \rho_{X_j,X_i}. \tag{3.3}$$

Für einen p-dimensionalen Datensatz $\boldsymbol{X} = (\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_p)$ ergibt sich somit die symmetrische und positiv-semidefinite Korrelationsmatrix [27]

$$\boldsymbol{P} = Korr(\mathbf{X}) = \begin{pmatrix} 1 & \rho_{X_1,X_2} & \dots & \rho_{X_1,X_p} \\ \rho_{X_2,X_1} & 1 & \dots & \rho_{X_2,X_p} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{X_p,X_1} & \rho_{X_p,X_2} & \dots & 1 \end{pmatrix} \tag{3.4}$$

Um verschiedene Grade von Abhängigkeiten zu untersuchen, werden die paarweisen Korrelationen aus einem bestimmten Intervall gezogen, die von einer minimalen und maximalen Korrelation ρ_{min} und $\rho_{max},\ \rho_{min} < \rho_{max}$ definiert werden.
Somit gilt für p Zufallsvariablen $\boldsymbol{X}_i, \boldsymbol{X}_j,\ i, j = 1, ..., p,\ i \neq j$

$$\rho_{X_i,X_j} = \rho_{X_j,X_i} \in [\rho_{min}, \rho_{max}]. \tag{3.5}$$

Hierfür werden drei Intervalle betrachtet:

$I_{\rho_1} = [0.0, 0.3]$ Niedrige Abhängigkeit

$I_{\rho_2} = [0.4, 0.6]$ Moderate Abhängigkeit

$I_{\rho_3} = [0.7, 0.9]$ Hohe Abhängigkeit

Um die Korrelationsmatrix $\boldsymbol{P}$ für einen p-dimensionalen Datensatz zu generieren, müssen $q = \sum_{i=1}^{p-1} i$ paarweise Korrelationskoeffizienten aus den entsprechenden Intervallen gezogen werden. Aufgrund der Methodik, die im Folgenden noch erläutert werden soll, bietet es sich an die Korrelationskoeffizienten nicht aus einer Gleichverteilung, sondern aus einer Standardnormalverteilung zu ziehen, und diese dann auf das Intervall $[\rho_{min}, \rho_{max}]$ zu skalieren. Somit haben wir einen Vektor $\boldsymbol{R} = (r_1, r_2, \dots, r_q)$ der Länge q mit $\boldsymbol{R} \sim N(0, 1)$, welcher mit der Funktion

$$s(r_i, a, b) = \frac{(b-a)(r_i - min(\boldsymbol{R}))}{max(\boldsymbol{R}) - min(\boldsymbol{R})} + a \tag{3.6}$$

und $a = \rho_{min}, b = \rho_{max}$ zum Vektor $\widetilde{\boldsymbol{R}} = (\widetilde{r}_1, \widetilde{r}_2, \dots, \widetilde{r}_q)$, skaliert wird. Daraus erhält man eine mögliche Korrelationsmatrix

$$\widetilde{P} = \begin{pmatrix} 1 & \widetilde{r}_1 & \widetilde{r}_2 & \dots & \widetilde{r}_{p-1} \\ \widetilde{r}_1 & 1 & \widetilde{r}_p & \dots & \widetilde{r}_{2p-3} \\ \widetilde{r}_2 & \widetilde{r}_p & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \widetilde{r}_q \\ \widetilde{r}_{p-1} & \widetilde{r}_{2p-3} & \dots & \widetilde{r}_q & 1 \end{pmatrix} \tag{3.7}$$

welche mit dem Algorithmus von Higham [28] (im R-Paket `Matrix` [29] als Funktion `nearPD()` implementiert) zur nächstmöglichen validen Korrelationsmatrix korrigiert wird, falls $\widetilde{P}$ nicht positiv definit ist.

Da nach der Korrektur einige der paarweisen Korrelationen unter Umständen nicht mehr im Intervall $[\rho_{min}, \rho_{max}]$ liegen, wird der Algorithmus 1 angewendet, um mehrere $\widetilde{R}$ zu ziehen. Die Erzeugung einer validen Korrelationsmatrix aus negativen Intervallen hat sich bereits in niedriger Dimension als fast unmöglich dargestellt. Daher wird sich in dieser Arbeit auf die angegebenen positiven Intervalle beschränkt. Folglich wird bei Erwähnung der Begriffe Abhängigkeitsstruktur, Korrelation o.ä. immer vom *positiven* Fall gesprochen.

3.3 NORTA-Methode

Um nun einen Zufallsvektor $\boldsymbol{X}$ zu berechnen, dessen Variablen der spezifizierten ZINB-Verteilung folgen und gleichzeitig die Abhängigkeitsstruktur widerspiegeln, kann sich an einer prinzipiell einfachen Methode über die inverse Verteilungsfunktion bedient werden. Cario und Nelson [30] schlagen die Methode *NORTA* (*NOR*mal *T*o *A*nything) vor, die die Generation eines multivariaten Zufallsvektors mit beliebigen Randverteilungen und vorgegebener Korrelationsmatrix $\boldsymbol{P_X}$ ermöglicht. Der Prozess kann in drei Schritte zusammengefasst werden:

1. Simulation von $\boldsymbol{Z} \sim N(\boldsymbol{0}, \boldsymbol{P_Z})$.
2. Berechnung der Quantile der Standardnormalverteilung
   $\boldsymbol{U} = (\Phi(\boldsymbol{Z}_1), \dots, \Phi(\boldsymbol{Z}_p))$.
3. Berechnung des Zufallsvektors über die inversen Verteilungsfunktionen der Randverteilungen
   $\boldsymbol{X} = (F_{X_1}^{-1}(\boldsymbol{U}_1), \dots, F_{X_p}^{-1}(\boldsymbol{U}_p))$.

Die Herausforderung liegt darin $\boldsymbol{P_Z}$ so zu wählen, dass $Korr(\boldsymbol{X}) \approx \boldsymbol{P_X}$. Yahav und Shmueli [31] untersuchen in ihrem Artikel eine Korrektur der NORTA-Methode, um $\boldsymbol{P_Z}$ für eine

multivariate Poissonverteilung zu approximieren. Barbiero und Ferrari [32] zeigen, dass diese Korrektur für beliebige diskrete Randverteilungen verwendet werden kann.

Die erreichbare Korrelation zweier Zufallsvariablen $\boldsymbol{X}_i$ und $\boldsymbol{X}_j$ liegt nach Whitt [33] nicht im Intervall $[-1, 1]$, sondern zwischen der unteren und oberen Schranke

$$\underline{\rho} = cor(F_{X_i}^{-1}(\boldsymbol{U}),\ F_{X_j}^{-1}(1 - \boldsymbol{U})), \tag{3.8}$$

$$\bar{\rho} = cor(F_{X_i}^{-1}(\boldsymbol{U}),\ F_{X_j}^{-1}(\boldsymbol{U})). \tag{3.9}$$

$\boldsymbol{U}$ ist hierbei ein Zufallsvektor aus einer stetigen Gleichverteilung auf dem Intervall $[0, 1]$. Die spezifizierten paarweisen Korrelationen ρ_X werden am besten durch eine Exponentialfunktion approximiert:

$$\rho_X = a \cdot e^{b \cdot \rho_Z} + c \tag{3.10}$$

mit

$$a = -\frac{\bar{\rho} \cdot \underline{\rho}}{\bar{\rho} + \underline{\rho}} \qquad b = log\left(\frac{\bar{\rho} + a}{a}\right) \qquad c = -a\ ,$$

wodurch sich mit Umformung von (3.10) $\boldsymbol{P_Z}$ berechnen lässt

$$\rho_Z = \frac{log\left(\frac{\rho_X - c}{a}\right)}{b}. \tag{3.11}$$

Letztendlich ergeben sich für die Simulation eines p-dimensionalen Datensatzes mit ZINB-Randverteilungen F_{X_i} $(i = 1, ..., p)$ mit Parametern $\mu_i,\ \phi_i,\ \pi_i$ und inversen Verteilungsfunktionen $F_{X_i}^{-1}$ sowie valider Korrelationsmatrix $\boldsymbol{P_X}$ folgende Schritte:

1. Berechne $\bar{\rho}, \underline{\rho}$ mit (3.8) und (3.9)
2. Berechne $\boldsymbol{P_Z}$ aus (3.11) mit a, b, c
3. Ziehe $\boldsymbol{Z}$ aus $N(\boldsymbol{0}, \boldsymbol{P_Z})$
4. Berechnung der Quantile der Standardnormalverteilung
   $\boldsymbol{U} = (\Phi(\boldsymbol{Z}_1), \ldots, \Phi(\boldsymbol{Z}_p))$

5. Transformation von U über die inversen Verteilungsfunktionen
   $\boldsymbol{X} = (F_{X_1}^{-1}(\boldsymbol{U}_1), \ldots, F_{X_p}^{-1}(\boldsymbol{U}_p))$

Paarweise Korrelationen

In der Umsetzung hat sich gezeigt, dass die Methode die ursprüngliche Korrelation ρ_X umso besser erreicht, je mehr Beobachtungen generiert werden. Da wir in unserem Fall von hochdimensionalen Gendaten ausgehen und eine eher niedrige Anzahl an Beobachtungen simulieren möchten, ist immer mit einer gewissen Abweichung von ρ_X zu rechnen. Damit die finalen Korrelationen dennoch so gut wie möglich in den vorgegebenen Intervallen liegen, wurde die zuvor beschriebene Kombination aus Standardnormalverteilung und Skalierung gewählt, um die paarweisen Korrelationen zu generieren.

Ein weiterer Punkt, der für die Verwendung dieser Kombination spricht, hängt mit der Korrektur zur nächstmöglichen Korrelationsmatrix mit `nearPD()` zusammen. Liegen viele paarweise Korrelationen an den Rändern der Intervalle I_{ρ_i} ist zu erwarten, dass nach der Korrektur des Algorithmus einige der ρ_{ij} nicht mehr im Intervall I_{ρ_i} liegen. Durch die hier verwendete Methode befinden sich nur noch sehr wenige der ρ_{ij} an den Rändern der Intervalle, und das Generieren einer validen Korrelationsmatrix nimmt auch in höheren Dimensionen vergleichbar wenig Zeit in Anspruch.

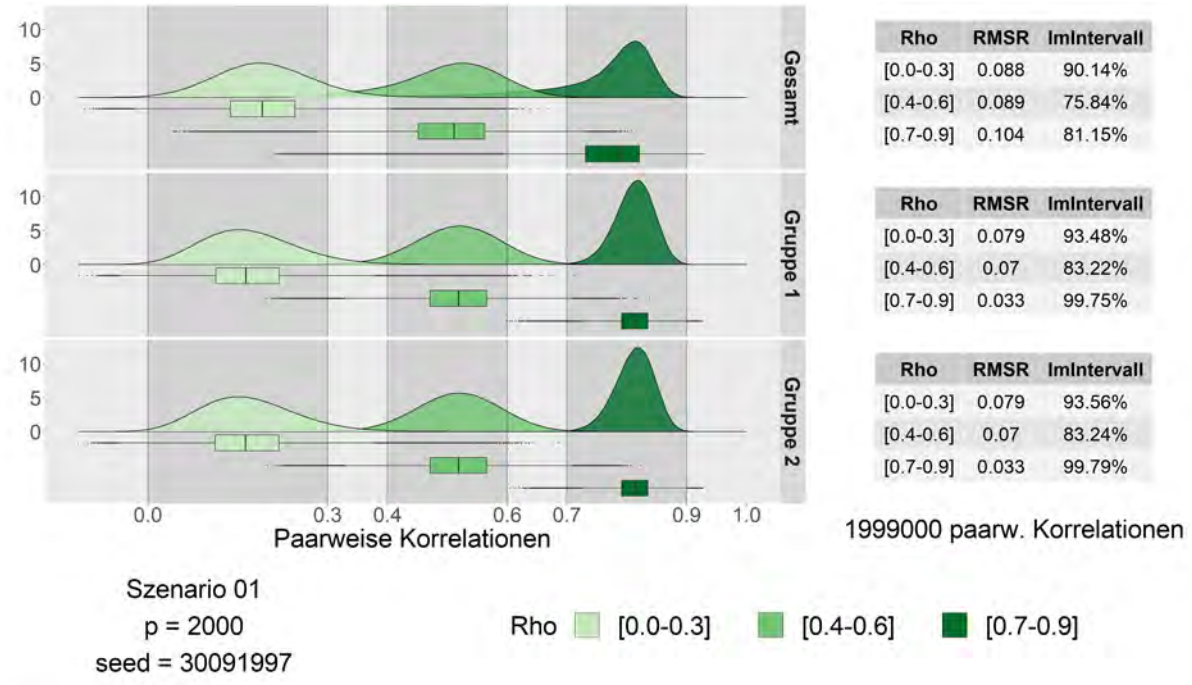


Abbildung 1: Paarweise Korrelationen

Exemplarisch sind hier die paarweisen Korrelationen für das Szenario 01 und $p = 2000$ von drei simulierten Datensätzen gezeigt. Zunächst fällt auf, dass die Korrelationen für I_{ρ_3} stärker konzentriert im vorgegebenen Intervall liegen. Gerade im Fall von hohen Korrelationskoeffizienten ist es schwierig, eine valide Korrelationsmatrix zu finden, weshalb die ursprünglichen Schranken mit dem Algorithmus 1 korrigiert werden mussten. Das Intervall, aus dem die Korrelationen gezogen werden, ist also kleiner. Daher liegen diese näher zusammen. In den Tabellen sind die durchschnittlichen absoluten Abweichungen der Korrelation des simulierten Datensatzes von der vorgegebenen Korrelationsmatrix (*RMSR*, Root Mean Square Residual) sowie der Anteil an paarweisen Korrelationen, die im definierten Intervall liegen. Hier sind von insgesamt 1,999,000 Korrelationskoeffizienten für beide Gruppen knapp 83% im vorgegebenen Abhängigkeitsintervall.

3.4 Umsetzung in R

Das `R`-Paket `SimCorrMix` von Fialkowski und Tiwari [34] implementiert genau diese Simulation korrelierter Daten mit unterschiedlichen Randverteilungen, und ermöglicht zusätzlich die Mischung stetiger und diskreter Variablen. Da allerdings schon die Berechnung von Datensätzen in Dimensionen wie $p = 500$ enorme Rechenzeit in Anspruch genommen hat, wurden die für den Fall von ZINB-Randverteilungen wichtigen Funktionen übernommen und teilweise überarbeitet.

Für die Berechnung der unteren und oberen Schranken (3.8) (3.9) der Korrelation wird zunächst ein Zufallsvektor $\boldsymbol{U} \sim U(0,1)$ gezogen. Je länger dieser Vektor, desto zeitintensiver ist die Berechnung der inversen Verteilungsfunktion, aber umso genauer werden auch die Schranken approximiert. Als Standardwert ist $|\boldsymbol{U}| = 100000$ (`= nbounds`) festgelegt. In der Implementation von Fialkowski und Tiwari werden für die beiden Schranken und jede Korrelation ρ_{ij} insgesamt 4 inverse Verteilungsfunktionen berechnet ($F_{X_i}^{-1}(\boldsymbol{U})$, $F_{X_j}^{-1}(1-\boldsymbol{U})$, $F_{X_i}^{-1}(\boldsymbol{U})$, $F_{X_j}^{-1}(\boldsymbol{U})$), was $4 \cdot \sum_{i=1}^{p-1} i$ Kalkulationen entspricht.
Berechnet man jedoch zuerst für alle Variablen $F_{X_i}^{-1}(\boldsymbol{U})$ und $F_{X_i}^{-1}(1-\boldsymbol{U})$ und speichert diese Ergebnisse in einer Liste um sie dann paarweise zu kombinieren, ergeben sich nur noch $2 \cdot p$ Berechnungen der inversen Verteilungsfunktionen.

Mithilfe der `R`-Bibliotheken `parallel` und `doSNOW` [35] wurde die Berechnung der Quantile zusätzlich parallelisiert, wodurch wiederum deutlich Rechenzeit gespart werden konnte. Da die Schranken unabhängig von der vorgegebenen Korrelationsmatrix sind, lohnt es sich ebenfalls die für ein Szenario und p kalkulierten Schranken zwischenzuspeichern. Diese können dann für alle verschiedenen Korrelationsmatrizen abgerufen und verwendet werden, um $\boldsymbol{P_Z}$ (3.11) zu berechnen. Durch diese Zeiteinsparungen lassen sich auch Datensätze in höhe-

ren Dimensionen in angemessener Zeit berechnen.

Schritt 3, das Ziehen aus einer multivariaten Normalverteilung, wird im Paket `SimCorrMix` durch Anwendung einer Eigenwertzerlegung auf $\boldsymbol{P_Z}$ und Singulärwertzerlegung eines multivariat standardnormalverteilten Zufallsvektors $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ ersetzt. Dadurch erhält man ebenfalls einen Zufallsvektor $\boldsymbol{Z}$ mit $Korr(\boldsymbol{Z}) = \boldsymbol{P_Z}$.

Um die Ergebnisse robuster zu gestalten, wird jeder Datensatz 10 Mal mit unterschiedlichen Seeds simuliert. Dadurch ergeben sich letztendlich für 3 Szenarien, 3 Abhängigkeitsintervalle, 4 Variablenanzahlen und 10 Seeds insgesamt $3 \cdot 3 \cdot 4 \cdot 10 = 360$ zu simulierende Datensätze.

Im Laufe der Berechnungen der Clusteringergebnisse kam es zu Problemen mit `Mclust()`. Bei wenigen Datensätze führte das Aufrufen der Funktion zum Absturz des Programms. In Rücksprache mit dem Paketersteller Luca Scrucca konnte er mir bestätigen, dass das Modell `VVE` in Kombination mit 2 Clustern zwar ein Ergebnis liefert, dies aber unbestimmt viel Zeit in Anspruch nimmt. Daher wurde für die Datensätze, bei denen dieses Problem auftritt, das Modell `VVE` ausgeschlossen. Diese sind in der Funktion `is_exception()` aufgeführt.

Die Dimensionsreduktionsmethoden werden mit den Funktionsstandardwerten ausgeführt, in der PCA werden die Daten zusätzlich vorher zentriert, was meist bessere Ergebnisse liefert. Da t-SNE und PCA Projektionen im hohen Wertebereich erzeugen, werden diese mit der Funktion `scale()` standardisiert, da hier die distanzbasierten Clusteringverfahren bessere Ergebnisse erzeugen.

4 Ergebnisse

Um an die Ergebnisse heranzuführen sollen zunächst wieder die Datensätze für $p = 2000$ und alle Szenarien einzeln betrachtet werden. Die Grafiken sind wie folgt aufgebaut: In einer Zeile ist in allen Streudiagrammen die von der Dimensionsreduktion erzeugte zweidimensionale Darstellung abgebildet, jeweils mit eigener Skalierung. Für die PCA ist die erste Hauptkomponente auf der x-Achse, die zweite auf der y-Achse abgebildet. Im linken Diagramm sind die wahren Cluster farbig markiert, die drei folgenden Streudiagramme stellen die Ergebnisse der Zuordnung der jeweiligen Clusteringverfahren dar. In der unteren Tabelle sind die entsprechenden ARI's dargestellt. In der rechten oberen Ecke des mclust-Streudiagramms ist das Model angegeben, welches die Funktion anhand des BIC ausgewählt hat.

Anschließend sollen alle Datensätze betrachtet werden. Für jeden Datensatz wird mit jeweils 10 verschiedenen Seeds eine zweidimensionale Darstellung aus UMAP, t-SNE und PCA erzeugt, auf welche dann die Clustering-Funktionen `kmeans()`, `hclust()` und `Mclust()` angewendet werden. Somit ergeben sich für jede mögliche Kombination aus Szenarien, Abhängigkeitsintervallen und Variablenanzahlen 100 Clustering-Ergebnisse.
Die Güte der Zuordnung wird mit dem Adjusted Rand Index (ARI) gewertet. Aufgeteilt nach den Methoden zur Dimensionsreduktion und den unterschiedlichen Szenarien werden diese mit Boxplots visualisiert.

Zuletzt werden die durchschnittlichen Ergebnisse aller Kombinationen der Methoden gemeinsam betrachtet. Hierfür sind Tabellen angegeben, die den durchschnittlichen ARI enthalten. In den Tabellen ist der beste durchschnittliche ARI für die Kombination aus I_{ρ_i} und p jeweils **fett** markiert. Im Falle mehrerer Werte sind diese zusätzlich ***kursiv***. Hieraus kann dann abgelesen werden, welche Kombination aus Dimensionsreduktion und Clusteringverfahren im Durchschnitt die besten Ergebnisse für eine vorhandene Abhängigkeitsstruktur und feste Variablenanzahl sowie Heterogenität der Daten liefert.

4.1 UMAP

Visualisierung

Bei hoher Homogenität der Gruppen und niedriger Abhängigkeit ist mit UMAP keine klare Zuordnung zu erreichen. Je höher die Korrelation jedoch ansteigt, desto besser werden die beiden Gruppen voneinander gespalten.
Für Szenario 02 können die Cluster bei I_{ρ_1} zwar besser gruppiert, aber nicht klar separiert werden. Dafür werden sie jetzt bereits bei moderater Abhängigkeit gut voneinander getrennt. Unterscheiden sich die Parameter der Gruppen stark können diese in niedriger Abhängigkeit wieder etwas besser geclustert, aber immer noch nicht klar voneinander unterschieden werden.

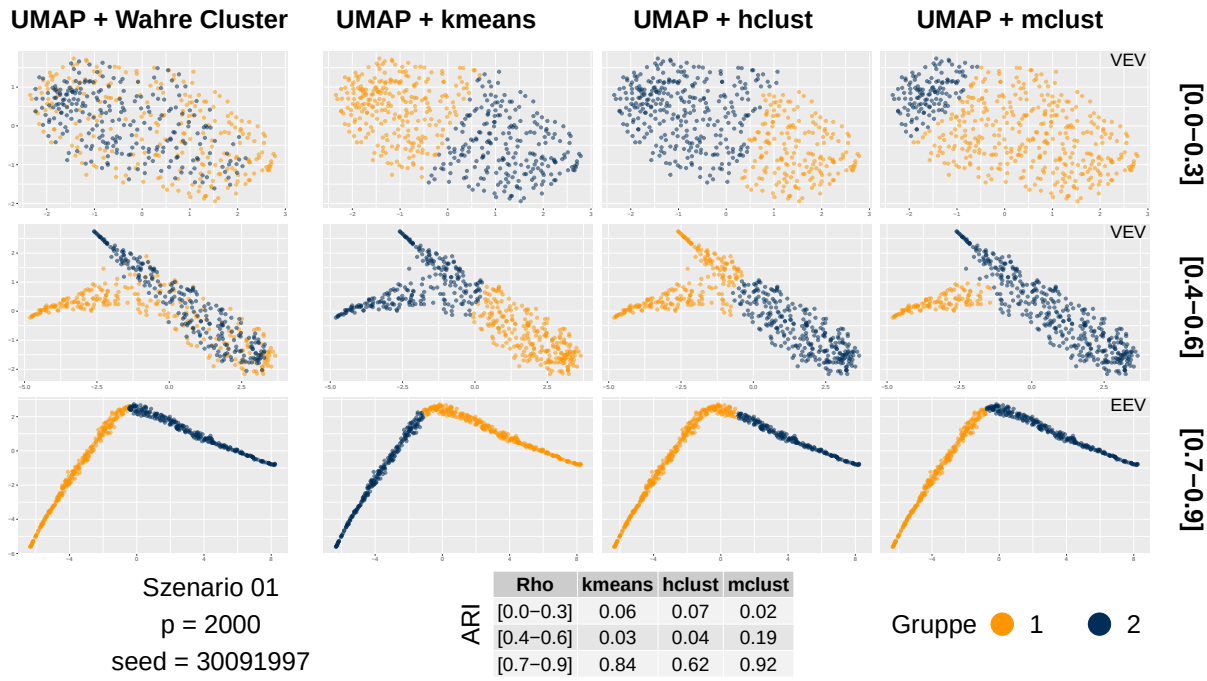


Abbildung 2: Zuordnung und ARI, Szenario 01, p = 2000, UMAP & Clustering

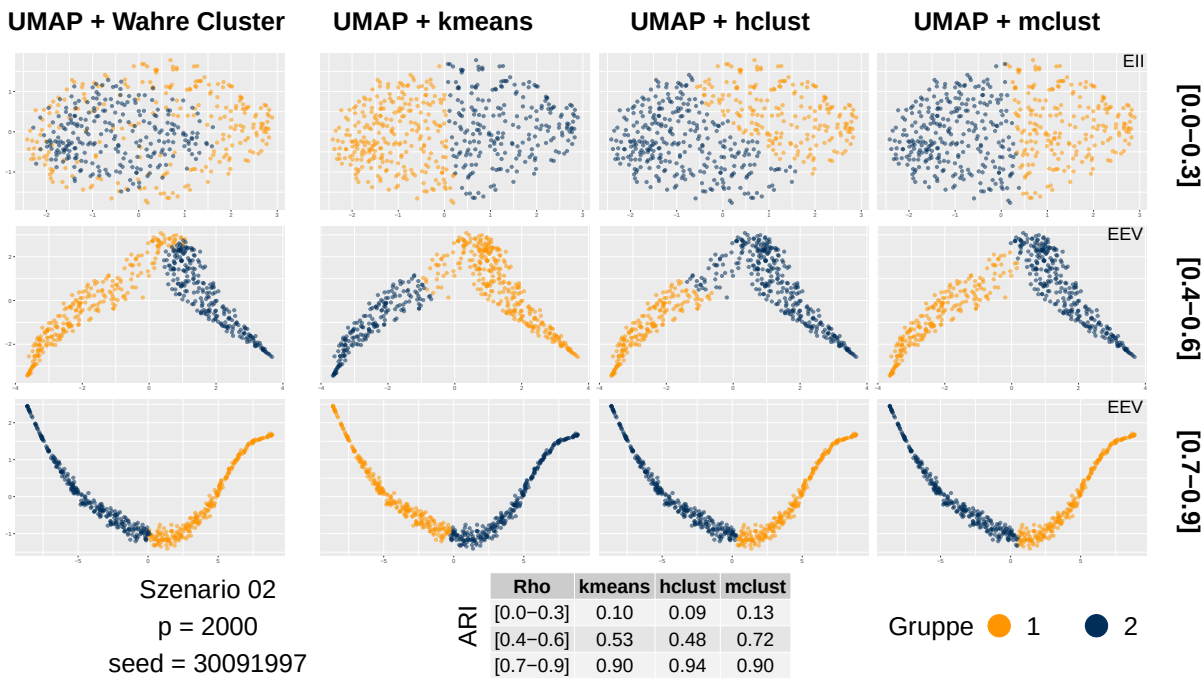


Abbildung 3: Zuordnung und ARI, Szenario 02, p = 2000, UMAP & Clustering

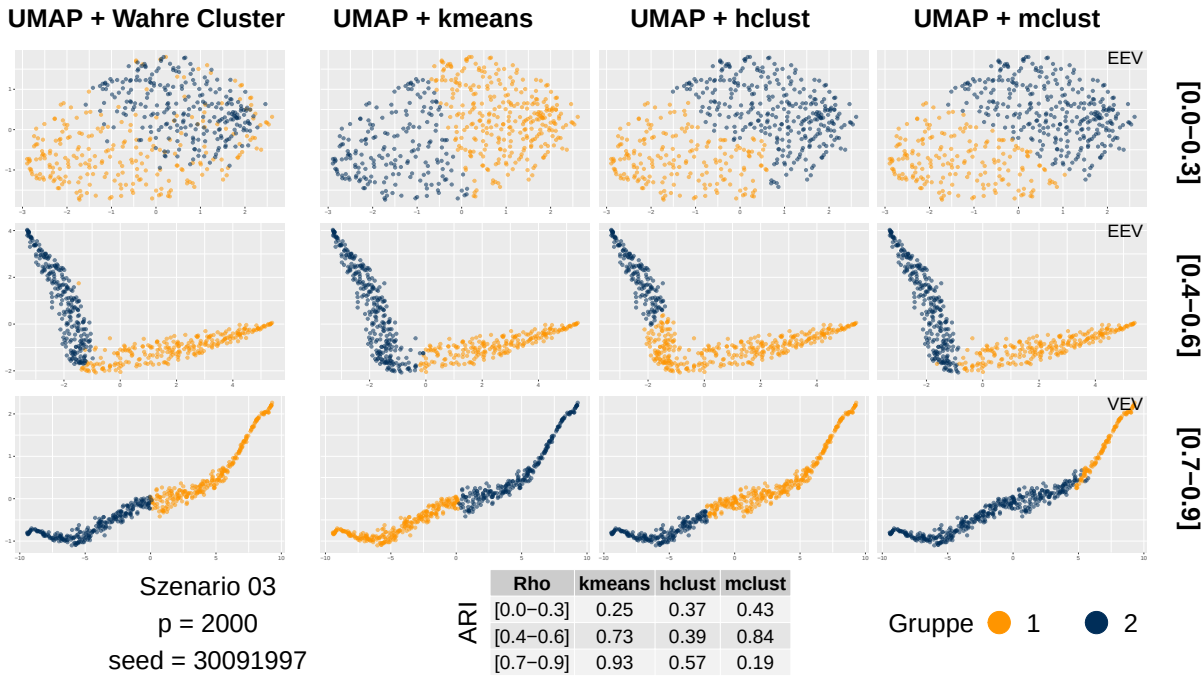


Abbildung 4: Zuordnung und ARI, Szenario 03, p = 2000, UMAP & Clustering

ARI

In Verbindung mit kmeans werden die Gruppen in der homogenen Situation in Szenario 01 bei niedriger und moderater Korrelation nicht erkannt. In Szenario 02 und 03 kann hier ebenfalls keine klare Zuordnung erreicht werden. Je mehr Variablen hinzugezogen werden, desto besser werden die Ergebnisse bei hoher Abhängigkeit.
Mit hclust können die Gruppen nie ganz klar erkannt werden, einige gute Zuordnungen gibt es in Szenario 02 und 03 bei hoher Korrelation. Die Tendenz eines steigenden ARI's mit zunehmender Abhängigkeit kann über die Szenarien beobachtet werden, jedoch ist die durchschnittliche Zuordnung in keinem Fall perfekt.
Vor allem in Szenario 01 wird deutlich, dass mclust hier bei hoher Abhängigkeit die besten Ergebnisse erzeugt. Unterscheiden sich die Gruppen jedoch stärker, ist die Zuordnung mit mclust zwar noch gut, kmeans liefert aber konsistenter eine bessere Zuordnung. Bei moderater Abhängigkeit in Szenario 02 und 03 erhält man mit modellbasiertem Clustering bessere Ergebnisse als mit kmeans und hclust.

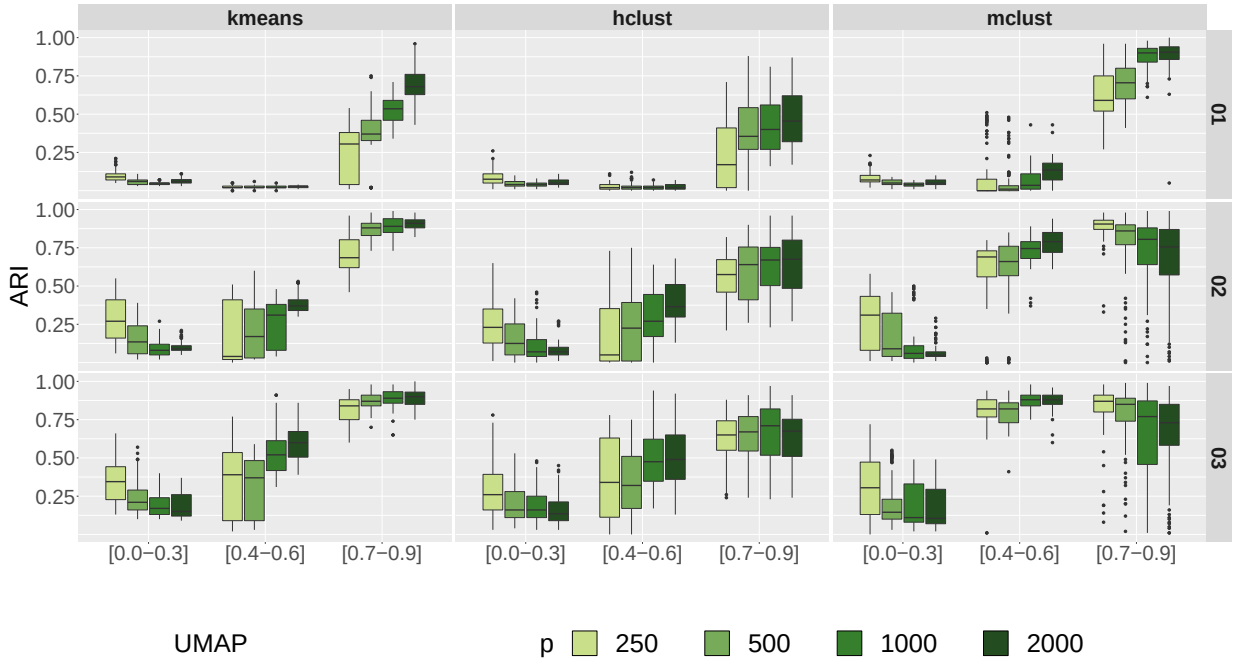


Abbildung 5: ARI, UMAP & Clustering

4.2 t-SNE

Visualisierung

In Szenario 01 kann t-SNE ähnlich wie UMAP die Gruppen mit zunehmender Abhängigkeit besser visuell separieren. Vor allem bei hoher Korrelation sind die Gruppen klar voneinander zu unterscheiden, allerdings liefert hier nur die Verbindung mit mclust gute Ergebnisse.
Bei moderater Abhängigkeit können die Gruppen in Szenario 02 etwas besser unterschieden werden. Gruppe 2 wird bei niedriger Korrelation zwar mehr zusammen geclustert, kann aber nicht genau von Gruppe 1 unterschieden werden.
In Szenario 03 sieht die Projektion ähnlich aus, die Gruppen werden gemeinsam gut angeordnet. Jedoch liegt hier keine klare visuelle Separation vor, falls die wahren Cluster unbekannt sind.

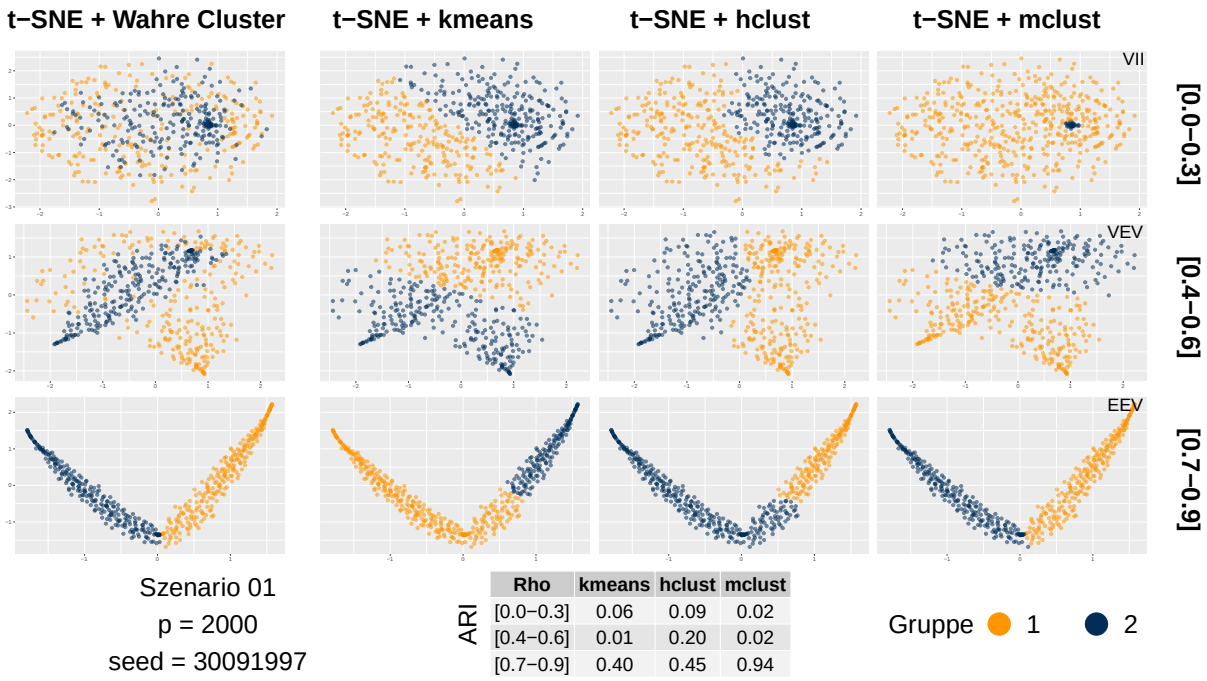


Abbildung 6: Zuordnung und ARI, Szenario 01, p = 2000, t-SNE & Clustering

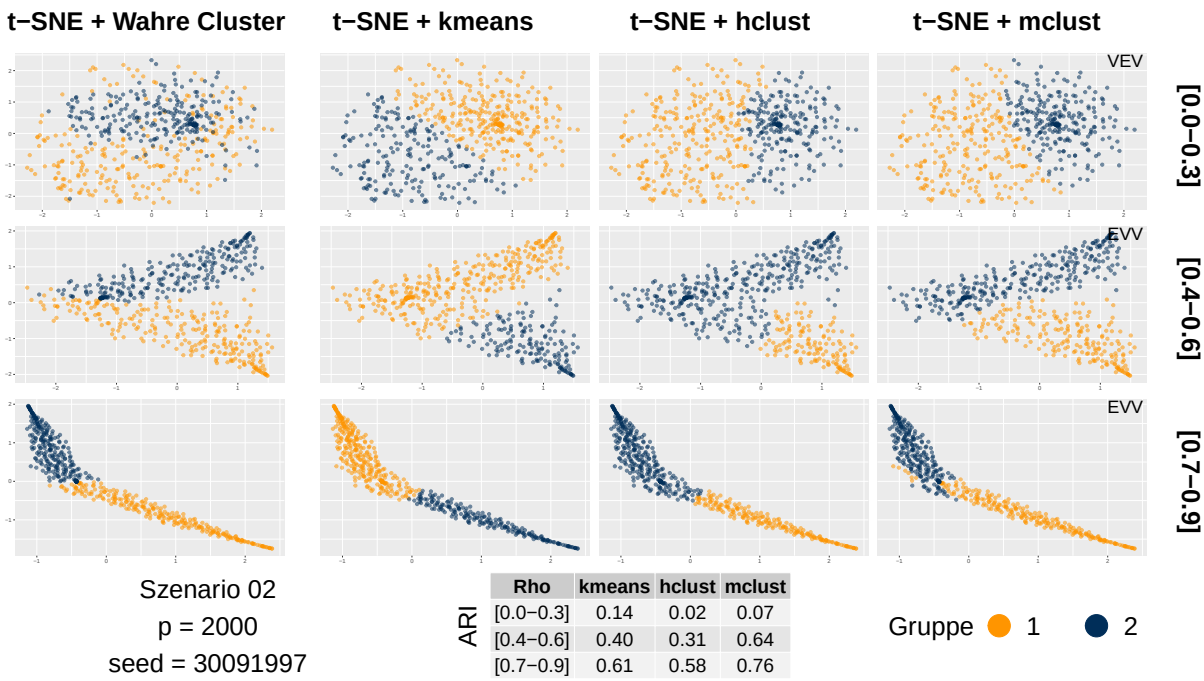


Abbildung 7: Zuordnung und ARI, Szenario 02, p = 2000, t-SNE & Clustering

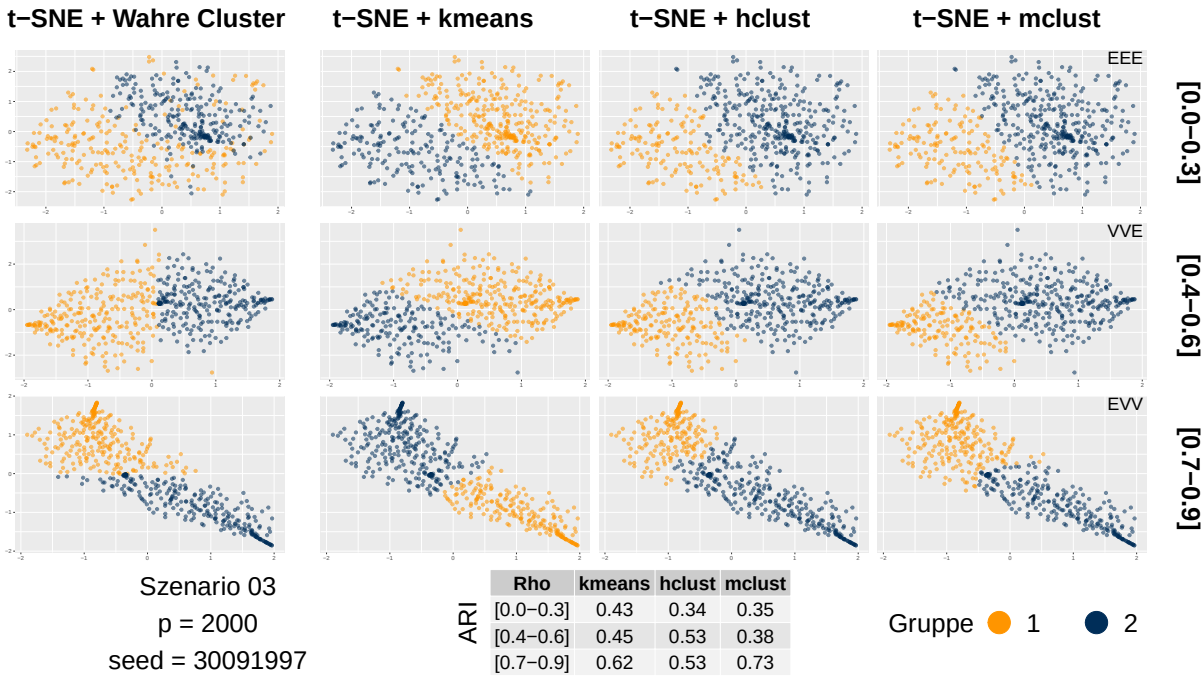


Abbildung 8: Zuordnung und ARI, Szenario 03, p = 2000, t-SNE & Clustering

ARI

Für die Kombination mit kmeans ist eine steigende Güte der Zuordnung mit steigender Korrelation zu beobachten. Auch mit zunehmender Heterogenität der Gruppen erhält man bessere Ergebnisse. Eine eindeutige Zuordnung kann jedoch in fast keinem Fall erreicht werden.
Auch bei hclust sieht man, dass sich die Ergebnisse mit zunehmender Heterogenität und Abhängigkeit verbessern. Wie bei kmeans kann für Szenario 01 und niedrige Korrelation keine gute Zuordnung erreicht werden.
Mclust produziert vor allem für $p = 2000$ sehr gute Ergebnisse bei hoher Korrelation und Homogenität der Gruppen. Für geringe Variablenanzahl können gute Ergebnisse bei hoher Abhängigkeit erzielt werden. Auch hier ist ein leichter Trend der Verbesserung der Zuordnung mit steigender Abhängigkeit zu beobachten.

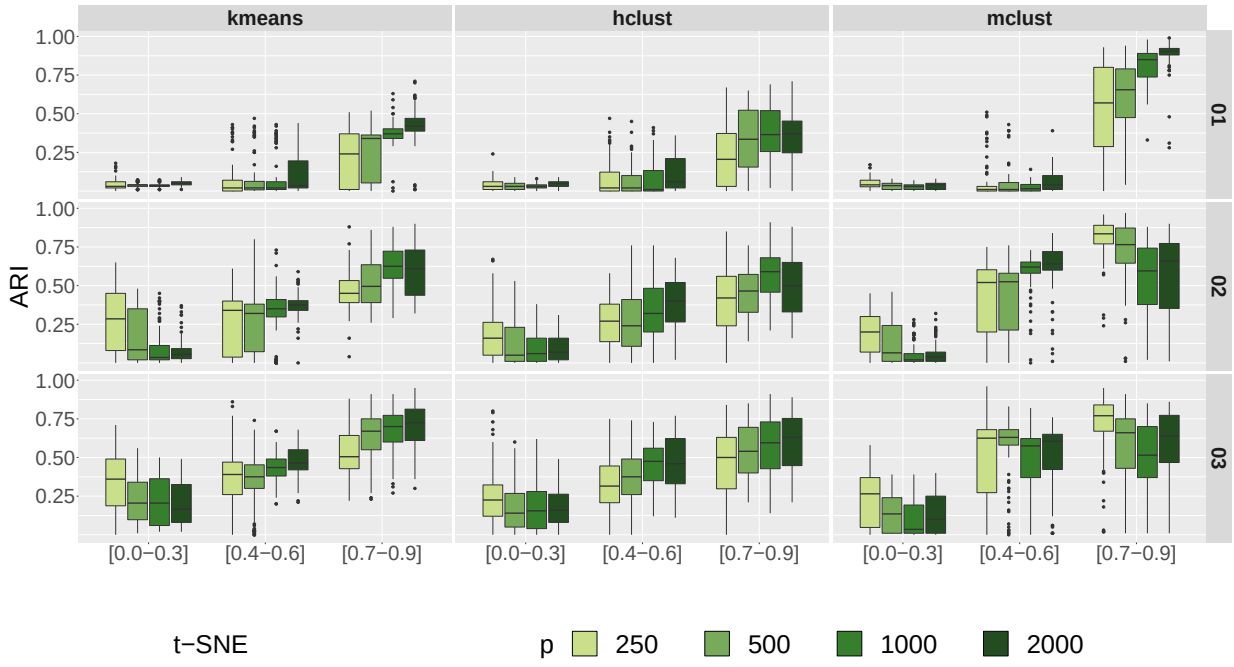


Abbildung 9: ARI, t-SNE & Clustering

4.3 Hauptkomponentenanalyse

Visualisierung

Der größte Anteil der Varianz wird von der ersten Hauptkomponente auf der x-Achse dargestellt. Wird die zweite Hauptkomponente hinzugezogen, lassen sich visuell in jeder Grafik zwei Gruppen erkennen. Gruppe 2, deren Parameterintervalle für μ kleiner waren als die von Gruppe 1 streut innerhalb des Clusters weniger. Da die PCA ein Verfahren ist das neue unkorrelierte Variablen bildet, hat die Abhängigkeit der Daten hier nur wenig Einfluss.

Die Hauptkomponentenanalyse liefert aufgrund der visuellen Projektion nur in Kombination mit mclust gute Ergebnisse, für das Intervall I_{ρ_1} sogar eine perfekte Zuordnung. Die Heterogenität der Gruppen hat keinen starken sichtbaren Einfluss auf die Darstellung.

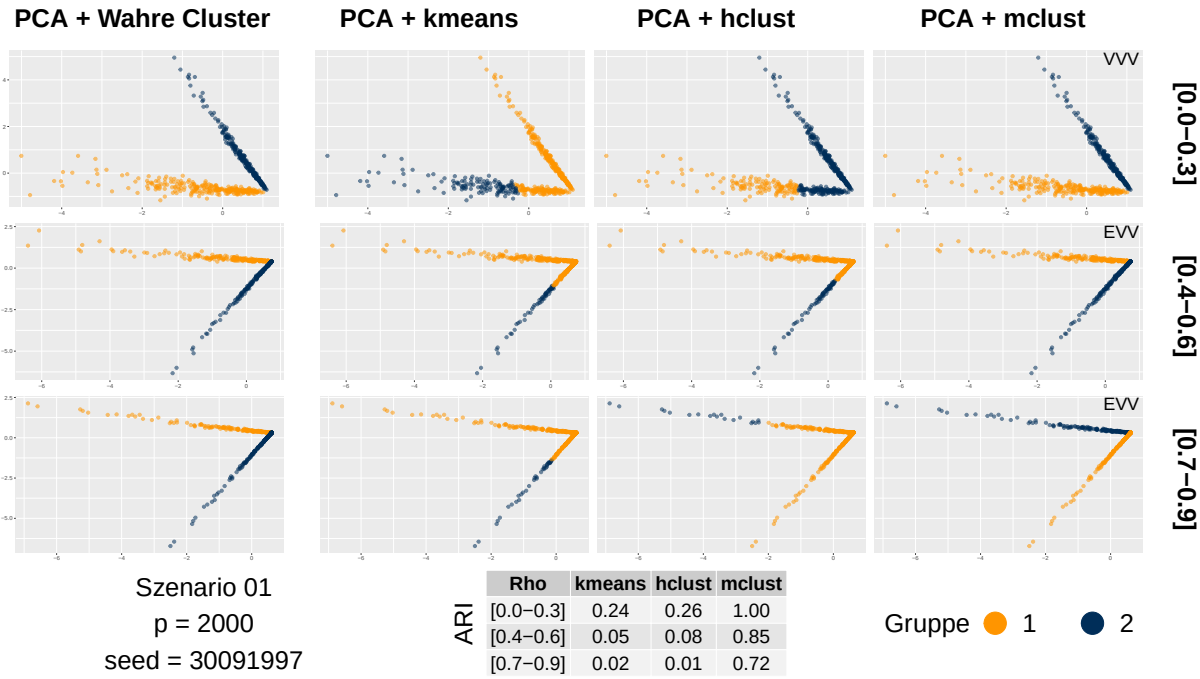


Abbildung 10: Zuordnung und ARI, Szenario 01, p = 2000, PCA & Clustering

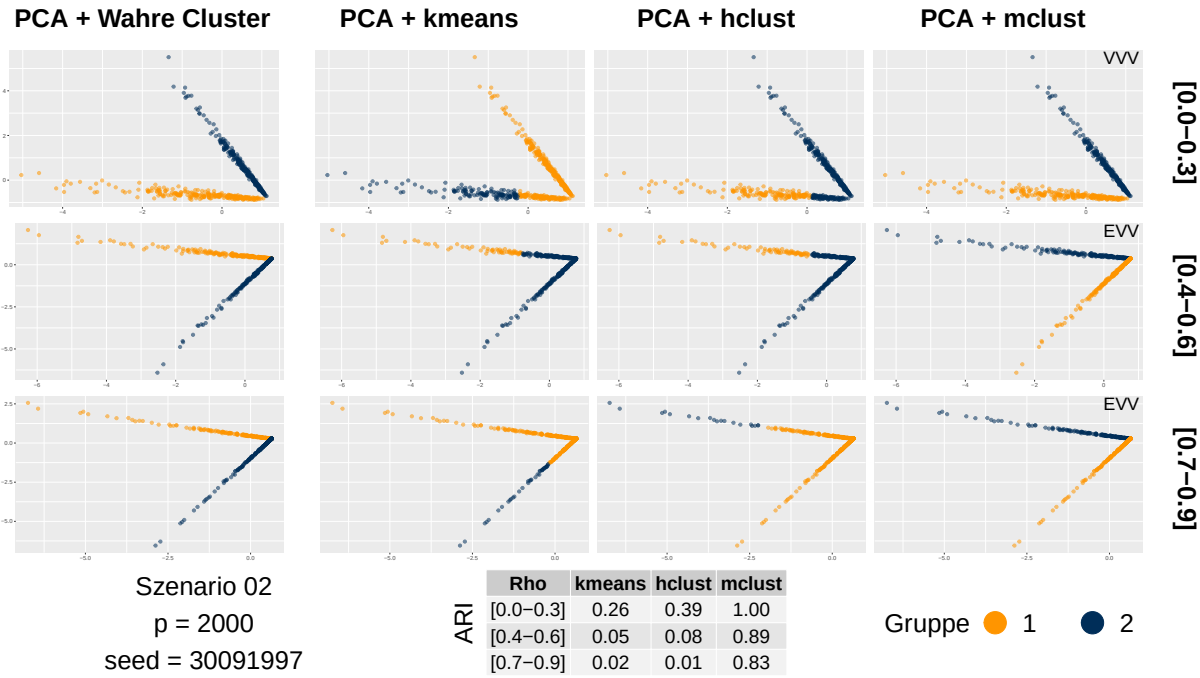


Abbildung 11: Zuordnung und ARI, Szenario 02, p = 2000, PCA & Clustering

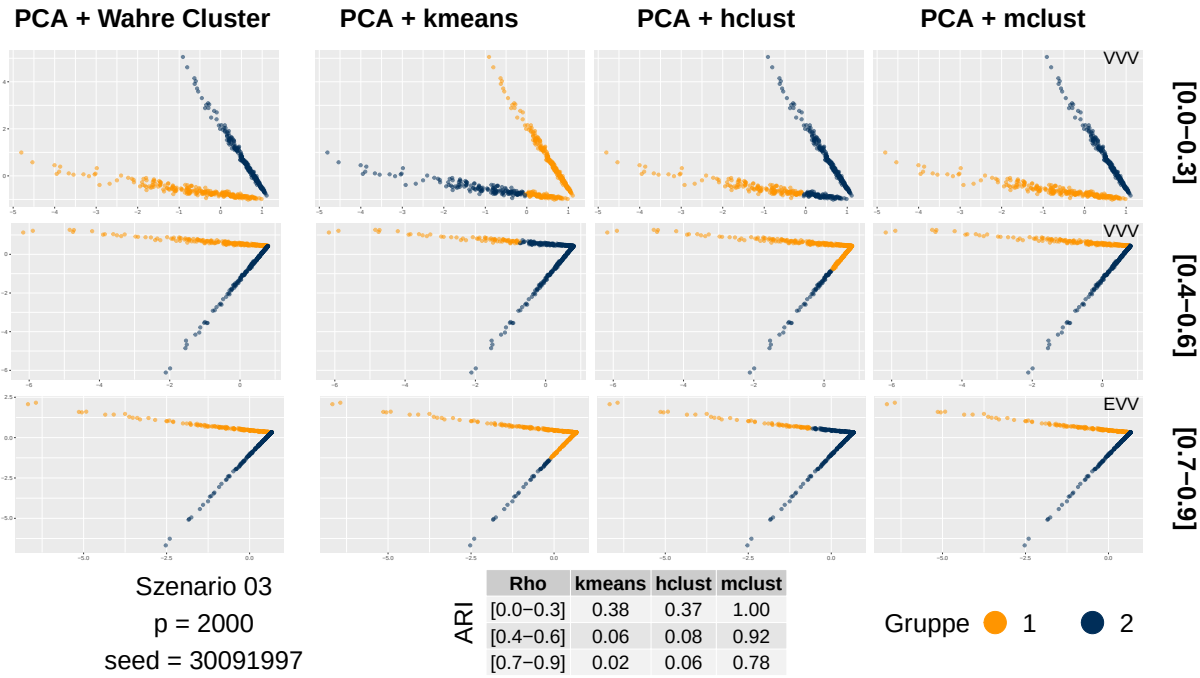


Abbildung 12: Zuordnung und ARI, Szenario 03, p = 2000, PCA & Clustering

ARI

In moderater und hoher Abhängigkeitsstruktur kann mit kmeans keine Zuordnung erreicht werden. In Szenario 03 mit heterogenen Gruppen und niedriger Korrelation dagegen erhält man eine perfekte Gruppeneinteilung.
Mit hclust können für $I_{\rho_{2,3}}$ ebenfalls keine guten Ergebnisse produziert werden. In I_{ρ_1} ist die Zuordnung etwas besser, jedoch nicht eindeutig.
Bei niedriger Korrelation erreicht mclust zuverlässig nahezu perfekte Ergebnisse. In den übrigen Abhängigkeitsstrukturen können auch gute Zuordnungen erzeugt werden, die sich mit zunehmender Anzahl an Variablen verbessern. Über alle Szenarien kann ein leichter Rückgang des ARI's mit steigender Korrelation beobachtet werden.

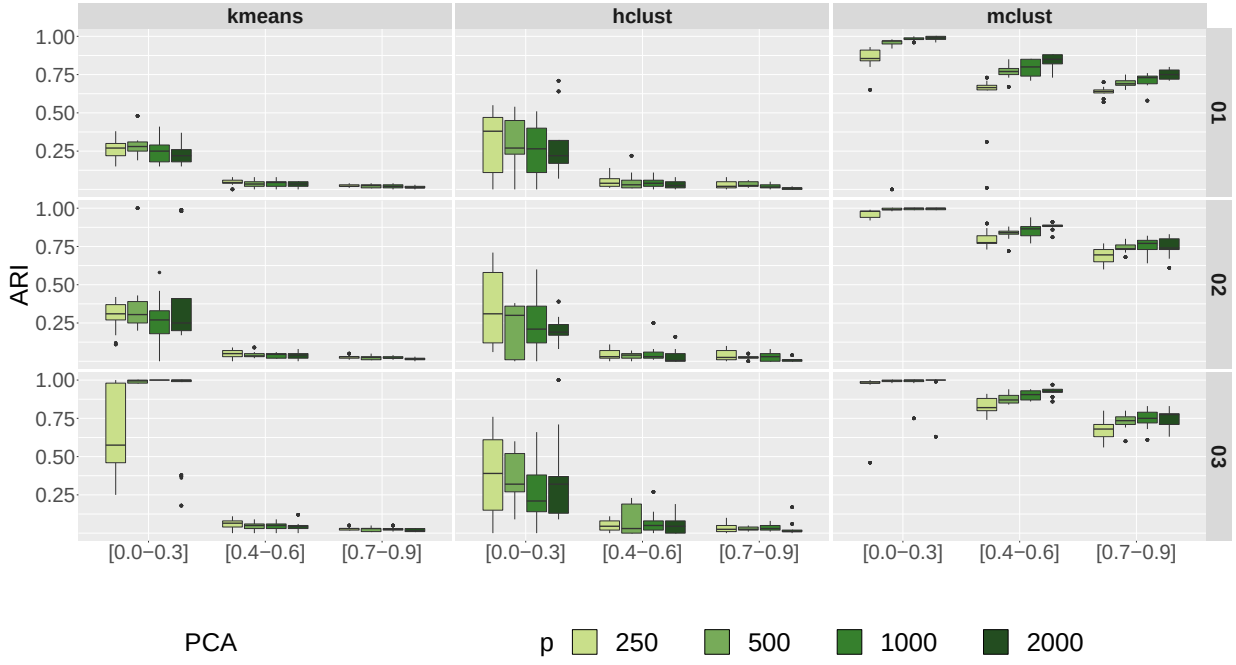


Abbildung 13: ARI, PCA & Clustering

4.4 Gesamtbetrachtung

Szenario 01

Die Hauptkomponentenanalyse liefert in Kombination mit mlcust die besten Ergebnisse für niedrige und moderate Abhängigkeit der Gruppen. Alle anderen Algorithmen zeigen hier deutliche Schwierigkeiten. Für den Fall von hoher Korrelation ist für $p = 250$ zunächst PCA und mclust immer noch die beste Alternative, bei mehr Variablen zeigt sich jedoch, dass UMAP und mclust die bessere Wahl ist. Für $p = 2000$ liefert auch t-SNE und mclust das beste Ergebnis.

		UMAP			t-SNE			PCA		
Rho	p	k-means	hclust	mclust	k-means	hclust	mclust	k-means	hclust	mclust
I_{ρ_1}	250	0.10	0.08	0.08	0.04	0.04	0.05	0.27	0.31	**0.85**
	500	0.06	0.05	0.05	0.04	0.03	0.03	0.29	0.29	**0.87**
	1000	0.05	0.04	0.04	0.04	0.03	0.03	0.25	0.26	**0.98**
	2000	0.06	0.06	0.05	0.05	0.05	0.03	0.23	0.30	**0.99**
I_{ρ_2}	250	0.02	0.03	0.10	0.07	0.08	0.05	0.05	0.05	**0.57**
	500	0.02	0.02	0.06	0.07	0.06	0.05	0.04	0.06	**0.77**
	1000	0.02	0.02	0.06	0.08	0.07	0.03	0.04	0.04	**0.79**
	2000	0.03	0.03	0.13	0.10	0.11	0.06	0.03	0.03	**0.84**
I_{ρ_3}	250	0.23	0.23	0.63	0.21	0.22	0.53	0.02	0.03	**0.64**
	500	0.37	0.38	**0.70**	0.26	0.32	0.61	0.02	0.03	0.69
	1000	0.53	0.42	**0.87**	0.37	0.37	0.81	0.02	0.02	0.71
	2000	0.69	0.48	***0.88***	0.42	0.37	***0.88***	0.01	0.01	0.75

Tabelle 3: Durchschnittlicher ARI - Szenario 01 - Dimensionsreduktion & Clustering

Szenario 02

Auch in Szenario 02 schneidet die PCA bei niedriger und moderater Abhängigkeit am besten ab. Hier liefern im Vergleich zu Szenario 01 auch die anderen Methoden teilweise bessere Ergebnisse, beispielsweise UMAP und mclust für I_{ρ_2} und $p = 1000, 2000$. Bei hoher Korrelation erhält man wieder die besten Ergebnisse mit UMAP. Die Kombination mit mclust ist hier für $p = 250$ am besten, bei mehr Variablen setzt sich kmeans durch.

		UMAP			t-SNE			PCA		
Rho	p	k-means	hclust	mclust	k-means	hclust	mclust	k-means	hclust	mclust
I_{ρ_1}	250	0.28	0.24	0.27	0.28	0.18	0.19	0.31	0.37	**0.96**
	500	0.15	0.15	0.16	0.17	0.12	0.13	0.37	0.24	**0.99**
	1000	0.09	0.10	0.10	0.10	0.10	0.05	0.25	0.26	**1.00**
	2000	0.10	0.08	0.07	0.08	0.09	0.06	0.40	0.20	**1.00**
I_{ρ_2}	250	0.19	0.19	0.59	0.27	0.26	0.42	0.05	0.04	**0.80**
	500	0.21	0.22	0.60	0.27	0.27	0.42	0.04	0.04	**0.83**
	1000	0.27	0.29	0.73	0.32	0.32	0.58	0.04	0.06	**0.86**
	2000	0.38	0.39	0.78	0.37	0.39	0.62	0.04	0.03	**0.88**
I_{ρ_3}	250	0.70	0.56	**0.88**	0.46	0.41	0.81	0.03	0.04	0.69
	500	**0.88**	0.60	0.77	0.52	0.46	0.73	0.02	0.02	0.74
	1000	**0.89**	0.64	0.72	0.62	0.56	0.52	0.02	0.03	0.76
	2000	**0.91**	0.65	0.67	0.60	0.50	0.55	0.02	0.01	0.74

Tabelle 4: Durchschnittlicher ARI - Szenario 02 - Dimensionsreduktion & Clustering

Szenario 03

Die PCA erzeugt für niedrige und moderate Korrelation wieder die besten Ergebnisse. Aber je heterogener die Gruppen werden, desto besser können auch die anderen Methoden mithalten. Auch UMAP und mclust erzeugt vergleichbar gute Ergebnisse für moderate Abhängigkeit. In hohen Abhängigkeitsstrukturen kann man wieder wie in Szenario 02 die beste Zuordnung mit UMAP und entsprechender Clusteringmethode erhalten.

		UMAP			t-SNE			PCA		
Rho	p	k-means	hclust	mclust	k-means	hclust	mclust	k-means	hclust	mclust
I_{ρ_1}	250	0.36	0.29	0.31	0.35	0.25	0.25	0.66	0.37	**0.93**
	500	0.25	0.21	0.20	0.23	0.18	0.14	***0.99***	0.36	***0.99***
	1000	0.20	0.19	0.19	0.22	0.18	0.11	**1.00**	0.27	0.97
	2000	0.19	0.16	0.19	0.21	0.19	0.14	0.85	0.37	**0.96**
I_{ρ_2}	250	0.35	0.36	0.80	0.36	0.32	0.50	0.06	0.05	**0.83**
	500	0.33	0.33	0.80	0.34	0.37	0.57	0.05	0.07	**0.88**
	1000	0.53	0.48	0.87	0.43	0.45	0.49	0.05	0.07	**0.90**
	2000	0.60	0.49	0.87	0.47	0.46	0.51	0.04	0.06	**0.92**
I_{ρ_3}	250	0.81	0.63	**0.82**	0.54	0.47	0.71	0.03	0.04	0.68
	500	**0.87**	0.64	0.78	0.64	0.55	0.55	0.03	0.03	0.73
	1000	**0.89**	0.67	0.65	0.68	0.56	0.50	0.03	0.03	0.74
	2000	**0.89**	0.63	0.66	0.70	0.60	0.58	0.02	0.03	0.75

Tabelle 5: Durchschnittlicher ARI - Szenario 03 - Dimensionsreduktion & Clustering

5 Fazit und Ausblick

In dieser Arbeit wurden verschiedene Methoden zur Dimensionsreduktion und Clusteranalyse vorgestellt. Die NORTA-Methode zur Erzeugung von korrelierten Simulationsdaten wurde adaptiert und die Berechnung für diskrete Verteilungen effizient gestaltet. Dadurch konnten auch hochdimensionale Datensätze in relativ kurzer Zeit leicht simuliert werden. Die Erzeugung einer validen Korrelationsmatrix wurde speziell auf die Simulationsmethode und hier definierten Abhängigkeitsintervalle angepasst.

Für die Methoden UMAP und t-SNE konnte eine Verbesserung der Clusteringergebnisse bei steigender Abhängigkeit festgestellt werden, bei der PCA verschlechtern sich diese eher. Insgesamt lieferte die PCA in Kombination mit modellbasiertem Clustering die besten Ergebnisse für niedrige und moderate Korrelation, UMAP und mclust bzw. kmeans kann die besten Resultate für hohe Abhängigkeit erzeugen. Die Ergebnisse sollen lediglich der Orientierung dienen, und sind nicht als endgültige Best Practise Verfahren zu sehen. Bei der Analyse eines neuen Datensatzes ist es immer von Vorteil, mehrere Methoden zu verwenden und die Ergebnisse visuell sowie analytisch zu betrachten.

Es wurden Datensätze mit vorgegebener Korrelation innerhalb der Cluster simuliert. Die Herausforderung, multivariate diskrete Daten mit vorgegebenen Clustern und übergreifender Abhängigkeitsstruktur zu generieren ist keine triviale Aufgabe, und bis zum jetzigen Zeitpunkt existiert noch keine bekannte Lösung dieses Problems. Sahin und Czado [36] präsentieren in Ihrem erst kürzlich veröffentlichtem Artikel eine Methode mithilfe von Copulas, um dieses Problem für stetige Verteilungen zu lösen. Die Autoren weisen darauf hin, dass diese Methode auch auf andere Verteilungen erweitert werden kann. Es ist zu erwarten, dass hier in naher Zukunft Fortschritte gemacht werden. Sobald dies möglich ist, wäre es sicherlich interessant den Einfluss der übergreifenden Abhängigkeitsstruktur des gesamten Datensatzes auf die Clusteringperformance zu untersuchen.

Ein weiterer Ansatz wäre die Abhängigkeitsintervalle breiter zu definieren und andere Verteilungen zum Ziehen der paarweisen Korrelationen zu verwenden, beispielsweise bimodale Verteilungen, sodass sich zwei unterschiedliche Abhängigkeitsstrukturen in einem Datensatz wiederfinde. Hierfür kann natürlich auch einfach aus mehreren Intervallen gezogen werden.

Auch interessant wäre es, mehr als zwei Gruppen zu untersuchen. Dieser Fall wird mit den angefügten Funktionen abgedeckt und ist somit leicht umsetzbar. Die Simulation in dieser Arbeit wurde auf zwei sich in allen Variablen unterscheidenden Gruppen beschränkt.

Es ist natürlich auch möglich, nur einen gewissen Teil der Variablen mit unterschiedlichen Gruppen zu modellieren, oder nur gewisse Parameter zu unterscheiden. Dies ist ebenfalls in die Funktionen mit eingebaut und kann leicht simuliert werden.
Eine genauere Beschreibung ist im R-Code an entsprechender Stelle zu finden.

A Algorithmen

Der in Kapitel 3.2 verwendete Algorithmus zur Erzeugung einer validen Korrelationsmatrix. Die Elemente aus der Normalverteilung können mit `rnorm()` gezogen werden. Die Transformation zur nächsten positiv definiten Matrix wird mit der Funktion `nearPD()` umgesetzt. Der Algorithmus ist im `R`-File `data_generation.R` unter `generate_rho()` zu finden.

Algorithm 1 Erzeugen einer validen Korrelationsmatrix

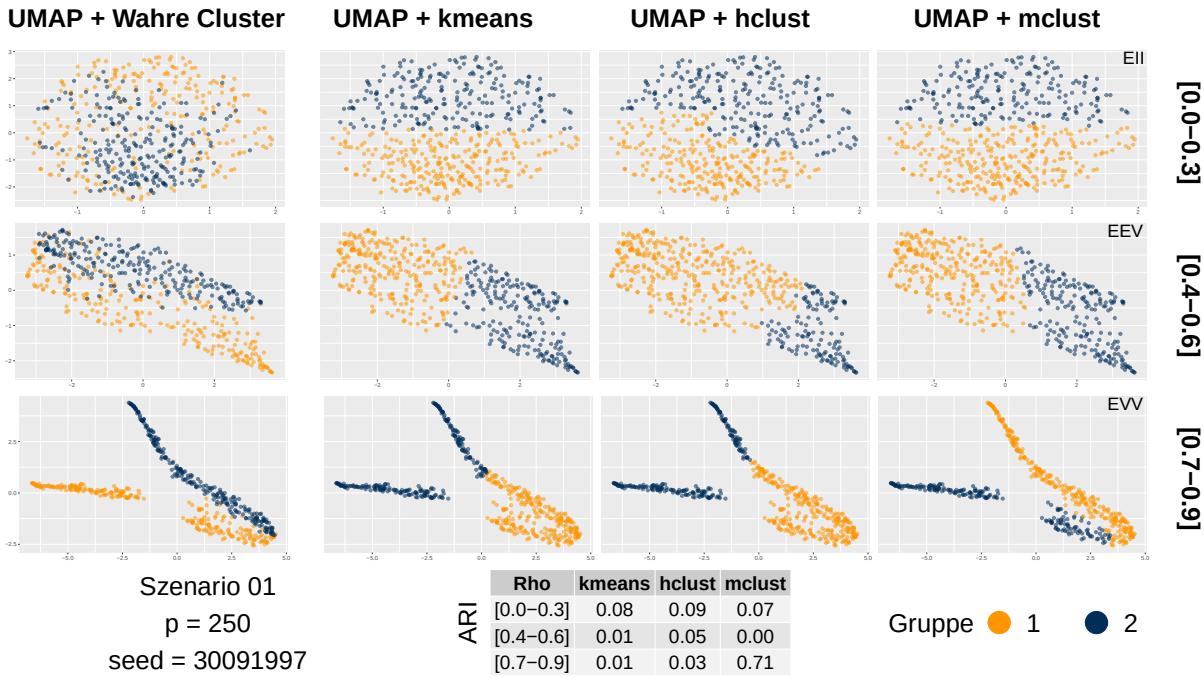
$n \leftarrow 10$ ▷ Maximale Anzahl an Iterationen
$c \leftarrow 0$ ▷ Verkleinerung des Intervalls nach n Iterationen

for i in $1 : n$ **do**
  for j in $1 : n$ **do**
    $R \leftarrow$ Ziehe q Elemente aus $N(0, 1)$
    $\widetilde{R} \leftarrow s(R, \rho_{min} - c,\ \rho_{max} + c)$ ▷ Funktion aus (3.6)
    $\widetilde{P} \leftarrow$ Erzeuge Matrix mit Elementen aus $\widetilde{R}$, 1 auf der Diagonalen ▷ (3.7)
    $\widetilde{P} \leftarrow$ Transformation zur nächsten positiv definiten Matrix

    if $\widetilde{P}$ = Valide Korrelationsmatrix & alle $\rho \in [\rho_{min}, \rho_{max}]$ **then**
      return $\widetilde{P}$
    end if
  end for
  $c \leftarrow 0.001 \cdot i$ ▷ Verkleinerung des Intervalls nach n Iterationen
end for

B Abbildungen

Im Folgenden sind weitere Abbildungen aufgelistet, die mit dem ersten Seed 30091997 erstellt wurden.



Rho	kmeans	hclust	mclust
[0.0−0.3]	0.08	0.09	0.07
[0.4−0.6]	0.01	0.05	0.00
[0.7−0.9]	0.01	0.03	0.71

Abbildung 14: Zuordnung und ARI, Szenario 01, p = 250, UMAP & Clustering

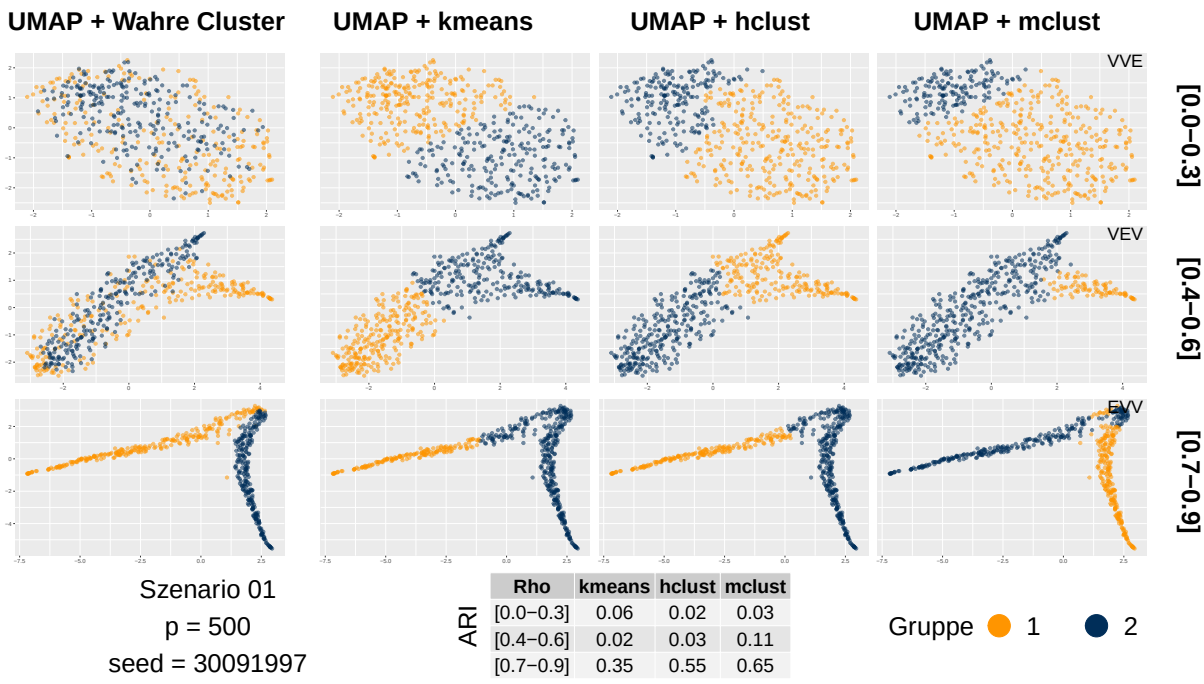


Abbildung 15: Zuordnung und ARI, Szenario 01, p = 500, UMAP & Clustering

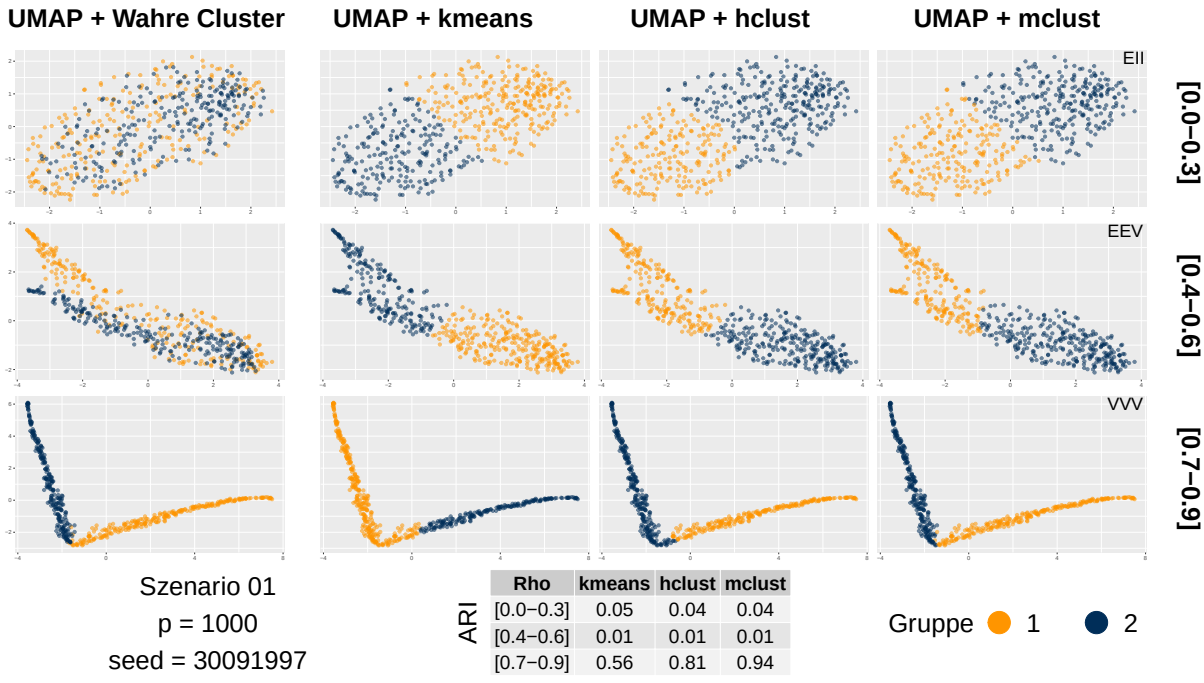
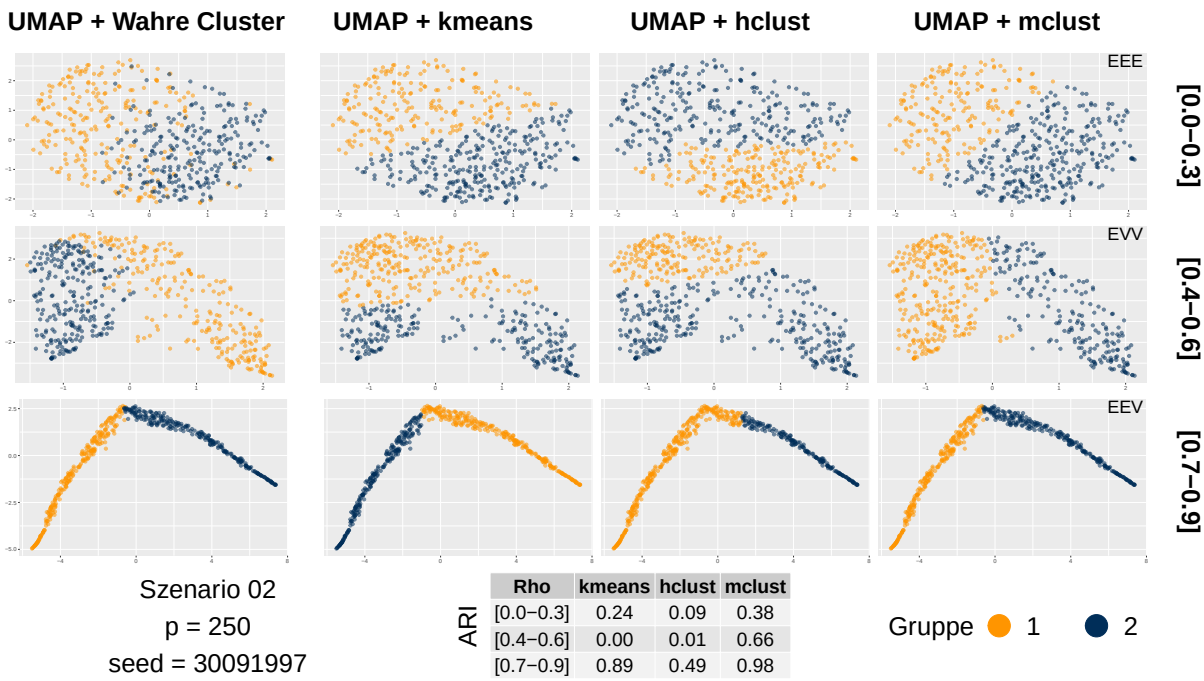
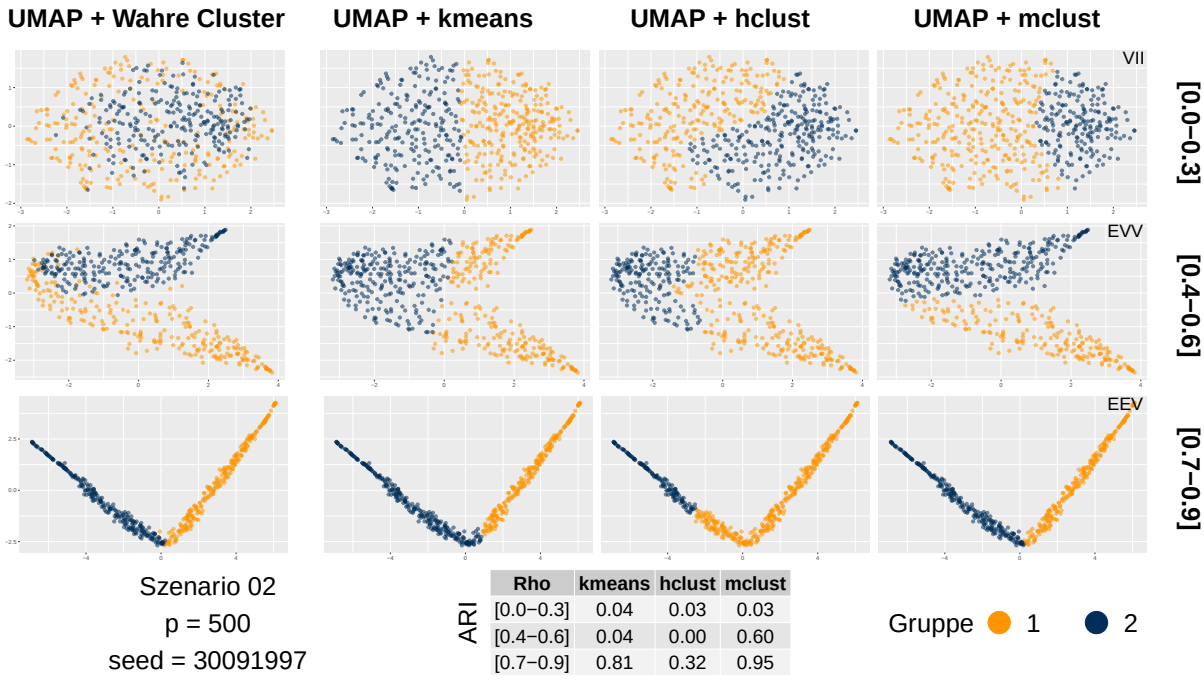


Abbildung 16: Zuordnung und ARI, Szenario 01, p = 1000, UMAP & Clustering



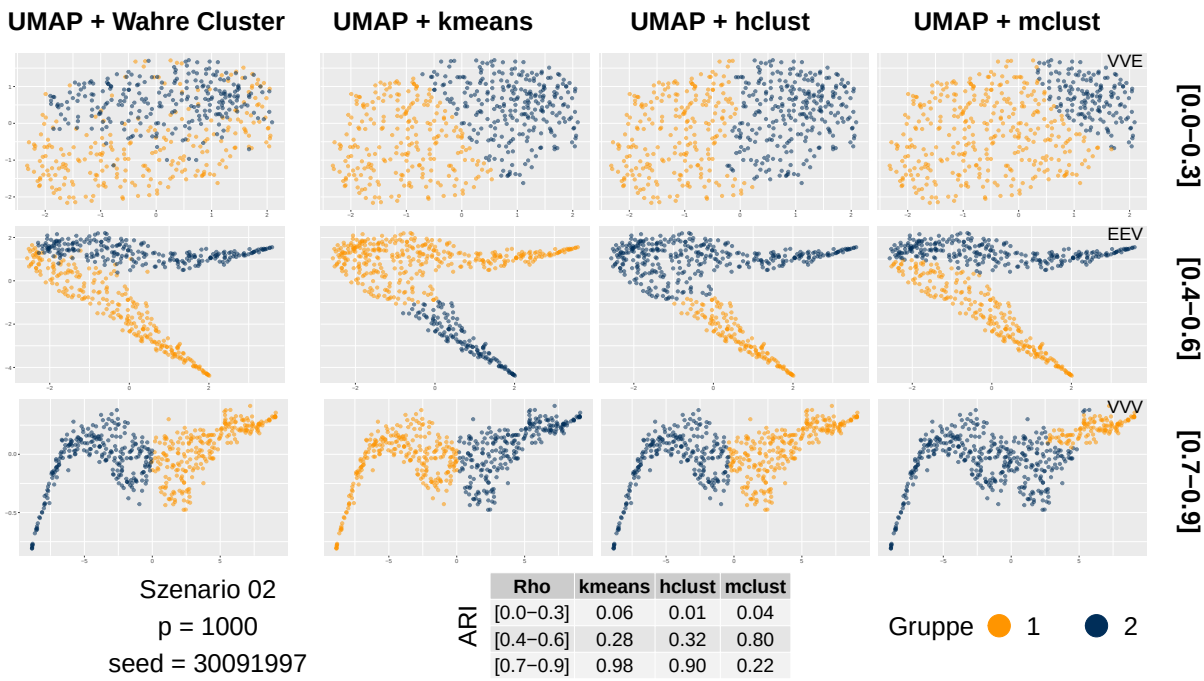
Rho	kmeans	hclust	mclust
[0.0–0.3]	0.24	0.09	0.38
[0.4–0.6]	0.00	0.01	0.66
[0.7–0.9]	0.89	0.49	0.98

Abbildung 17: Zuordnung und ARI, Szenario 02, p = 250, UMAP & Clustering



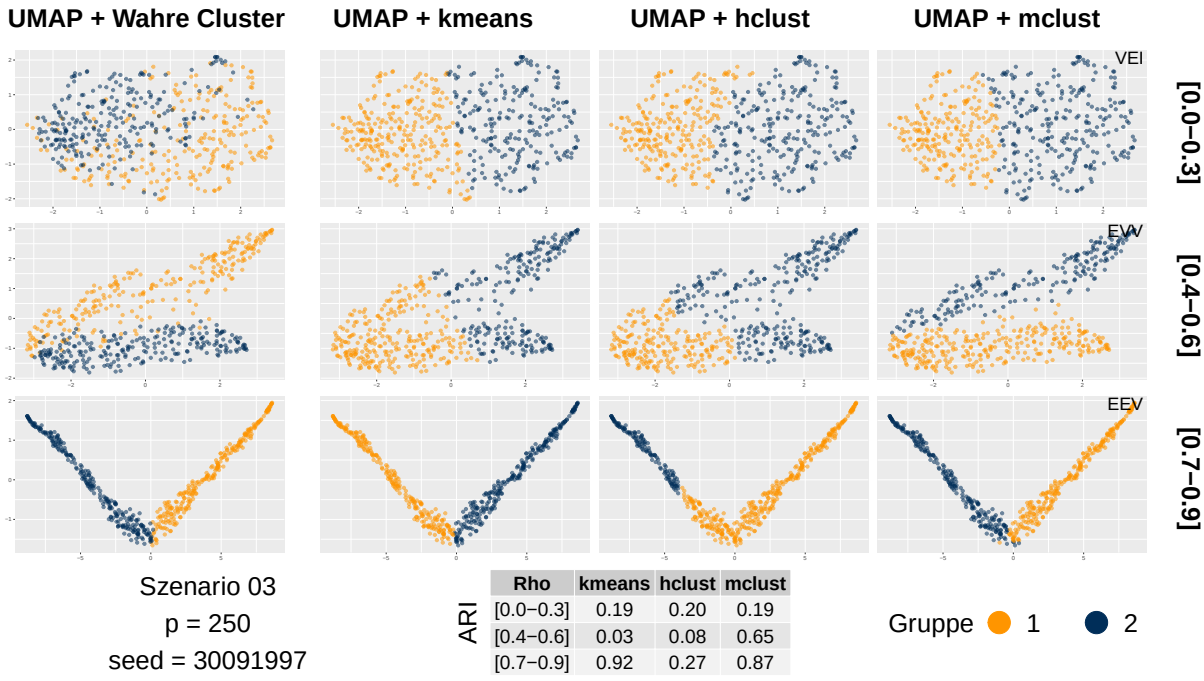
Rho	kmeans	hclust	mclust
[0.0–0.3]	0.04	0.03	0.03
[0.4–0.6]	0.04	0.00	0.60
[0.7–0.9]	0.81	0.32	0.95

Abbildung 18: Zuordnung und ARI, Szenario 02, p = 500, UMAP & Clustering



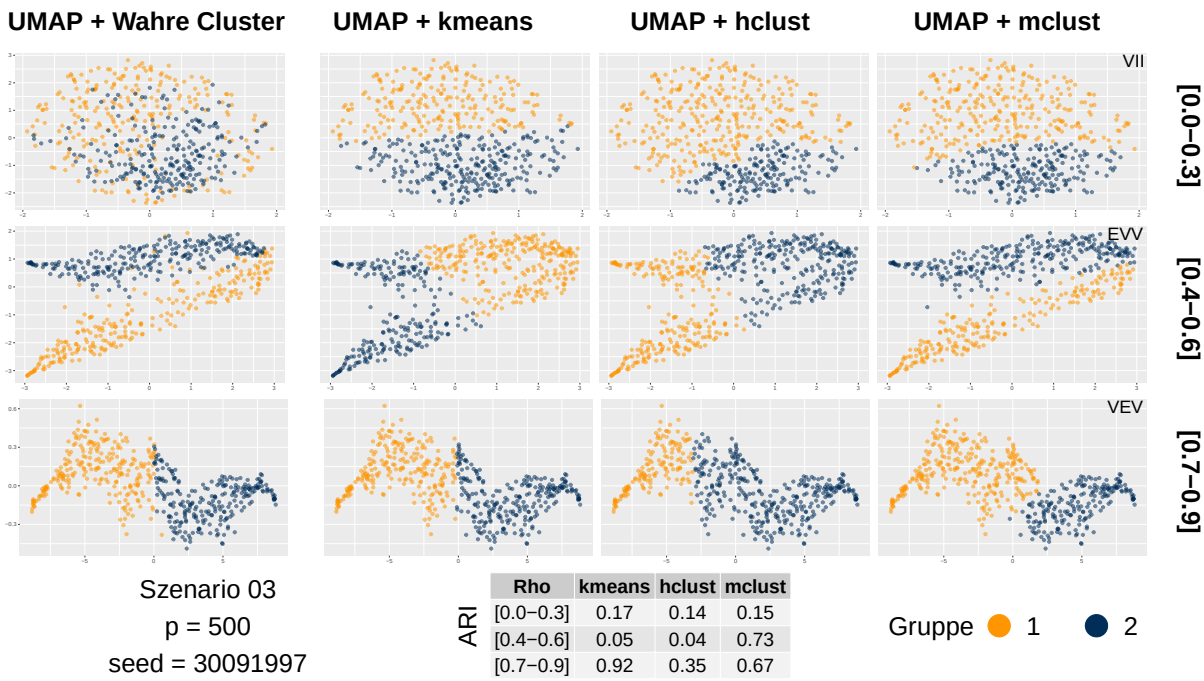
Rho	kmeans	hclust	mclust
[0.0−0.3]	0.06	0.01	0.04
[0.4−0.6]	0.28	0.32	0.80
[0.7−0.9]	0.98	0.90	0.22

Abbildung 19: Zuordnung und ARI, Szenario 02, p = 1000, UMAP & Clustering



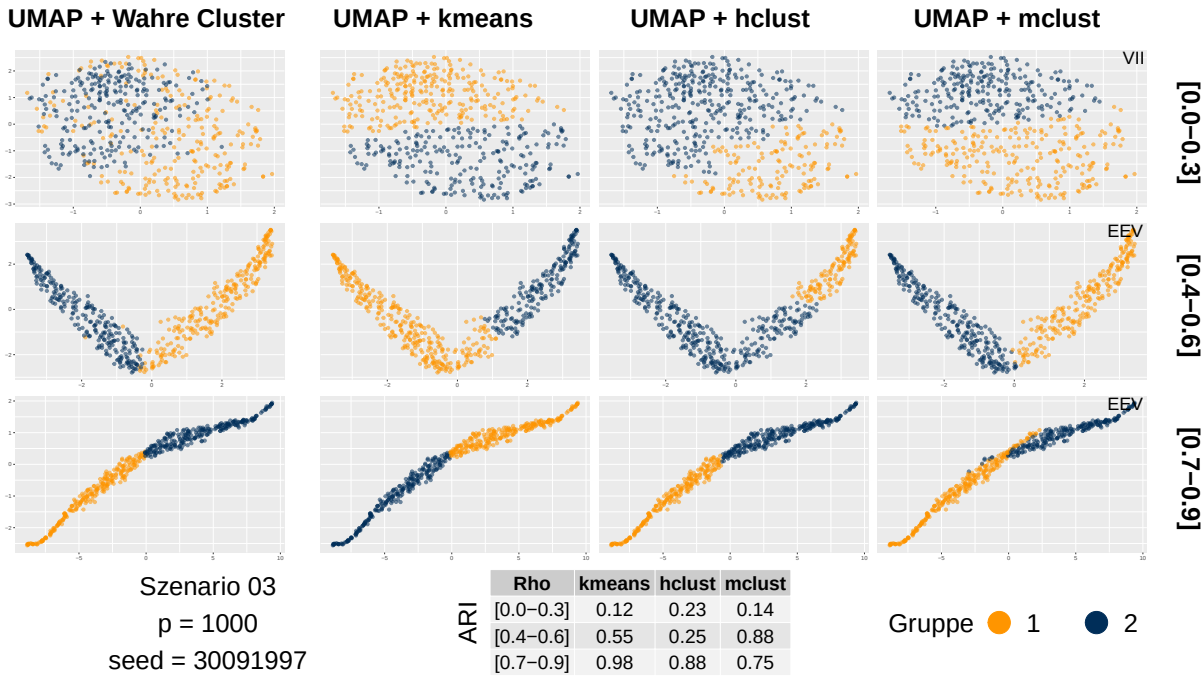
Rho	kmeans	hclust	mclust
[0.0−0.3]	0.19	0.20	0.19
[0.4−0.6]	0.03	0.08	0.65
[0.7−0.9]	0.92	0.27	0.87

Abbildung 20: Zuordnung und ARI, Szenario 03, p = 250, UMAP & Clustering



Rho	kmeans	hclust	mclust
[0.0–0.3]	0.17	0.14	0.15
[0.4–0.6]	0.05	0.04	0.73
[0.7–0.9]	0.92	0.35	0.67

Abbildung 21: Zuordnung und ARI, Szenario 03, p = 500, UMAP & Clustering



Rho	kmeans	hclust	mclust
[0.0–0.3]	0.12	0.23	0.14
[0.4–0.6]	0.55	0.25	0.88
[0.7–0.9]	0.98	0.88	0.75

Abbildung 22: Zuordnung und ARI, Szenario 03, p = 1000, UMAP & Clustering

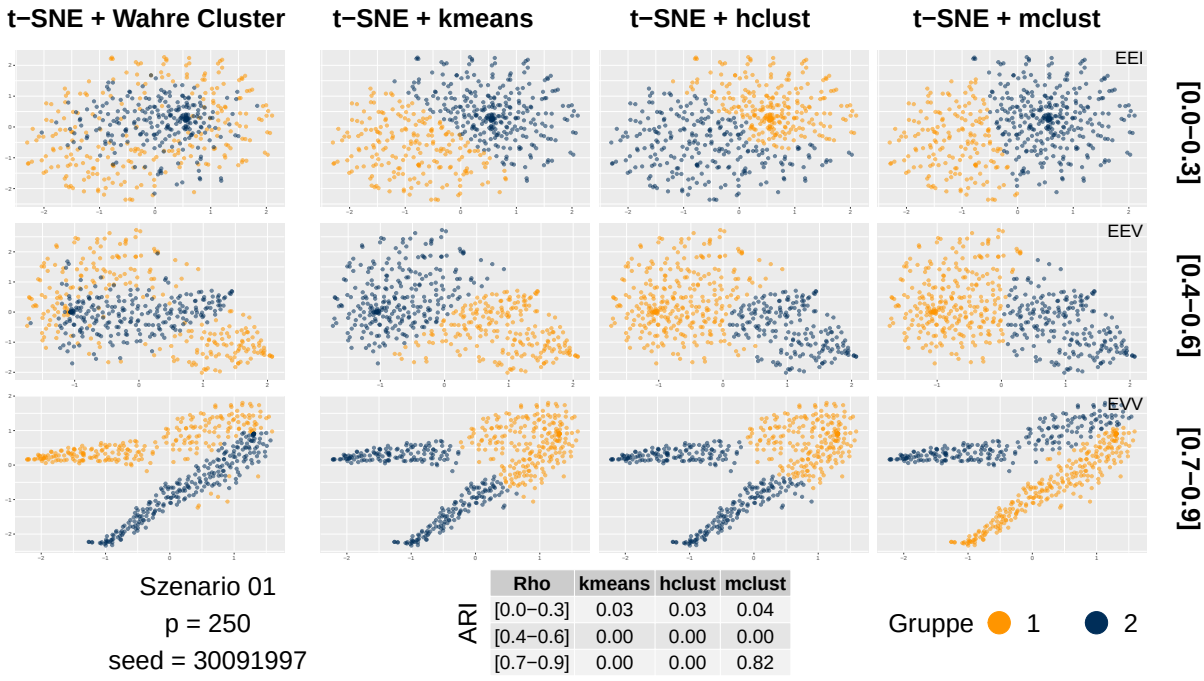


Abbildung 23: Zuordnung und ARI, Szenario 01, p = 250, t-SNE & Clustering

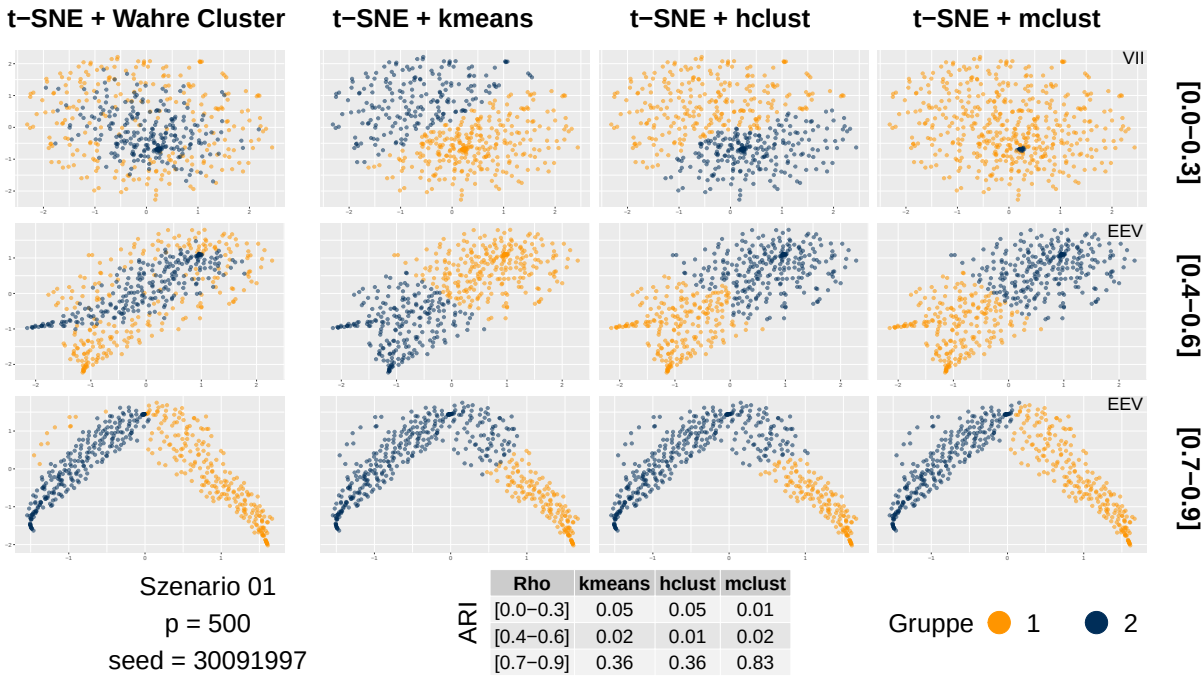


Abbildung 24: Zuordnung und ARI, Szenario 01, p = 500, t-SNE & Clustering

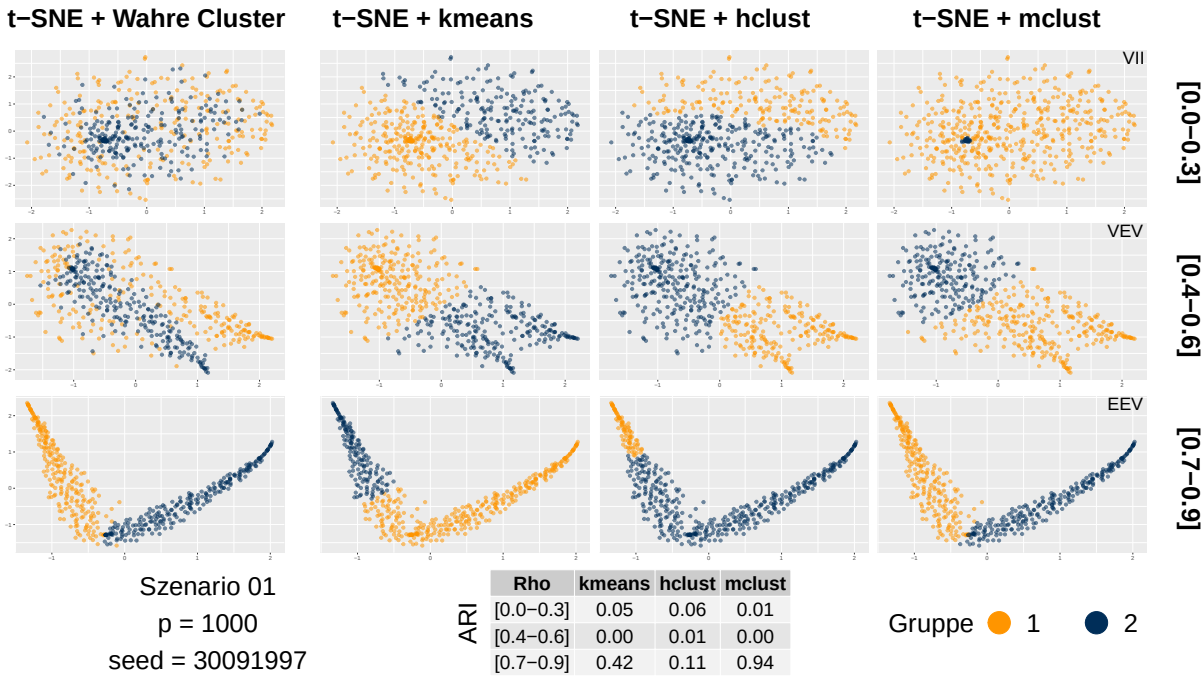


Abbildung 25: Zuordnung und ARI, Szenario 01, p = 1000, t-SNE & Clustering

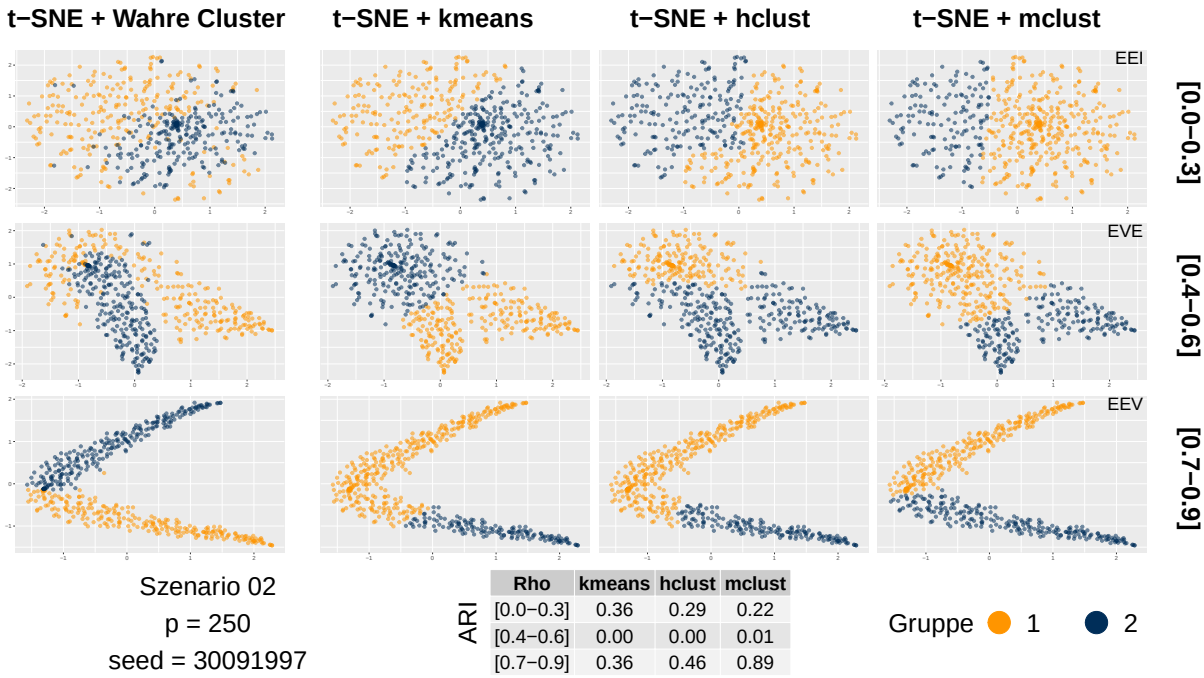


Abbildung 26: Zuordnung und ARI, Szenario 02, p = 250, t-SNE & Clustering

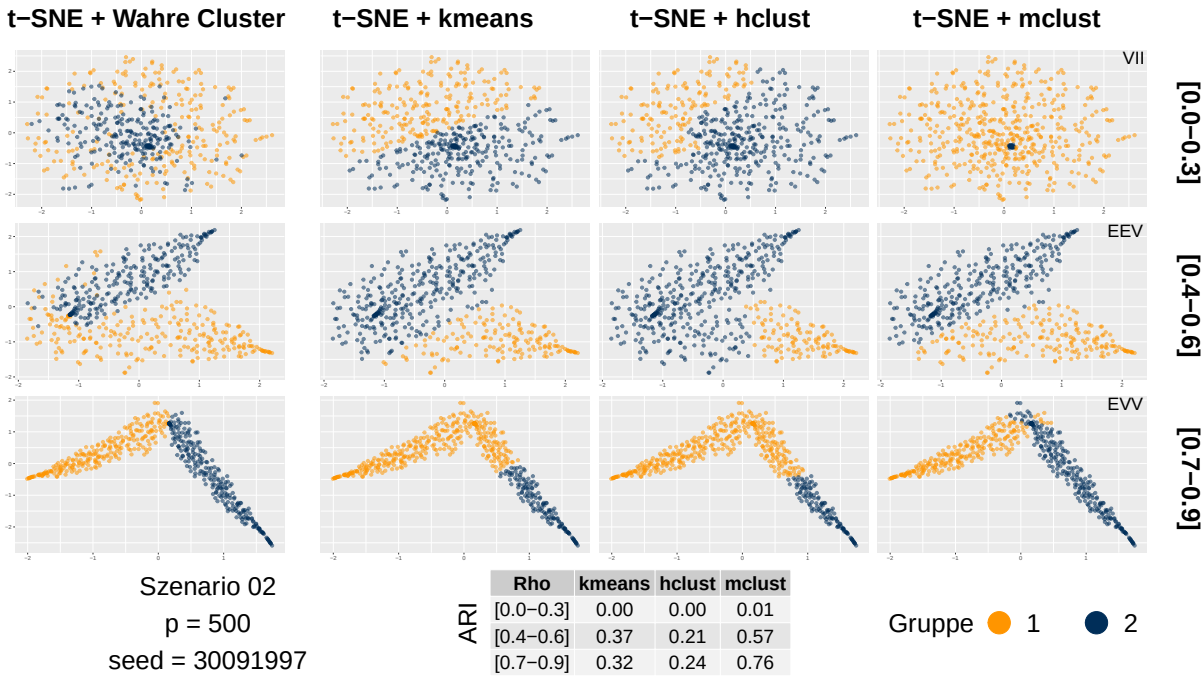


Abbildung 27: Zuordnung und ARI, Szenario 02, p = 500, t-SNE & Clustering

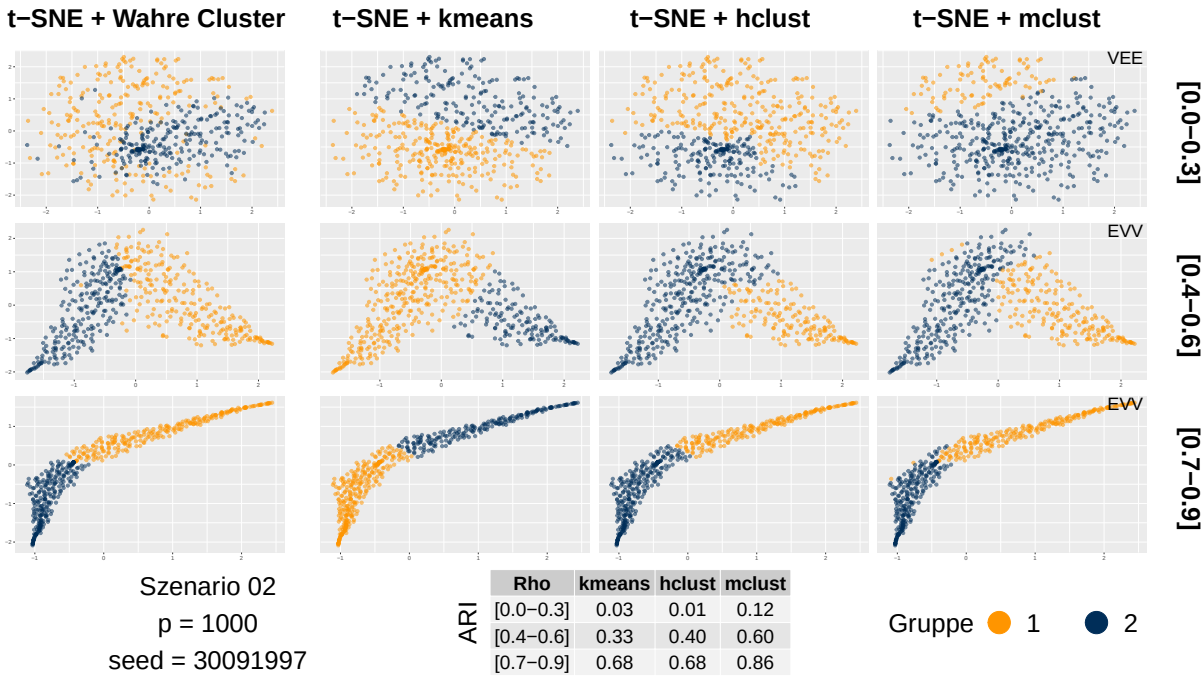


Abbildung 28: Zuordnung und ARI, Szenario 02, p = 1000, t-SNE & Clustering

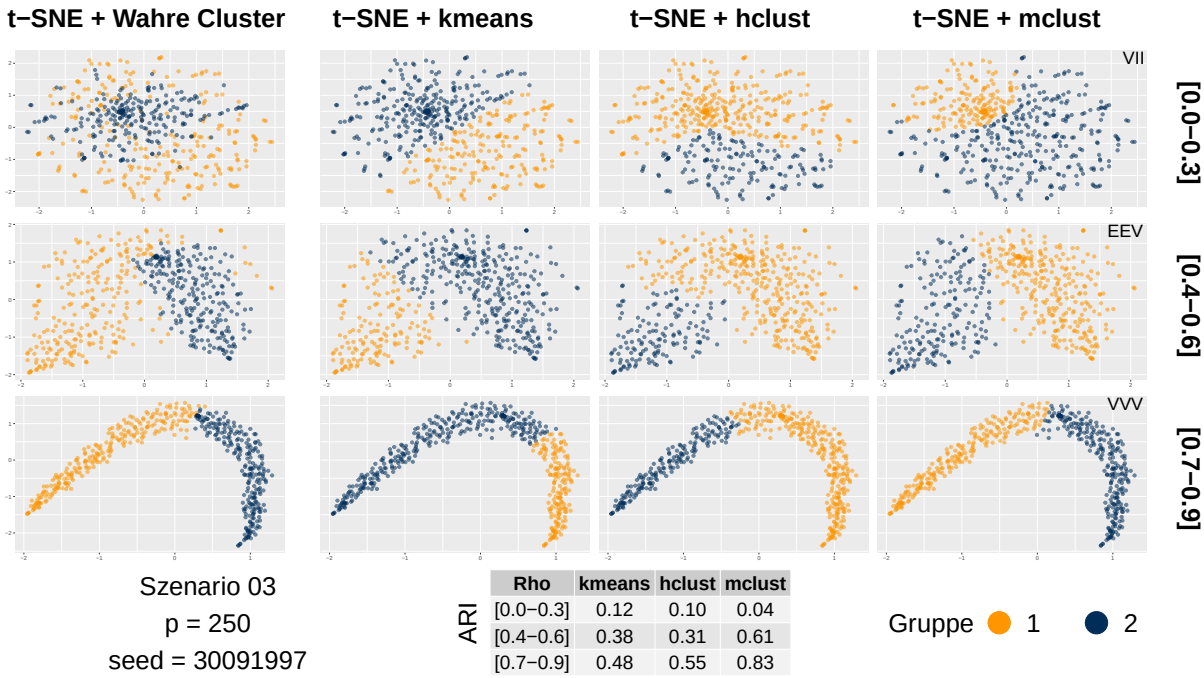


Abbildung 29: Zuordnung und ARI, Szenario 03, p = 250, t-SNE & Clustering

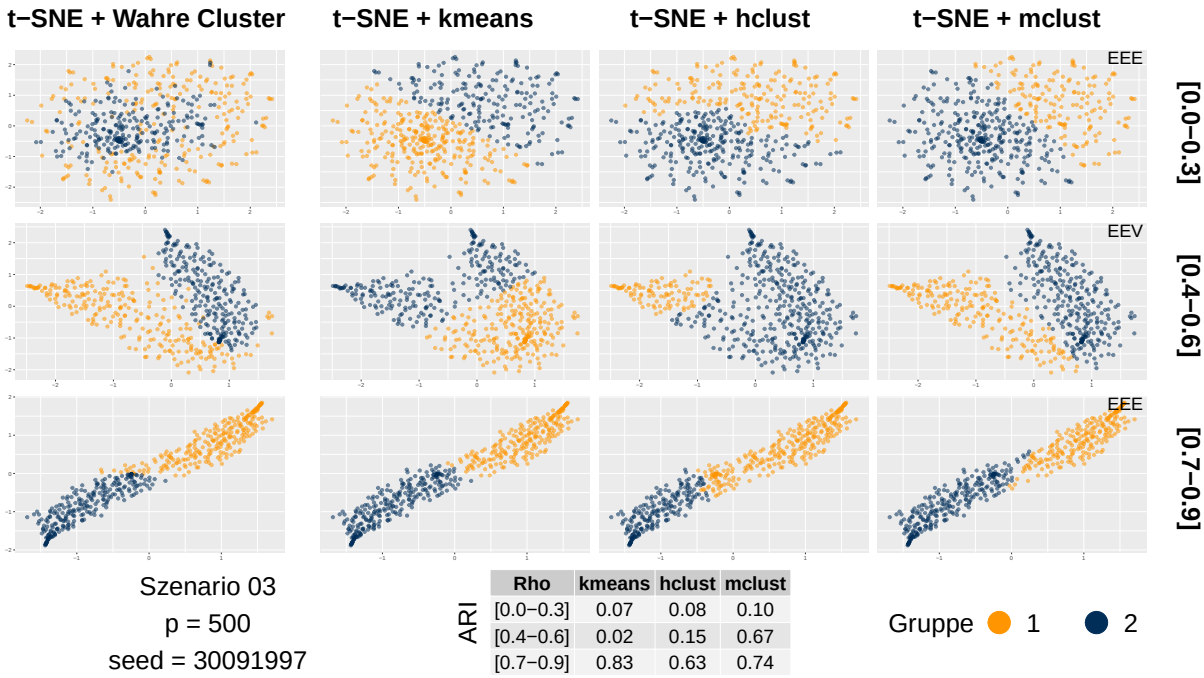


Abbildung 30: Zuordnung und ARI, Szenario 03, p = 500, t-SNE & Clustering

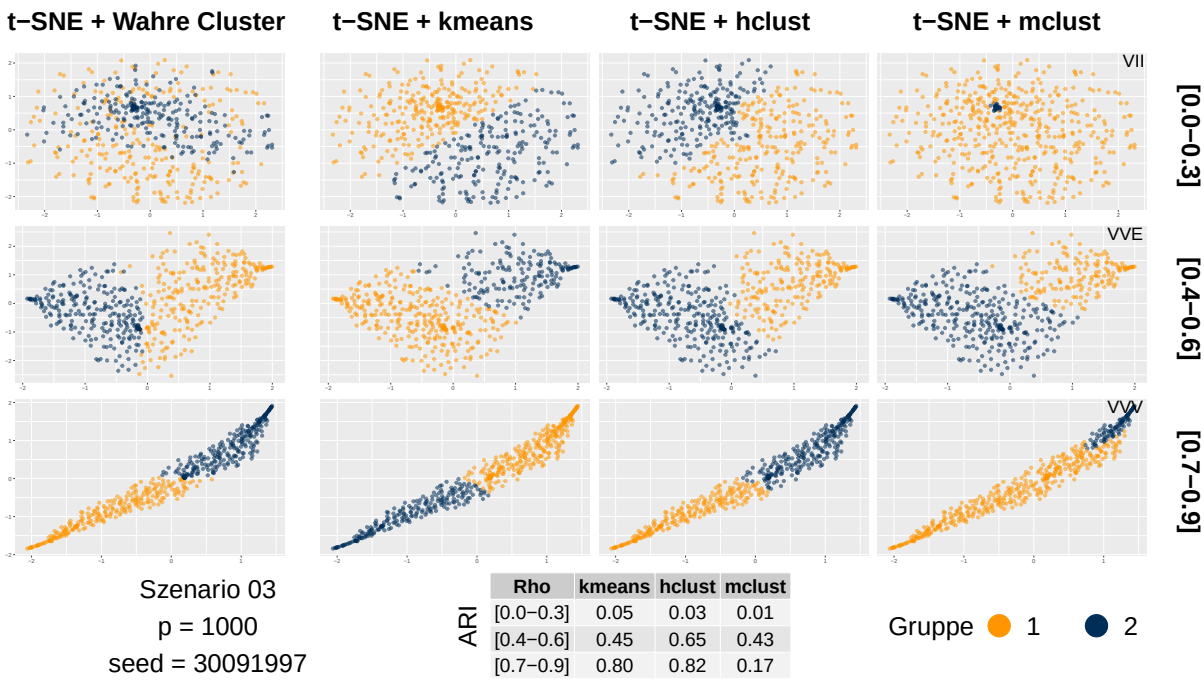


Abbildung 31: Zuordnung und ARI, Szenario 03, p = 1000, t-SNE & Clustering

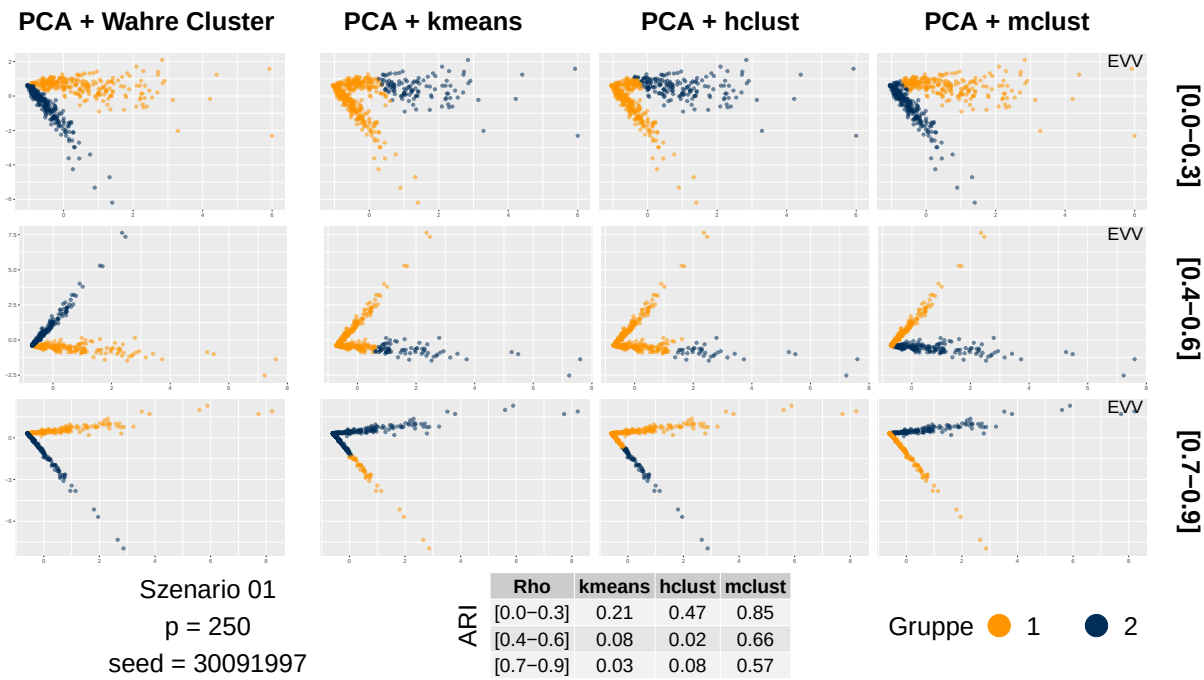


Abbildung 32: Zuordnung und ARI, Szenario 01, p = 250, PCA & Clustering

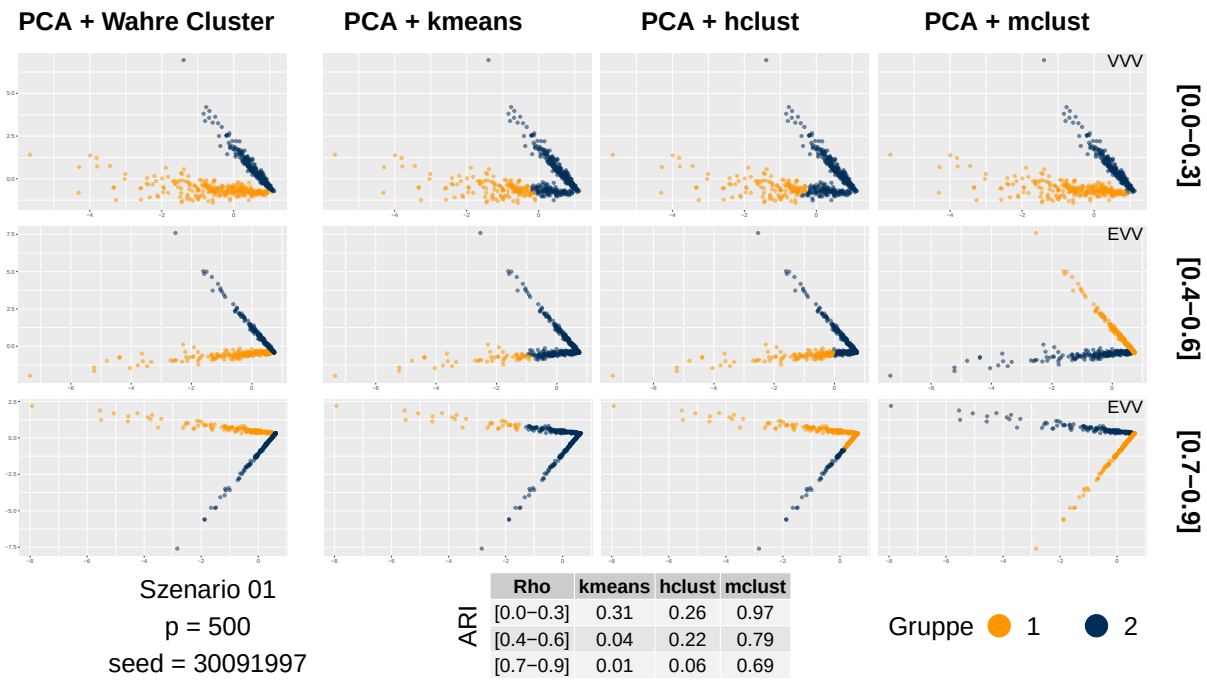


Abbildung 33: Zuordnung und ARI, Szenario 01, p = 500, PCA & Clustering

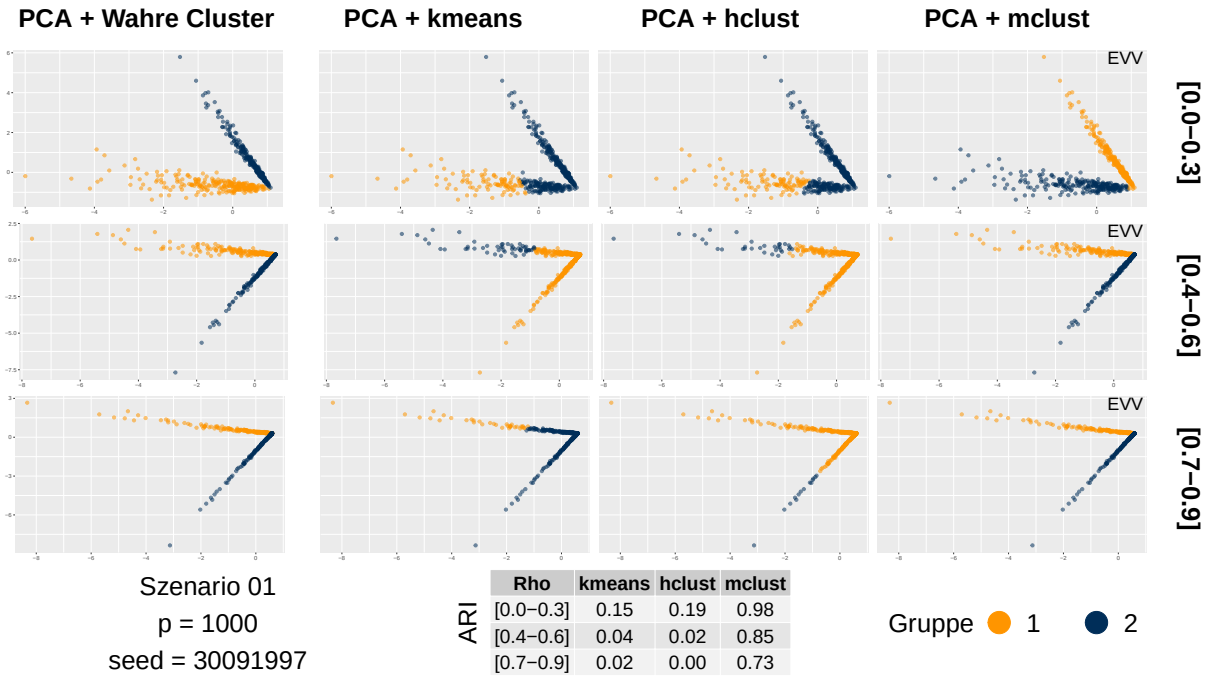


Abbildung 34: Zuordnung und ARI, Szenario 01, p = 1000, PCA & Clustering

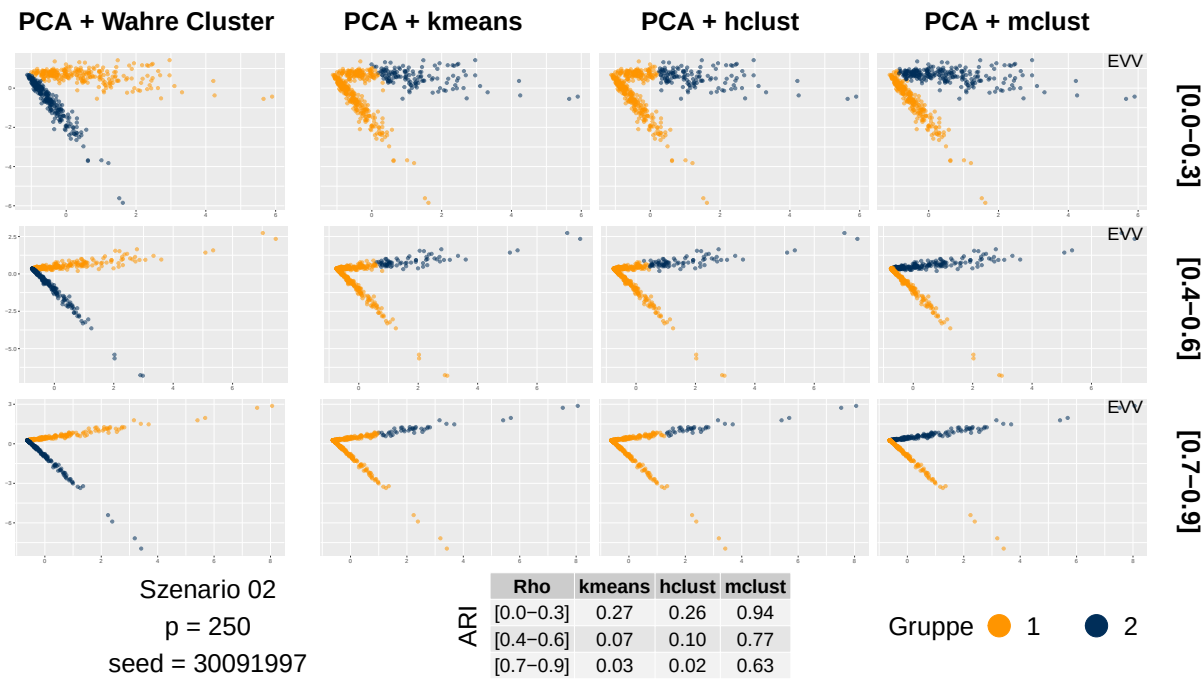


Abbildung 35: Zuordnung und ARI, Szenario 02, p = 250, PCA & Clustering

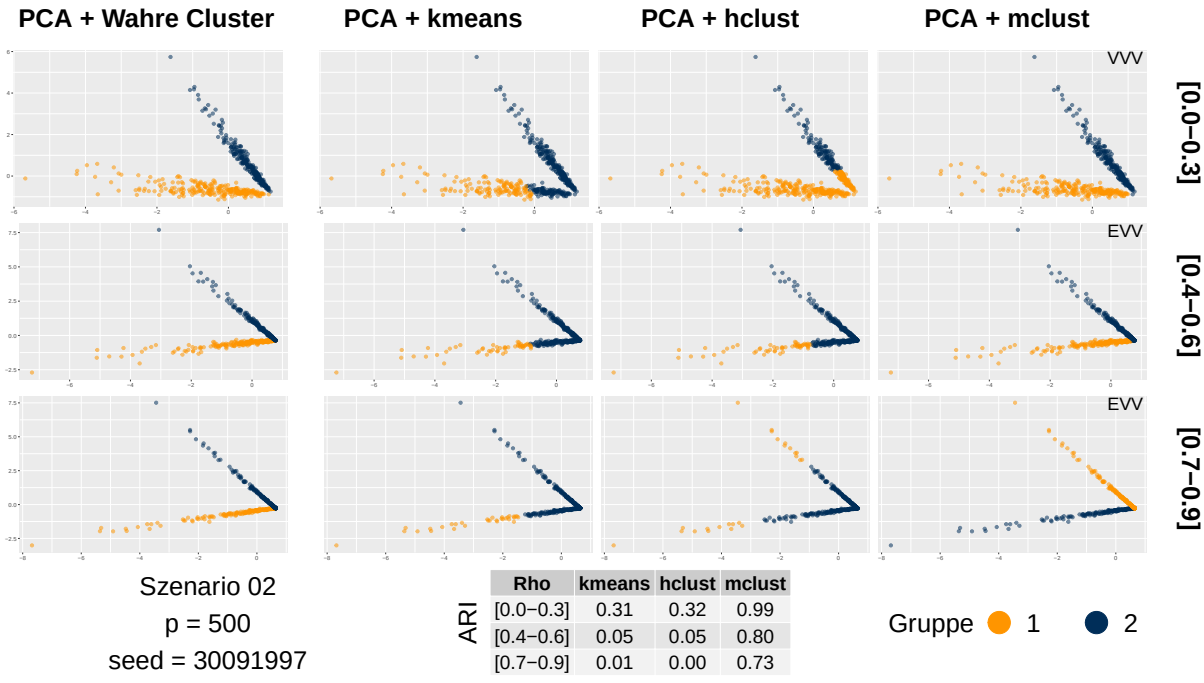


Abbildung 36: Zuordnung und ARI, Szenario 02, p = 500, PCA & Clustering

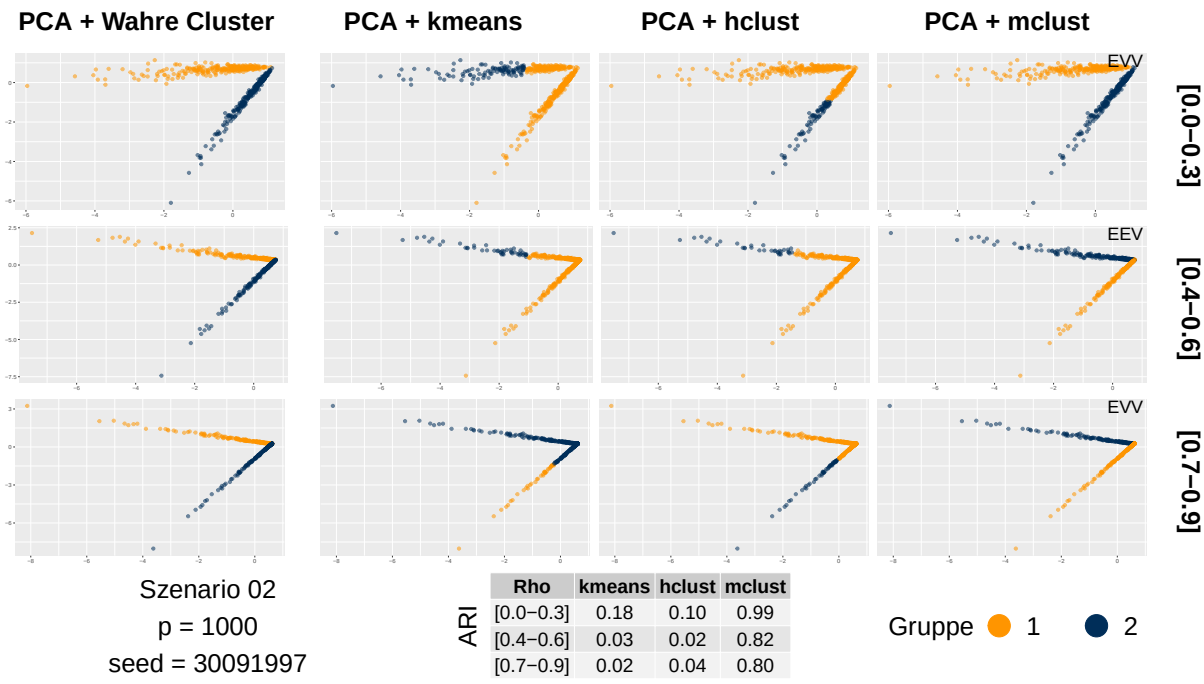


Abbildung 37: Zuordnung und ARI, Szenario 02, p = 1000, PCA & Clustering

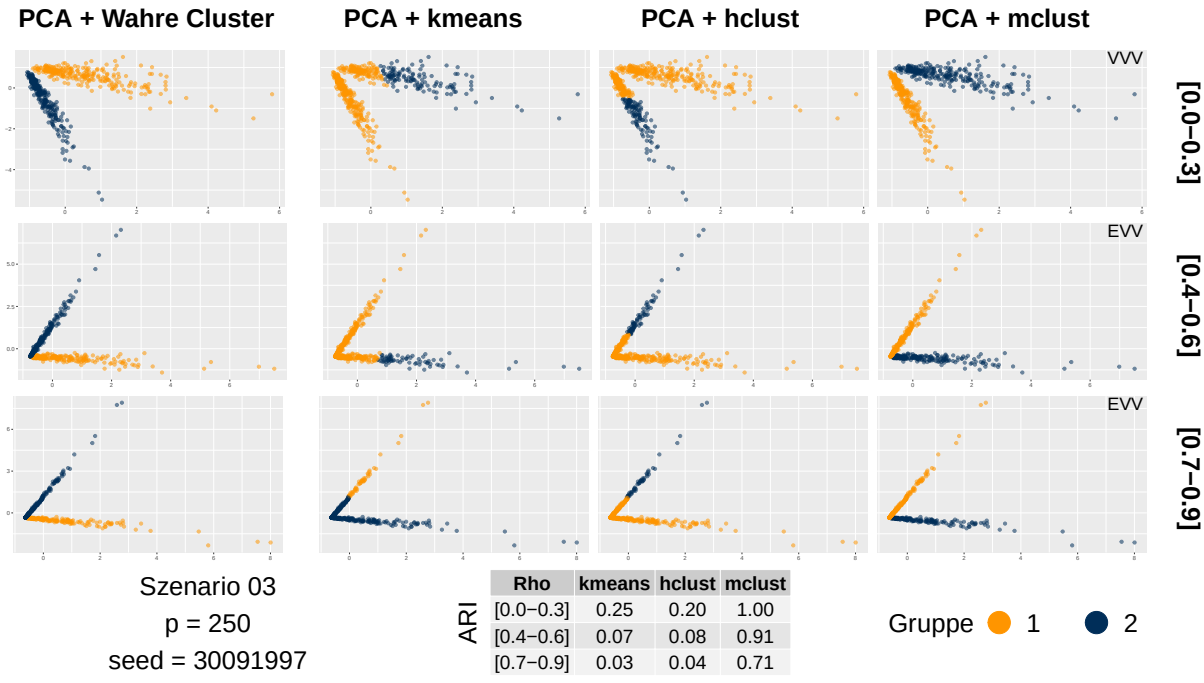


Abbildung 38: Zuordnung und ARI, Szenario 03, p = 250, PCA & Clustering

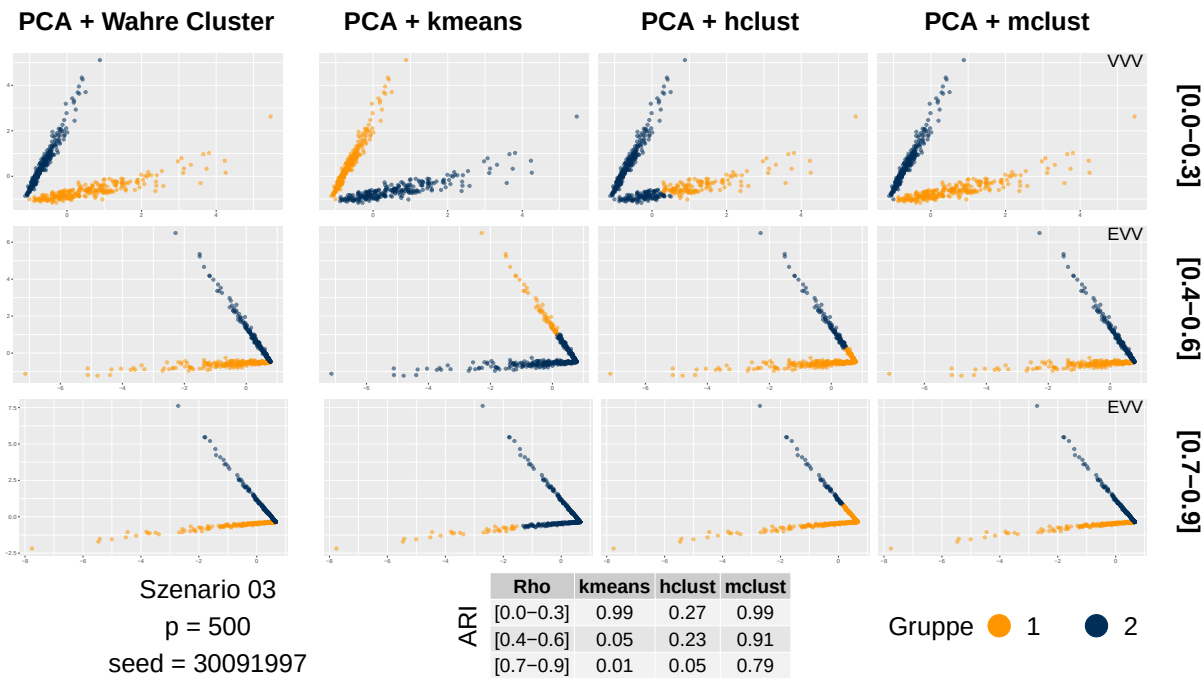


Abbildung 39: Zuordnung und ARI, Szenario 03, p = 500, PCA & Clustering

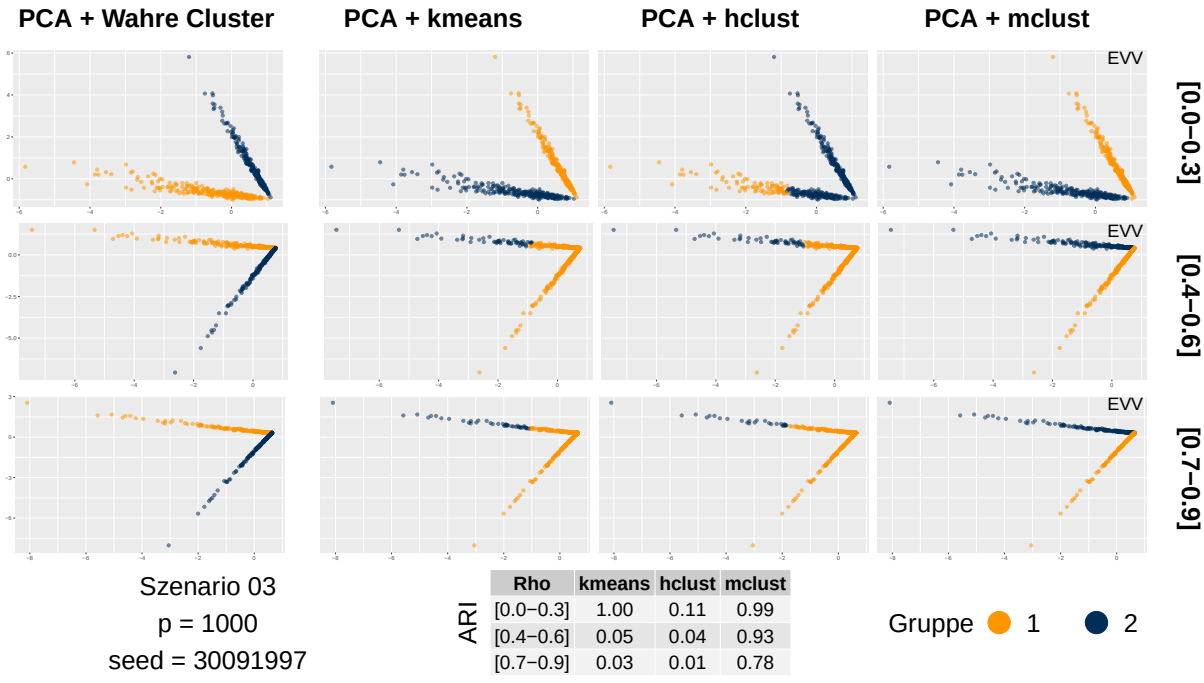


Abbildung 40: Zuordnung und ARI, Szenario 03, p = 1000, PCA & Clustering

C Elektronischer Anhang

Auf dem beiliegenden USB-Stick befinden sich alle verwendeten Datensätze, Korrelationsmatrizen, Korrelationsschranken sowie der gesamte R-Code mit dem diese erstellt wurden. Die Ergebnisse sind mit den entsprechenden Seeds reproduzierbar. Um sich besser zurechtzufinden ist hier die Ordner- und Datenstruktur angegeben.

```
R
├── data
│   │
│   ├── datasets                <- Datensätze inklusive Verteilungsparameter,
│   │                              Korrelationsmatrizen, Kovarianzmatrizen,
│   │                              Rechenzeit sowie dem Seed
│   ├── correlation_matrices    <- Korrelationsmatrizen
│   └── correlation_bounds      <- Obere und untere Schranken der Korrelation
│       └── quantiles           <- Berechnungen der Quantilsfunktion die für
│                                  die oberen und unteren Schranken verwendet
│                                  wurden*
│
├── ari                         <- Listen mit Ergebnissen der DR+CA
├── graphics                    <- Erzeugte Grafiken
│
├── execution_file.R            <- Hauptcode, der Funktionen aus den unteren
│                                  Dateien aufruft
│
├── data_generation.R           <- Funktionen zur Erstellung der Datensätze
├── dim_red.R                   <- Funktionen zur Dimensionsreduktion
├── clustering.R                <- Funktionen zur Clusteranalyse
├── correlations.R              <- Funktionen zur Visualisierung der
│                                  Korrelationen
├── auxillary.R                 <- Hilfsfunktionen, die in den oberen Datein
│                                  benötigt werden
└── session_info.txt            <- Info über die R-Version und geladene Pakete
```

* Diese sind in der Abgabe aufgrund der Größe (1.4 GB pro Gruppe, Szenario und Seed) und da diese nur für die Berechnung der Korrelationsschranken relevant sind nicht enthalten. Für die erneute Generation von Datensätzen wird dieser Ordner aber benötigt, weshalb er auch mit aufgeführt ist.

Folgende Seeds wurden zu Generation der Datensätze verwendet:

30091997	30092047
30092007	30092057
30092017	30092067
30092027	30092077
30092037	30092087

Die 10 Dimensionsreduktions- bzw. Clusteringergebnisse wurden dann mit dem Ausgangsseed und den 9 folgenden ganzen Zahlen als Seed berechnet.

Der `R`-Code ist ebenfalls unter https://github.com/fstermann/bthesis zu finden.

Abbildungsverzeichnis

Tabellenverzeichnis

Literatur

[1] Jörn Walter and Hannah Schickl. Einzelzellanalyse in forschung und medizin. 2019.

[2] Introduction to single-cell rna-seq. `https://hbctraining.github.io/scRNA-seq/lessons/01_intro_to_scRNA-seq.html`. Zuletzt besucht: 06.05.2021.

[3] Malte D Luecken and Fabian J Theis. Current best practices in single-cell rna-seq analysis: a tutorial. *Molecular systems biology*, 15(6):e8746, 2019.

[4] Richard E Bellman. *Adaptive control processes: a guided tour*. Princeton university press, 2015.

[5] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[6] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

[7] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

[8] James Melville. *uwot: The Uniform Manifold Approximation and Projection (UMAP) Method for Dimensionality Reduction*, 2020. URL `https://CRAN.R-project.org/package=uwot`. R package version 0.1.10.

[9] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[10] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[11] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901.

[12] Hervé Abdi and Lynne J Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[13] John A Hartigan and Manchek A Wong. Ak-means clustering algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):100–108, 1979.

[14] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. URL `https://www.R-project.org/`.

[16] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1): 86–97, 2012.

[17] Fionn Murtagh and Pierre Legendre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification*, 31(3): 274–295, 2014.

[18] Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association*, 97(458):611–631, 2002.

[19] Luca Scrucca, Michael Fop, T Brendan Murphy, and Adrian E Raftery. mclust 5: clustering, classification and density estimation using gaussian finite mixture models. *The R journal*, 8(1):289, 2016.

[20] Geoffrey J McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2007.

[21] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[22] Günter P Wagner, Koryu Kin, and Vincent J Lynch. A model based criterion for gene expression calls using rna-seq data. *Theory in Biosciences*, 132(3):159–164, 2013.

[23] Thomas W Yee et al. The vgam package for categorical data analysis. *Journal of Statistical Software*, 32(10):1–34, 2010.

[24] Achim Zeileis, Christian Kleiber, and Simon Jackman. Regression models for count data in r. *Journal of statistical software*, 27(8):1–25, 2008.

[25] Cornelia Fuetterer, Georg Schollmeyer, and Thomas Augustin. Constructing simulation data with dependency structure for unreliable single-cell rna-sequencing data using copulas. In *International Symposium on Imprecise Probabilities: Theories and Applications*, pages 216–224. PMLR, 2019.

[26] Aleksandra A Kolodziejczyk, Jong Kyoung Kim, Jason CH Tsang, Tomislav Ilicic, Johan Henriksson, Kedar N Natarajan, Alex C Tuck, Xuefei Gao, Marc Bühler, Pentao Liu, et al. Single cell rna-sequencing of pluripotent states unlocks modular transcriptional variation. *Cell stem cell*, 17(4):471–485, 2015.

[27] Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. *Regression*. Springer, 2007.

[28] Nicholas J Higham. Computing the nearest correlation matrix—a problem from finance. *IMA journal of Numerical Analysis*, 22(3):329–343, 2002.

[29] Douglas Bates and Martin Maechler. *Matrix: Sparse and Dense Matrix Classes and Methods*, 2021. URL `https://CRAN.R-project.org/package=Matrix`. R package version 1.3-2.

[30] Marne C Cario and Barry L Nelson. Modeling and generating random vectors with arbitrary marginal distributions and correlation matrix. Technical report, Citeseer, 1997.

[31] Inbal Yahav and Galit Shmueli. On generating multivariate poisson data in management science applications. *Applied Stochastic Models in Business and Industry*, 28(1):91–102, 2012.

[32] Alessandro Barbiero and Pier Alda Ferrari. Simulation of correlated poisson variables. *Applied Stochastic Models in Business and Industry*, 31(5):669–680, 2015.

[33] Ward Whitt et al. Bivariate distributions with given marginals. *The Annals of statistics*, 4(6):1280–1289, 1976.

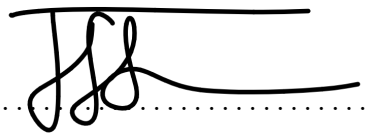
[34] Allison Fialkowski and Hemant Tiwari. Simcorrmix: Simulation of correlated data with multiple variable types including continuous and count mixture distributions. *R J.*, 11 (1):250, 2019.

[35] Microsoft Corporation and Stephen Weston. *doSNOW: Foreach Parallel Adaptor for the 'snow' Package*, 2020. URL `https://CRAN.R-project.org/package=doSNOW`. R package version 1.0.19.

[36] Özge Sahin and Claudia Czado. Vine copula mixture models and clustering for non-gaussian data. *arXiv preprint arXiv:2102.03257*, 2021.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

München, den 11.05.2021 

Fabian Stermann