Methoden

Volker Willert* und Matthias Schnaubelt

Die Regelung von Daten: Eine Idee zur Clusteranalyse von vernetzten Datenbeständen

Data control: a feedback control design for clustering of networked data

DOI 10.1515/auto-2016-0059
Eingang 1. April 2016; angenommen 8. Juli 2016

Zusammenfassung: Der Beitrag befasst sich mit der Frage, wie die Clusteranalyse von dezentral abgelegten vernetzten Datenbeständen sowohl im Bezug auf die Konvergenzgeschwindigkeit als auch auf die zu erreichende Güte der Datenzerlegung verbessert werden kann. Dazu wird die Idee der Regelung von Daten über einen Datenregler vorgestellt. Für das K-means Clusteringverfahren wird ein beweisbar konvergenter Datenregler entworfen und anhand eines umfangreichen Benchmarks evaluiert. Des Weiteren wird eine Matrix zur Beschreibung von Zugehörigkeitsübergängen von Clusterdynamiken eingeführt und die Verwandtschaft des Datenreglers zu Kernelmethoden aufgezeigt. Außerdem wird die Beziehung zwischen dem Datenregler und Consensusdynamiken für Multi-Agenten-Systeme hergestellt. Damit ist das vorgestellte Verfahren verteilt implementierbar und auf große dezentral abgelegte Datenmengen anwendbar.

Schlüsselwörter: Datenanalyse, Clustering, verteilte Regelung, Multi-Agenten-Systeme.

Abstract: This paper deals with the question how clustering of decentrally stored and networked data can be improved in matters of convergence speed and the clustering performance via the influence of data points using a new data controller. For the K-means clustering algorithm a provably convergent data controller is designed and evaluated on a comprehensive benchmark. Further on, a matrix to describe assignment changes along the clustering iterations is introduced and the affinity of controlling data and kernel methods is shown. In addition, the relation between the designed data controller and the consensus protocol for multi-agent-systems is presented. This shows, that the proposed method can be implemented distributively and be applied to decentrally stored big data.

Keywords: Data analytics, clustering, distributed control, multi-agent-systems.

1 Einleitung

Die Umsetzung der Vision von Industrie 4.0 hält laut einem Expertengespräch im Rahmen der Plattform Industrie 4.0, abgehalten auf der Hannover Messe im Jahr 2015 und auszugsweise abgedruckt in der Zeitschrift Elektrotechnik und Automation [1], jede Menge Herausforderungen an die technische Realisierung bereit. Ein großes Potential bezüglich der Automation von Prozessen wird hierbei im Austausch von Messdaten unterschiedlicher örtlich verteilter Sensoren über ein Netzwerk gesehen. Messdaten von einem Sensor an einem bestimmten Ort im Produktionsprozess können beispielsweise nicht mehr für nur eine Steuereinheit, die zuständig für nur einen Prozess am gleichen Ort ist benutzt werden, sondern im Prinzip überall für jede beliebige Steuereinheit nutzbar gemacht werden. Des Weiteren können zur Automatisierung nicht nur die rohen Messdaten, sondern auch daraus abgeleitete (abstraktere) Informationen von Nutzen sein. Schließlich beinhaltet die sogenannte *Cloud* auch Metadaten, welche über das Netzwerk erreichbar sind und in die Prozessleittechnik integriert werden können.

Werden diese Aussagen aus der Expertenrunde gegenüber gestellt, dann ermöglicht das Netzwerk zum einen das Erzeugen einer Hierarchie unterschiedlicher Informationen mit unterschiedlichem Abstraktionsgrad und damit Informationsgehalt basierend auf örtlich und zeitlich verteilten Messdaten und zum anderen den Zugriff auf diese Informationen, im Idealfall an jedem Ort zu jeder Zeit, beispielsweise über mobile Webanwendungen [2].

***Korrespondenzautor: Volker Willert,** Technische Universität Darmstadt, Institut für Automatisierungstechnik und Mechatronik, Fachgebiet Regelungsmethoden und Robotik, Landgraf-Georg-Str. 4, 64283 Darmstadt, E-Mail: vwillert@rtr.tu-darmstadt.de
Matthias Schnaubelt: Rudolph-Bultmann-Str. 9, 35039 Marburg

Wenn man unter diesem Gesichtspunkt an eine Realisierung denkt, dann tauchen unter anderem zwei Fragen auf:

Erstens, welche relevante Information bzw. Bedeutung tragen die Daten und wie kann man diese Information aus den Daten extrahieren?

Zweitens, welche der vielen Daten im Netzwerk liefern überhaupt relevante Informationen für die Automatisierungsaufgabe, die an einem bestimmten Ort zu einer bestimmten Zeit gerade ausgeführt werden soll?

Das Extrahieren von relevanter Information aus Datenbeständen entspricht einer klassischen Mustererkennungsaufgabe. Diese Aufgabe kann unter anderem mit einer Clusteranalyse gelöst werden. Clusteranalyseverfahren gehören zur Kategorie des unüberwachten Lernens [3]. Die Clusteranalyse zielt darauf ab gewisse Strukturen in Daten zu finden, oder präziser Gruppierungen von Mustern, Punkten oder Objekten zu entdecken [4]. Sie wird dazu benutzt, um Einsichten in die Struktur von Daten zu erhalten, wie beispielsweise die Repräsentation von Information in der Genexpression [5] und wird in vielen Anwendungen benötigt, wie zum Beispiel der Detektion und Isolation von Ereignissen und Anomalien für die industrielle Prozesssteuerung oder der Generierung von Hypothesen und Vorhersagen aus Wetter-, Verkehrs-, Finanz- oder Wirtschaftsdaten [6]. Dazu wird eine kleine Anzahl an Repräsentativen, sogenannten Prototypen (Referenzvektoren) aus Datenbeständen extrahiert, indem der Grad an Ähnlichkeit zwischen den Daten identifiziert wird. Das wiederum kann weiterführend zur Klassifikation von Daten, zur Datenkompression und zum Codebuchlernen verwendet werden [7].

1.1 Herausforderungen an die Clusteranalyse vernetzter Datenmengen

Gängige Algorithmen der Clusteranalyse setzen voraus, dass zu jeder Zeit ein Zugriff auf alle Daten möglich ist und die Berechnung der Cluster und der Clusterzugehörigkeit zentral an einem Ort stattfinden kann. Sollen diese Algorithmen nun auf eine sehr große Menge an Daten angewendet werden, wobei der Zugriff auf die Daten über ein Netzwerk stattfindet, dann ändert sich nichts an der Aufgabe der Clusteranalyse, aber an den Randbedingungen denen ein Algorithmus zur Umsetzung der Datenzerlegung ausgesetzt ist. Das Datenvolumen nimmt zu und es können Unterschiede in Dimensionalität und Wertebereich zwischen den Daten vorliegen. Außerdem kann der Zugriff auf die Daten nicht mehr synchron erfolgen, wenn die Clusteranalyse nicht mehr zentral, sondern dezentral auf vielen Berechnungseinheiten verteilt ausgeführt wird.

In der Literatur finden sich schon einige Lösungsvorschläge zur verteilten, dezentralen Implementierung von Clusteralgorithmen [8–10] und der Berücksichtigung von sehr großen Datenmengen mit unterschiedlich dimensionalen Datenräumen und Wertebereichen [11–13]. Alle diese Lösungen haben eine starke Zunahme der Laufzeit der Clusteralgorithmen zur Folge. Jain formuliert deswegen als dringlichstes Ziel momentaner Zerlegungsalgorithmen effizientere, schnell berechenbare Lösungen zu finden [4]. Viele Ansätze beschäftigen sich daher mit approximativen Methoden, die bei geringerem Rechenaufwand dennoch eine Lösung mit ausreichender Güte der Zerlegung finden [14]. Diese Klasse von Algorithmen sind meist als Optimierungsproblem formuliert und werden iterativ gelöst, wobei Rechenersparnis zum einen über approximative Gradientenverfahren oder Abschätzungen von Distanzmaßen [15] erreicht wird und zum anderen über das Vernachlässigen von Rechenschritten, indem die Auswirkung der Rechenschritte auf die Veränderung der momentanen Zerlegungsgüte bewertet wird [16].

Man könnte auch einen komplementären Gedanken verfolgen, der zur Rechenersparnis keine *approximativen Rechenschritte* auf Basis von *exakten Daten* heranzieht, sondern dazu *approximative Daten* mit *exakten Rechenschritten* verarbeitet. Dieser Gedanke ist vor allem bei großen Datenvolumina einleuchtend, da ein beliebiges einzelnes Datum sowie der exakte Wert dieses Datums mit hoher Wahrscheinlichkeit umso weniger relevant für das Ergebnis der Zerlegung wird, je mehr das Datenvolumen zunimmt. In [17] formuliert Mézard eine weitere Herausforderung, um die Rechenersparnis bei gleichbleibender Zerlegungsgüte weiter voran zu treiben:

The main open challenge for the improvement of such algorithms is to understand the limits of these clustering algorithms. At the lowest level this means controlling the convergence properties or the quality of the approximate solutions that they find.

Unter Berücksichtigung dieser Aussagen, wird in diesem Artikel ein alternatives Verfahren vorgestellt, das Einfluss auf die Konvergenzeigenschaften von Zerlegungsalgorithmen nimmt, jedoch nicht über eine bestimmte Approximation von Rechenschritten, sondern über die gezielte Veränderung der Datenwerte in Abhängigkeit von der momentanen Zerlegungsgüte. Da jeder einzelne Datenpunkt unabhängig von allen anderen Daten verändert wird, ist dieses Verfahren komplett verteilt implementierbar und somit geeignet für die Verarbeitung von dezentral abgelegten Datenbeständen.

1.2 Idee der Datenregelung

Jede Datenzerlegung basiert auf einer endlichen Anzahl N an D-dimensionalen, meist reellwertigen Daten $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T \in \mathbb{R}^{N\times D}$. Bei der Modellierung gehen fast alle partitionierenden Clusterverfahren von einem linearen Zusammenhang zwischen den Daten und einer festen Anzahl an K Prototypen $\mathbf{M} = [\mathbf{m}_1, \ldots, \mathbf{m}_K]^T \in \mathbb{R}^{K\times D}$ aus, wobei jedes Datum $\mathbf{x}_n$ über eine Linearkombination von Zugehörigkeiten r_{kn} der Zugehörigkeitsmatrix $\mathbf{R} \in \mathbb{R}^{N\times K}$ und Prototypen $\mathbf{m}_k$ rekonstruiert werden kann:

$$\mathbf{x}_n \approx \sum_{k=1}^{n} r_{kn}\mathbf{m}_k\,, \quad \text{bzw.} \quad \mathbf{X} \approx \mathbf{RM}\,. \tag{1}$$

Hierbei wird angenommen, dass die Anzahl der Prototypen wesentlich kleiner als die Anzahl der Daten ist $K \ll N$. Solche Ansätze, welche direkt oder indirekt die Verteilung und deren Zerlegung modellieren, werden *generative Modelle* genannt, weil es mit solchen Modellen möglich ist neue synthetische Datenpunkte zu *generieren* [15].

Ziel der Datenzerlegung ist es sowohl einen Satz an Prototypen $\mathbf{M}^\star$ als auch Zugehörigkeiten $\mathbf{R}^\star$ für alle Daten $\mathbf{X}$ zu finden, so dass die Rekonstruktion der Daten über die Prototypen und die Zugehörigkeiten so gut wie möglich unter Berücksichtigung des Modells (1) wird. Dies wird in einem Optimierungsproblem formuliert

$$\begin{aligned}(\mathbf{M}^\star, \mathbf{R}^\star) &= \operatorname{argmin}_{\mathbf{R},\mathbf{M}} J(\mathbf{X}, \mathbf{R}, \mathbf{M})\,, \\ &\text{mit} \quad J(\mathbf{X}, \mathbf{R}, \mathbf{M}) = \|\mathbf{X} - \mathbf{RM}\|\,, \\ &\text{s.t.} \quad \text{Nebenbedingungen}(\mathbf{R}, \mathbf{M})\,,\end{aligned} \tag{2}$$

wobei $\|\cdot\|$ einer beliebigen Matrixnorm entspricht. Die einzelnen Zerlegungsmethoden unterscheiden sich dann lediglich in der Art der Norm und den Nebenbedingungen, die zusätzlich an $\mathbf{R}$ und $\mathbf{M}$ gestellt werden. Nebenbedingungen können sowohl an die Werte der Zugehörigkeiten und Prototypen (beispielsweise Nichtnegativität) als auch an die Eigenschaften der Zugehörigkeiten und Muster untereinander (beispielsweise Spärlichkeit oder Orthogonalität) gestellt werden. Verfahren, die dieser Klasse von Problemen zugeordnet werden können sind unter vielen anderen die *Principal Component Analysis*, die *Independent Component Analysis*, die *Nonnegative Matrix Factorization* und das *K-means Clustering* [15, 18].

Solche Optimierungsprobleme können oder müssen sogar in den meisten Fällen iterativ gelöst werden. Konvergente iterative Lösungsmethoden können als stabiles zeitdiskretes (meist) nichtlineares dynamisches System aufgefasst werden, wobei die Zeit über den einzelnen Iterationsschritten t diskretisiert wird, der Systemeingang den Daten $\mathbf{X}$ entspricht und der Ausgang den Prototypen $\mathbf{M}^t$ und der Zerlegung $\mathbf{R}^t$ zum jeweiligen Iterationsschritt t.

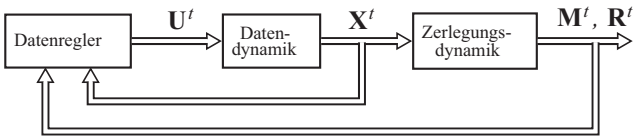

Abbildung 1: Regelkreis der Datenzerlegung.

Die Idee der Datenregelung geht von solch einer Zerlegungsdynamik aus und möchte die Daten als Eingang des Systems nun so verändern, dass sich die Daten dem System günstiger für eine Zerlegung präsentieren, so dass die Zerlegungsdynamik schneller konvergiert und im besten Fall auch die Zerlegungsgüte der stationären Lösung $(\mathbf{M}^\star, \mathbf{R}^\star)$ zunimmt. Es wird also angenommen, die Daten $\mathbf{X}^t$ können sich virtuell im Datenraum bewegen und werden iterationsabhängig. Die Frage für einen Reglerentwurf ist nun:

Wie müssen sich die Daten im Datenraum bewegen, damit sie der Zerlegungsalgorithmus schneller und besser zerlegen kann?

Dazu muss eine geeignete Eigendynamik der Daten (Datendynamik) angenommen werden und ein Datenregler entworfen werden, der die Daten nur so bewegt, dass die Struktur und damit die den Daten innewohnende Information über die Zusammengehörigkeit nicht verloren geht. In Abbildung 1 ist das Prinzip der Datenregelung als Regelkreis skizziert. In dieser Arbeit wird diese Idee über eine iterative Lösung folgender Erweiterung des Optimierungsproblems (2) realisiert:

$$\begin{aligned}(\mathbf{X}^\star, \mathbf{M}^\star, \mathbf{R}^\star) &= \operatorname{argmin}_{\mathbf{X},\mathbf{M},\mathbf{R}} J(\mathbf{X}, \mathbf{M}, \mathbf{R})\,, \\ &\text{mit} \quad J(\mathbf{X}, \mathbf{R}, \mathbf{M}) = \|\mathbf{X} - \mathbf{RM}\|\,, \\ &\text{s.t.} \quad \text{Nebenbedingungen}(\mathbf{X}, \mathbf{R}, \mathbf{M})\,.\end{aligned} \tag{3}$$

Damit reduziert sich das Problem auf das Finden einer strukturerhaltenden Nebenbedingung($\mathbf{X}$) für den Bewegungsraum der Daten.

2 Datenregler für die K-means Clusteranalyse

Für einen ersten Entwurf eines Datenreglers betrachten wir eine sehr einfache und weit verbreitete Methode der Clusteranalyse, den sogenannten K-means Algorithmus [19]. Er basiert auf einem quadratischen Gütekriterium und zerlegt die Daten eines D-dimensionalen Datenraumes in eine *vorgegebene* Anzahl an K Gruppen (Clustern), wobei die Zerlegung *linear separierbar* ist, was

bedeutet die Gruppen sind durch $(D-1)$-dimensionale Hyperebenen, also eine Voronoi-Zerlegung, trennbar.

2.1 K-means Algorithmus

Gegeben ist eine Datenmatrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T \in \mathbb{R}^{N\times D}$, die N Datenpunkte $\mathbf{x}_n \in \mathbb{R}^D$ des D-dimensionalen Raumes (als Zeilenvektoren) beinhaltet. Es wird eine feste Menge an K sogenannten Clusterzentren $\mathbf{m}_k \in \mathbb{R}^D$ im gleichen D-dimensionalen Raum vorgegeben und in der Matrix $\mathbf{M} = [\mathbf{m}_1, \ldots, \mathbf{m}_K]^T \in \mathbb{R}^{K\times D}$ zusammengefasst. Jeder Datenpunkt $\mathbf{x}_n$ wird über einen Zugehörigkeitsvektor $\mathbf{r}_n \in \{0,1\}^K$ eindeutig einer Gruppe zugeteilt. Der Zugehörigkeitsvektor realisiert eine *1-aus-k Kodierung* der Gruppen und besitzt binäre Vektorelemente $r_{nk} \in \{0,1\}$, wobei das Vektorelement $r_{nk} = 1$ zu eins gesetzt wird, falls der Datenpunkt zu Cluster k gehört. Alle anderen Elemente werden zu Null $r_{nj} = 0\,, \forall\, j \neq k$ gesetzt. Damit gilt für die Summe aller Vektorelemente jedes Zugehörigkeitsvektors: $\sum_{k=1}^{K} r_{nk} = 1\,, \forall\, n$ [15]. Die Zugehörigkeitsmatrix $\mathbf{R} = [\mathbf{r}_1, \ldots, \mathbf{r}_N]^T \in \mathbb{R}^{N\times K}$ vereint die Zugehörigkeitsvektoren aller Datenpunkte, wobei die Spaltensumme $N_k = \sum_{n=1}^{N} r_{nk}$ der Anzahl N_k der Datenpunkte des entsprechenden Clusters k entspricht.

Das Ziel des K-means Algorithmus ist die Minimierung folgender Gütefunktion

$$J(\mathbf{X},\mathbf{M},\mathbf{R}) = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\mathbf{x}_n - \mathbf{m}_k\|^2 \tag{4}$$

$$= \|\mathbf{X} - \mathbf{R}\mathbf{M}\|_F^2\,, \tag{5}$$

was der Minimierung der Summe der quadrierten euklidischen Distanzen zwischen jedem Datenpunkt $\mathbf{x}_n$ und dem jeweiligen Clusterzentrum $\mathbf{m}_k$ seiner Gruppe $\mathbf{r}_n$ entspricht[1]. Es kann gezeigt werden, dass dies sowohl gleichbedeutend mit der Minimierung der totalen Intra-Cluster-Varianz

$$J(\mathbf{X},\mathbf{R}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K} r_{ik}r_{jk}\|\mathbf{x}_i - \mathbf{x}_j\|^2 \tag{6}$$

als auch der Maximierung der Inter-Cluster-Varianz

$$J(\mathbf{X},\mathbf{R}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}(1 - r_{ik}r_{jk})\|\mathbf{x}_i - \mathbf{x}_j\|^2 \tag{7}$$

ist [21]. Das globale Minimum der nichtkonvexen Gütefunktion (5)

$$(\mathbf{M}^\star, \mathbf{R}^\star) = \mathrm{argmin}_{\mathbf{M},\mathbf{R}}\|\mathbf{X} - \mathbf{R}\mathbf{M}\|_F^2 \tag{8}$$

entspricht einer optimalen Zerlegung $\mathbf{R}^\star$ mit den dazugehörigen Clusterzentren $\mathbf{M}^\star$. Der K-means Algorithmus realisiert die Optimierung der Gütefunktion (8) approximativ über ein Verfahren, welches iterativ, in jeder Iteration t alternierend, zuerst die Zugehörigkeiten $\mathbf{R}^t$ und danach die Clusterzentren $\mathbf{M}^t$ über die Optimierung von zwei vereinfachten Optimierungsproblemen (9) und (11) erneuert[2].

Ausgehend von einer Initialisierung der Clusterzentren $\mathbf{M}^0$ werden über folgendes Optimierungsproblem

$$\mathbf{R}^{t+1} = \mathrm{argmin}_{\mathbf{R}'}\|\mathbf{X} - \mathbf{R}'\mathbf{M}^t\|_F^2 \tag{9}$$

zuerst neue Zugehörigkeiten $\mathbf{R}^{t+1}$ berechnet, während die Clusterzentren $\mathbf{M}^t$ als konstant angenommen werden. Dieser Optimierungsschritt führt zu einer Verbesserung der Zugehörigkeiten

$$r_{nk}^{t+1} = \begin{cases} 1, & \text{falls } k = \mathrm{argmin}_j\|\mathbf{x}_n - \mathbf{m}_j^t\|^2 \\ 0, & \text{sonst} \end{cases}. \tag{10}$$

und entspricht der geschlossenen Lösung des Optimierungsproblems (9), welche gleichzeitig garantiert, dass diese Lösung auch die Gesamtgütefunktion (5) monoton abnehmen lässt $J(\mathbf{X},\mathbf{M}^t,\mathbf{R}^{t+1}) \leq J(\mathbf{X},\mathbf{M}^t,\mathbf{R}^t)$. Danach wird Gleichung (5) nur bezüglich der Clusterzentren $\mathbf{M}^{t+1}$ optimiert, während die Zugehörigkeiten $\mathbf{R}^{t+1}$ unverändert bleiben

$$\mathbf{M}^{t+1} = \mathrm{argmin}_{\mathbf{M}'}\|\mathbf{X} - \mathbf{R}^{t+1}\mathbf{M}'\|_F^2\,. \tag{11}$$

Dieses Teilproblem führt ebenfalls zu einer geschlossenen Lösung

$$\mathbf{m}_k^{t+1} = \frac{1}{N_k^{t+1}}\sum_{n=1}^{N} r_{nk}^{t+1}\mathbf{x}_n\,, \quad \forall\, k \tag{12}$$

und garantiert ebenfalls eine monotone Abnahme der Gesamtgütefunktion $J(\mathbf{X},\mathbf{M}^{t+1},\mathbf{R}^{t+1}) \leq J(\mathbf{X},\mathbf{M}^t,\mathbf{R}^{t+1})$. Gleichung (12) entspricht dem arithmetischen Mittelwert aller Datenpunkte, die momentan zur Gruppe k zugeordnet werden.

Die Lösung der Teilprobleme (10) und (12) wird solange wiederholt, bis sich die Zugehörigkeiten nicht mehr verändern bzw. ein geeignetes Abbruchkriterium erfüllt wird. Üblicherweise wird die relative Änderung der Zugehörigkeitsmatrix

$$\epsilon^t = \frac{\|\mathbf{R}^t - \mathbf{R}^{t-1}\|_F}{\|\mathbf{R}^{t-1}\|_F} \tag{13}$$

1 $\|\cdot\|$ bezeichnet die euklidische Norm und $\|\cdot\|_F$ die Frobeniusnorm.

2 Diese Art von alternierender Optimierung wird auch *Expectation Maximization* (EM) Verfahren genannt und die einzelnen Schritte mit E-Schritt und M-Schritt bezeichnet [15].

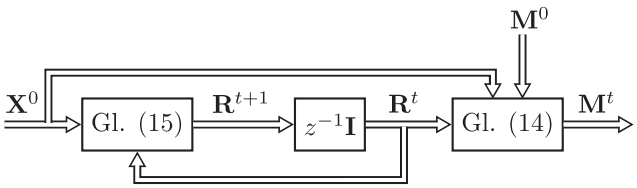

Abbildung 2: Der iterative K-means Algorithmus als stabiles zeitdiskretes nichtlineares dynamisches System mit den Daten $\mathbf{X}^0$ als Eingangsmatrix, den extrahierten Prototypen $\mathbf{M}^t$ der Gruppen als Ausgangsmatrix und den Zugehörigkeiten der Daten zu den Gruppen als Zustandsmatrix $\mathbf{R}^t$.

als Maß für das Abbruchkriterium $\epsilon^t < \epsilon_0$ benutzt, wobei ϵ_0 ein festzulegender Schwellwert ist.

2.2 K-means Dynamik und Konvergenz

Betrachtet man die iterative Vorschrift des K-means Algorithmus[3], so lässt sich ein diskretes nichtlineares System für die iterative Abfolge der Zugehörigkeiten angeben. Abbildung 2 zeigt das entsprechende Blockschaltbild. Dazu kann Gleichung (12) in Matrixschreibweise formuliert werden

$$\mathbf{M} = [\mathbf{R}^T\mathbf{R}]^{-1}\mathbf{R}^T\mathbf{X} = \mathbf{R}^\dagger\mathbf{X}\,, \tag{14}$$

wobei die Elemente der Diagonalmatrix $[\mathbf{R}^T\mathbf{R}]^{-1} = diag(1/N_k) \in \mathbb{R}_+^{K\times K}$ den reziproken Werten von N_k entsprechen und $\mathbf{R}^\dagger = [\mathbf{R}^T\mathbf{R}]^{-1}\mathbf{R}^T \in \mathbb{R}_+^{K\times N}$ die *Moore-Penrose*-Pseudoinverse der Zugehörigkeitsmatrix $\mathbf{R}$ darstellt. Die Pseudoinverse $\mathbf{R}^\dagger$ entspricht der mit N_k zeilenweise normierten transponierten Zugehörigkeitsmatrix, wobei für die Zeilensumme der Pseudoinversen gilt: $\sum_{n=1}^{N} r_{kn}^\dagger = 1\ \forall k$.

Wird Gleichung (14) in Gleichung (9) eingesetzt, dann ergibt sich die Dynamik der Zugehörigkeiten

$$\mathbf{R}^{t+1} = \text{argmin}_\mathbf{R}\|(\mathbf{I} - \mathbf{R}(\mathbf{R}^t)^\dagger)\mathbf{X}\|_F^2\,, \tag{15}$$

die über die rekursive Lösung eines Optimierungsproblems beschrieben ist. Die Rekursion (15) beinhaltet eine endliche Anzahl an Zugehörigkeiten, konvergiert nach endlichen Schritten in eine stationäre Lösung $\mathbf{R}^t \to \mathbf{R}_s$ für $t \to t_{end}$ und entspricht einem lokalen Minimum der nichtkonvexen Gütefunktion (15) [15, 20].

In der Literatur werden vier Nachteile des K-means Algorithmus hervorgehoben, welche sowohl die Güte als auch die Konvergenz negativ beeinflussen können, obwohl die Anzahl der Clusterzentren K gut gewählt ist[4].

Erstens, die Rekursion kann in lokalen Maxima und Minima sowie Sattelpunkten stagnieren [20]. Das bedeutet, die Zugehörigkeiten (15) können sich ändern, aber der Wert der Gütefunktion (8) ändert sich nicht und ist damit nicht streng monoton fallend. Zweitens, wegen der Nichtkonvexität der Gütefunktion ist die Güte der Gruppierung $\mathbf{R}^\star$ stark von der Initialisierung der Clusterzentren $\mathbf{M}^0$ abhängig. Drittens, die euklidische Norm ist nicht robust gegenüber Ausreißern in den Daten [22]. Viertens, die Daten sind über das Modell der Zerlegung mittels Hyperebenen häufig nicht separierbar [23]. Der K-means Algorithmus kann Datenmengen unterschiedlicher Gruppenausdehnungen und -dichten nur unzureichend trennen, da er dazu tendiert die Daten in Gruppen gleicher Größe und Dichte zu gruppieren.

Zur weiteren Beschreibung der K-means Dynamik führen wir hier eine Matrix $\Delta\mathbf{R}^t \in \mathbb{R}_+^{K\times K}$ ein, die nützliche Informationen über die Dynamik der Clustergrößen beinhaltet:

$$\Delta\mathbf{R}^t := (\mathbf{R}^t)^\dagger\mathbf{R}^{t+1} = [(\mathbf{R}^t)^T\mathbf{R}^t]^{-1}(\mathbf{R}^t)^T\mathbf{R}^{t+1}\,. \tag{16}$$

Diese Matrix ist eine zeilenstochastische Matrix und beschreibt die Übergänge der Clustergrößen und den Wechsel von Zugehörigkeiten zwischen den Clustern aufeinanderfolgender Iterationen. Anhand der Elemente Δr_{kl}^t dieser Matrix kann abgelesen werden, welcher Cluster wieviele Datenpunkte an welche Cluster abgibt und welcher Cluster von welchen Clustern wieviele Datenpunkte bekommt, normiert auf die jeweilige momentane Clustergröße N_k^t. Die Spaltensummen der Matrix $(\mathbf{R}^t)^T\mathbf{R}^{t+1} \in \mathbb{R}_+^{K\times K}$ ergeben die Clustergrößen N_k^{t+1} zum neuen Iterationsschritt $t+1$ und die Zeilensummen ergeben die Clustergrößen N_l^t zum momentanen Iterationsschritt t. Die Diagonalelemente Δr_{kk}^t entsprechen den Clustergrößen nach Abgabe aller Datenpunkte an alle anderen Cluster, die Spaltenelemente $\Delta r_{lk}^t\ \forall\, l \neq k$ der Spalte k geben an, wieviele Datenpunkte Cluster k von Cluster l bekommt und die Zeilenelemente $\Delta r_{lk}^t\ \forall\, k \neq l$ der Zeile l geben an, wieviele Datenpunkte Cluster l an Cluster k abgibt, inklusive entsprechender Zeilennormierung mittels der aktuellen Clustergrößen N_l^t. Ist der K-means Algorithmus konvergiert, dann gilt: $\Delta\mathbf{R}^{t_{end}} = \mathbf{I}$ und es kann das alternative Maß $\epsilon^t = \|\Delta\mathbf{R}^t - \mathbf{I}\|_F$ zum Abbruch benutzt werden.

Die Matrix $\Delta\mathbf{R}$ entspricht einer gewichteten Adjazenzmatrix für einen gerichteten schleifenbehafteten Graphen,

3 Der Iterationsindex t wird ab jetzt vernachlässigt, falls er nichts zur Verständlichkeit beiträgt, um die Lesbarkeit zu vereinfachen.

4 Bzw. bei Vorwissen über die Gruppenanzahl exakt gewählt werden kann.

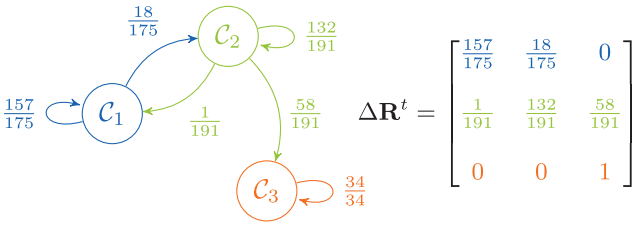

Abbildung 3: Beispiel eines gerichteten, schleifenbehafteten Clustergraphen mit dazugehöriger Cluster-Adjazenzmatrix $\Delta\mathbf{R}^t$.

wobei jeder Knoten die Größe eines Clusters repräsentiert und jede Kante den Übergang der Anzahl von Zugehörigkeiten von einem zum anderen Cluster darstellt. Abbildung 3 zeigt ein Beispiel einer solchen Matrix und den dazugehörigen Graphen für Clustergrößenänderungen von drei Clustern über einem Datenbestand von 400 Datenpunkten.

Abschließend ist der K-means Algorithmus nochmals im Pseudocode 1 zusammengefasst.

Pseudocode 1 K-means Algorithmus

Wähle K, ϵ_0
Initialisiere $\mathbf{M}$
while $\epsilon \geq \epsilon_0$ **do**
  $\mathbf{R} \leftarrow \text{argmin}_{\mathbf{R}} ||\mathbf{X} - \mathbf{RM}||_F^2$
  $\mathbf{M} \leftarrow \mathbf{R}^\dagger \mathbf{X}$
  $\epsilon \leftarrow ||\Delta\mathbf{R} - \mathbf{I}||$
end while

2.3 Entwurf eines Datenreglers

Abbildung 4 zeigt die Struktur des Regelkreises für eine Datenregelung zur Clusteranalyse über den K-means-Algorithmus. Es wird von einer allgemeinen nichtlinearen Dynamik der Daten ausgegangen $\mathbf{X}^{t+1} = f(\mathbf{X}^t, \mathbf{R}^t)$, was bedeutet, die Daten sind beliebig transformierbar.

Ziel der Regelung Es wird eine Stellgrößenmatrix $\mathbf{U}^t \in \mathbb{R}^{N\times D}$ gesucht, um die Daten iterativ so zu transformieren, dass zum einen die Iterationsanzahl der Datenzerlegung verringert wird und zum anderen die Zerlegungsgüte erhöht wird. Beide Forderungen können nicht explizit formuliert werden und beeinflussen sich gegenseitig. Die Clusteranalyse ist ein unüberwachter Lernvorgang und deshalb liegen keine Lernvektoren (*teaching vectors*) für eine Bewertung der Güte der Datenzerlegung vor. Die Iterationszahl hängt von der Art und Weise ab, wie die Gütefunktion *durchschritten* wird und ist von dieser Schrittweite abhängig. Die Schrittweite der Zugehörigkeitsänderung ist in Gleichung (15) implizit über die Lösung des Optimierungsproblems (9) vorgegeben.

Da angenommen wird, dass kein Wissen über die spezifische Struktur der Daten vorhanden ist, wird die momentane Intra-Cluster-Varianz in Gleichung (5) zur Datentransformation herangezogen und es werden zwei Bedingungen an die Eigenschaft des Reglers gestellt.

Bedingung I Die Datentransformation soll auf jeden Fall eine monotone Abnahme $J(\mathbf{X}^{t+1}, \mathbf{M}^t, \mathbf{R}^t) \leq J(\mathbf{X}^t, \mathbf{M}^t, \mathbf{R}^t)$ der Gütefunktion (5) garantieren.

Des Weiteren ist keine explizite Regel dafür ableitbar, inwieweit die Datentransformation strukturerhaltend ist.

Bedingung II Solange die Änderung der Zugehörigkeiten $\|\Delta\mathbf{R}^t - \mathbf{I}\|_F$ groß ist, soll die Änderung der Daten aufgrund der Datentransformation viel kleiner sein als die Änderung der Clusterzentren und damit der Referenzdaten, welche die Gruppen am besten beschreiben $\|\mathbf{X}^{t+1} - \mathbf{X}^t\|_F/N \ll \|\mathbf{M}^{t+1} - \mathbf{M}^t\|_F/K$.

Die Optimierung der Gütefunktion soll also viel stärker von der Veränderung der Zugehörigkeiten $\mathbf{R}$ und damit der Clusterzentren $\mathbf{M}$ als von der Veränderung der Daten $\mathbf{X}$ abhängen.

Um eine Datentransformation zu finden, welche eine monotone Abnahme der Gütefunktion (5) garantiert, wird der Gradient der Gütefunktion nach den Daten gebildet, unter der Annahme, dass sich die Zugehörigkeiten und die Clusterzentren nicht ändern

$$\frac{\partial J(\mathbf{X}^t, \mathbf{M}^t, \mathbf{R}^t)}{\partial \mathbf{X}^t} = \frac{\partial}{\partial \mathbf{X}^t} \|\mathbf{X}^t - \mathbf{R}^t\mathbf{M}^t\|_F^2 \propto \left(\mathbf{X}^t - \mathbf{R}^t\mathbf{M}^t\right). \tag{17}$$

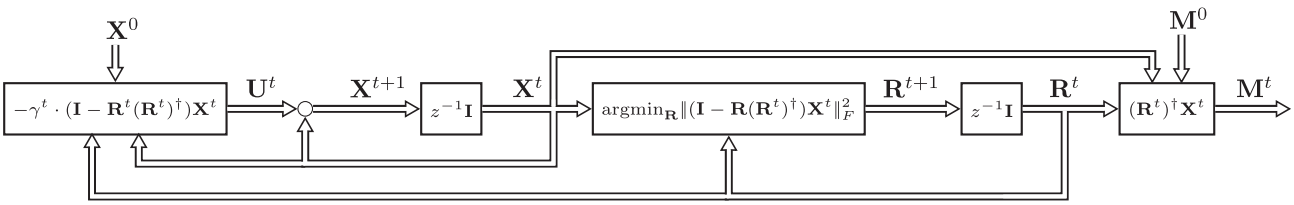

Abbildung 4: Regelkreis zur Datenanpassung für den K-means Algorithmus.

Die optimale Datentransformation bezüglich der Gütefunktion

$$\mathbf{X}^\star = \text{argmin}_{\mathbf{X}} J(\mathbf{X}^t, \mathbf{M}^t, \mathbf{R}^t) = \mathbf{R}^t\mathbf{M}^t \tag{18}$$

wäre bei einem Gradientenvektor $\partial J(\mathbf{X}, \mathbf{M}^t, \mathbf{R}^t)/\partial \mathbf{X} \stackrel{!}{=} \mathbf{0}$ und würde die Datenpunkte mit ihren momentanen Clusterzentren zusammenfallen lassen $\mathbf{X}^\star = \mathbf{R}^t\mathbf{M}^t$. Damit wären zwar alle Euklidischen Distanzen der Gütefunktion (5) gleich Null und die Gütefunktion minimal, aber die Datenstruktur nicht mehr vorhanden.

Damit die Datentransformation garantiert die Gütefunktion optimiert, aber dennoch die Datenstruktur größtenteils erhalten bleibt, verändern wir die Daten über eine kleine adaptive Schrittweite γ^t nur ein wenig in Richtung der negativen Gradientenmatrix (17) der Gütefunktion

$$\begin{aligned}
\mathbf{X}^{t+1} &= f(\mathbf{X}^t, \mathbf{R}^t) = \mathbf{X}^t + \mathbf{U}^t && (19)\\
&= \mathbf{X}^t - \gamma^t \cdot \frac{\partial J(\mathbf{X}^t, \mathbf{M}^t, \mathbf{R}^t)}{\partial \mathbf{X}^t} && (20)\\
&= \mathbf{X}^t - \gamma^t \cdot (\mathbf{X}^t - \mathbf{R}^t\mathbf{M}^t) && (21)\\
&= [\mathbf{I} - \gamma^t \cdot (\mathbf{I} - \mathbf{R}^t(\mathbf{R}^t)^\dagger)]\mathbf{X}^t && (22)\\
&= \mathbf{P}^t\mathbf{X}^t && (23)
\end{aligned}$$

und es ergibt sich zusammen mit der Teildynamik (15) ein nichtlineares diskretes adaptives Gesamtsystem für beide Größen $\{\mathbf{X}^t, \mathbf{R}^t\}$, wobei die Datendynamik (23) durch eine adaptive Systemmatrix $\mathbf{P}^t \in \mathbb{R}^{N\times N}$ beschrieben wird. Diese Matrix ist von der adaptiven Schrittweite γ^t und den Zugehörigkeiten $\mathbf{R}^t$ abhängig. Die Datendynamik führt dazu, dass sich alle Daten, die momentan zu einer Gruppe gehören, ein wenig an ihren momentanen Mittelwert annähern, also dem momentan angenommenen Clusterzentrum. Der Mittelwert der Daten verändert sich dadurch nicht (siehe dazu Abschnitt 2.6). Die Intra-Cluster-Varianz (6) wird künstlich über die virtuelle Datendynamik verringert. Jede Gruppe vergrößert also die Ähnlichkeit zwischen den Daten, die momentan der gleichen Gruppe zugeordnet werden.

Um die Separierbarkeit der Gruppierungen zu bewerten, sind die Datenpunkte, die am nächsten zu den Hyperebenen liegen, also deren orthogonaler Abstand – die sogenannte Toleranz (*margin*) – zur jeweiligen Hyperebene am geringsten ist, von Bedeutung. Diese Datenpunkte werden auch Stützvektoren (*support vectors*) genannt [24]. Die Region, welche durch eine Hyperebene und eine parallele Hyperebene durch einen Stützvektor aufgespannt wird, bezeichnet man als Toleranzband (*margin band*) [25]. Das Gütekriterium des K-means-Algorithmus maximiert genau diese Toleranzbänder. Betrachtet man die Auswirkung der Datendynamik nochmal bezogen auf die Separierbarkeit der Daten, dann bekräftigen die virtuellen Daten den momentanen Clusterzustand, indem sie ihren Abstand zu den Hyperebenen (der Voronoi Zerlegung) der momentanen Datenzerlegung vergrößern. Um eine Veränderung der Zerlegung durch eine Veränderung der Lage der Hyperebenen hervorzurufen, muss sich nun die Lage der Hyperebenen im Vergleich zur Situation ohne Datenbewegung stärker verändern.

Die Datendynamik (23) entsteht über einen Gradientenabstieg. Damit bleibt die Datenzerlegung stabil für genügend kleine Schrittweiten γ^t. Die adaptive Schrittweite γ^t ist ein ausschlaggebender Parameter dafür, wie strukturerhaltend sich die Datendynamik auswirkt. Damit sich eine Datenzerlegung ausprägen kann, muss die Veränderung der Daten viel langsamer geschehen, als die Veränderung der Zugehörigkeiten und damit der Clusterzentren. Hierzu ist eine sehr kleine Schrittweite γ^t nahe Null nötig. Je mehr sich die Datenzerlegung stabilisiert, desto stärker kann die Datendynamik die Zerlegung über größer werdende Schrittweiten γ^t unterstützen, da immer weniger Daten die Gruppenzugehörigkeit wechseln und immer kleinere Veränderungen der Zerlegung hervorrufen. Hierbei kann die Datendynamik die Entscheidung dieser Daten für eine Gruppe umso mehr begünstigen, je größer die Schrittweite γ^t gewählt wird.

Unter Berücksichtigung dieser qualitativen Zusammenhänge, wird folgende Adaption der Schrittweite vorgeschlagen:

$$\gamma^t = \gamma_0 \cdot \begin{cases} 1, & \text{falls } t \geq \tau \\ \frac{t}{\tau}, & \text{sonst} \end{cases}. \tag{24}$$

Die Schrittweite vergrößert sich linear über den Iterationen t von Null bis zu einer festgelegten Sättigung von γ_0 mit der Steigung $1/\tau$. Dies ist eine sehr einfache Adaption, welche die momentane Clustergüte und die Veränderung der Daten nicht berücksichtigt, sondern über die Parameterwahl von τ und γ_0 eine Adaption vorgibt.

Im Pseudocode 2 ist der Ablauf des datengeregelten K-means-Algorithmus zusammenfassend dargestellt. Im Folgenden werden die Eigenschaften der Datenregelung genauer untersucht.

Pseudocode 2 Datengeregelter K-means Algorithmus

Wähle $K, \epsilon_0, \gamma_0, \tau$
Initialisiere $\mathbf{M}$, $\mathbf{P} = \mathbf{I}, t = 0$
while $\epsilon \geq \epsilon_0$ **do**
  $\mathbf{R} \leftarrow \text{argmin}_{\mathbf{R}} ||\mathbf{X} - \mathbf{RM}||_F^2$
  $\mathbf{M} \leftarrow \mathbf{R}^\dagger \mathbf{X}$
  Datenregler
  $\mathbf{X} \leftarrow \mathbf{PX}$
  $t \leftarrow t + 1$
  if $t \geq \tau$ **then**
    $\gamma \leftarrow \gamma_0$
  else
    $\gamma \leftarrow t/\tau$
  end if
  $\mathbf{P} \leftarrow \left[\mathbf{I} - \gamma \cdot (\mathbf{I} - \mathbf{RR}^\dagger)\right]$
  $\epsilon \leftarrow ||\Delta\mathbf{R} - \mathbf{I}||$
end while

2.4 Konvergenz des datengeregelten K-means-Algorithmus

Um Konvergenz des datengeregelten K-means-Algorithmus zu garantieren, muss die Schrittweite in gewissen Schranken verbleiben. Die Teildynamik (23) beschreibt ein diskretes adaptives nichtlineares System, für das Stabilität normalerweise nur schwer zu beweisen ist. Da die Stellgröße jedoch einem Gradientenabstieg entspricht und wir außerdem eine analytische Lösung für das Optimum des Optimierungsproblems (18) ausrechnen können, ist der Bereich für eine stabile Schrittweite direkt angebbar: $0 \leq \gamma^t \leq 1$. Für eine Schrittweite von $\gamma^t = 0$ werden die Daten überhaupt nicht transformiert, es liegt ein normaler K-means-Algorithmus vor und dieser konvergiert beweisbar in endlichen Iterationsschritten [20]. Für eine Schrittweite von $\gamma^t = 1$ fallen die Datenpunkte mit ihren momentanen Clusterzentren zusammen $\mathbf{X}^t = \mathbf{R}^t\mathbf{M}^t$ (siehe auch Abschnitt 2.3). Dadurch ändern sich die Zugehörigkeiten über das Optimierungsproblem (9) nicht mehr $\mathbf{R}^{t+1} = \mathbf{R}^t, \forall t$ und die Clusterzentren bleiben bei der Optimierung des Optimierungsproblems (11) ebenfalls erhalten $\mathbf{M}^{t+1} = \mathbf{M}^t, \forall t$. Für alle Schrittweiten im Bereich $0 < \gamma^t \leq 1$ ist eine *strenge* monotone Abnahme $J(\mathbf{X}^{t+1}, \mathbf{M}^t, \mathbf{R}^t) < J(\mathbf{X}^t, \mathbf{M}^t, \mathbf{R}^t)$ von (18) garantiert. Deswegen verhindert die Datenregelung das Stagnieren der Optimierung in einem lokalen Maximum oder auf einem Sattelpunkt. Ob sich die Iterationszahl deswegen garantiert verringert im Vergleich zum K-means-Algorithmus ohne Datenregelung, konnte bis jetzt noch nicht bewiesen werden. Allerdings kann experimentell gezeigt werden, dass häufig ein Verlauf von γ^t gefunden werden kann, der bei gleicher Initialisierung wesentlich weniger Iterationen aufweist, als der herkömmliche K-means-Algorithmus und die Güte der Zerlegung gleich bleibt bzw. besser wird (siehe dazu Abschnitt 3).

2.5 Bezug zur Consensusdynamik

Consensusdynamiken beschreiben die Synchronisierung von Zuständen in einem Multi-Agenten-System zu einem gemeinsamen Zustand $\mathbf{x}_1 = \mathbf{x}_2 = \ldots \mathbf{x}_N$, wobei die N Agenten über ein Netzwerk kommunizieren und die Kommunikationsstruktur über einen Graphen dargestellt werden kann. Die Synchronisierung erfolgt nur über lokalen Informationsaustausch und kann somit verteilt implementiert werden. Im Folgenden wird die Äquivalenz der Datendynamik zu einer speziellen Consensusdynamik hergestellt und gezeigt, dass die Clusteranalyse über eine Datendynamik verteilt implementierbar ist und damit auf große dezentral abgelegte Datenmengen anwendbar ist.

Zeitinvariante diskrete Consensusdynamiken werden über die *Perron*-Matrix $\tilde{\mathbf{P}}_k$ beschrieben [26]. Sie sind für alle Graphen $\mathcal{G}_k$, die einen gerichteten Spannbaum enthalten und Abtastzeiten zwischen $0 < \Delta t < 1/\delta_k$ besitzen, asymptotisch stabil und konvergieren auf die Mittelwerte der Anfangszustände, da die Eigenwerte $\lambda_{ki} \in \mathbb{C}$ der Matrix $\tilde{\mathbf{P}}_k$ die Eigenschaft $1 = \lambda_{k1} > |\lambda_{ki}|, \forall\, i \geq 2$ erfüllen und damit alle $\lambda_{ki}, \forall\, i \geq 2$ im Einheitskreis der komplexen Ebene liegen, wobei λ_{k1} dem Spektralradius $\varrho(\tilde{\mathbf{P}}_k)$ entspricht [27]. Zeitvariante diskrete Consensusdynamiken $\tilde{\mathbf{P}}_k^t$ konvergieren, falls die Vereinigung der zeitvarianten Graphen $\mathcal{G}_k^{t:t+T} := \{\mathcal{G}_k^t \cap \ldots \cap \mathcal{G}_k^{t+T}\}$ innerhalb des endlichen Zeitintervalls $[t, t+T]$ zusammenhängend ist [28].

Wird die Matrix $\mathbf{R}$ entsprechend der Gruppenzugehörigkeit der Daten permutiert $\tilde{\mathbf{R}} = \mathbf{\Pi R}$ (Zeilenvertauschungen), wobei $\mathbf{\Pi}$ eine entsprechende Permutationsmatrix ist [29], so dass alle Zugehörigkeitsvektoren $\mathbf{r}_n$, welche eine identische Zugehörigkeit kodieren untereinander stehen und aufsteigend abgelegt werden, dann wird $\tilde{\mathbf{R}} = diag(\mathbf{1}_1, \ldots, \mathbf{1}_k, \ldots, \mathbf{1}_K) \in \{0,1\}^{N \times K}$ zu einer Blockdiagonalmatrix, wobei jeder Block einen Vektor $\mathbf{1}_k \in \{1\}^{N_k}$ darstellt. Die Matrix $[\mathbf{I} - \tilde{\mathbf{R}}\tilde{\mathbf{R}}^\dagger]$ entspricht dann einer mit $\mathbf{D} = diag(\mathbf{D}_1, \ldots, \mathbf{D}_k, \ldots, \mathbf{D}_K) \in \mathbb{R}_+^{N \times N}$ gewichteten Blockdiagonalmatrix $\mathbf{L} = diag(\mathbf{L}_1, \ldots, \mathbf{L}_k, \ldots, \mathbf{L}_K) \in \mathbb{Z}^{N \times N}$:

$$\mathbf{I} - \tilde{\mathbf{R}}\tilde{\mathbf{R}}^\dagger = \mathbf{DL}\,. \tag{25}$$

Jeder Block beinhaltet eine symmetrische *Laplace*-Matrix $\mathbf{L}_k \in \mathbb{Z}^{N_k \times N_k}$, die mit einer Diagonalmatrix

$\mathbf{D}_k = diag(\ldots, 1/N_k, \ldots) \in \mathbb{R}_+^{N_k \times N_k}$ gewichtet wird, so dass sich $\mathbf{D}_k\mathbf{L}_k = 1/N_k\mathbf{L}_k$ ergibt. Jede dieser Cluster-Laplacematrizen $\mathbf{L}_k$ beschreibt einen ungerichteten, vollständig zusammenhängenden, δ-regulären Graphen $\mathcal{G}_k = (\mathcal{V}_k, \mathcal{E}_k)$ mit $i = 1, \ldots, N_k$ Knoten $v_{ki} \in \mathcal{V}_k(\mathcal{G}_k)$, $j = 1, \ldots, (N_k - 1)$ Kanten $e_{kj} \in \mathcal{E}_k(\mathcal{G}_k)$ der endlichen Kantenmenge $\mathcal{E}_k(\mathcal{G}_k) \subseteq \mathcal{V}_k \times \mathcal{V}_k$ und Eingangsgrad $\delta_k = (N_k - 1)$. Damit entspricht die zeilensortierte Matrix der Datendynamik $\tilde{\mathbf{P}} = \mathbf{I} - \gamma\mathbf{D}\mathbf{L}$ ebenfalls einer symmetrischen Blockdiagonalmatrix $\tilde{\mathbf{P}} = diag(\tilde{\mathbf{P}}_1, \ldots, \tilde{\mathbf{P}}_k, \ldots, \tilde{\mathbf{P}}_K) \in \mathbb{R}_+^{N \times N}$, wobei jeder Block $\tilde{\mathbf{P}}_k = \mathbf{I} - \Delta t\mathbf{L}_k \in \mathbb{R}_+^{N_k \times N_k}$ eine *Perron*-Matrix mit Abtastzeit $\Delta t = \gamma/N_k$ darstellt.

Damit ist gezeigt, dass die Datendynamik verteilt implementiert werden kann und zur Clusteranalyse von dezentral gespeicherten Datenbeständen geeignet ist.

In Abbildung 5 ist ein Beispiel für zwei Cluster $\mathcal{C}_1$ und $\mathcal{C}_2$ bei einem Datenbestand von fünf zweidimensionalen Daten $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5)^T$ dargestellt, wobei momentan die ersten drei Daten $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\} \in \mathcal{C}_1$ dem Cluster eins und die letzten beiden Daten $\{\mathbf{x}_4, \mathbf{x}_5\} \in \mathcal{C}_2$ dem Cluster zwei zugeordnet sind. Es sind die entsprechenden vollständig zusammenhängenden Graphen $\mathcal{G}_1$ und $\mathcal{G}_2$ der Cluster-Laplacematrizen $\mathbf{L}_1$ und $\mathbf{L}_2$ sowie der Hyperebene $\mathcal{H}$ (Trennlinie) visualisiert.

Eine weitere interessante Beobachtung bei der Evaluation in Abschnitt 3 ist, dass sich bei Datensätzen, die nicht über Hyperebenen trennbar sind, eine bessere Clustergüte mit Datendynamik erreichen lässt. Aus diesen Experimenten kann man schlussfolgern, dass die rekursive Datentransformation über den Datenregler die Daten in einen Merkmalsraum transformiert, indem diese Merkmale wieder linear separierbar sind. Wäre das der Fall, dann würde eine enge Verwandtschaft zu Kernelmethoden bestehen [25]. Dies soll im folgenden Abschnitt näher betrachtet werden.

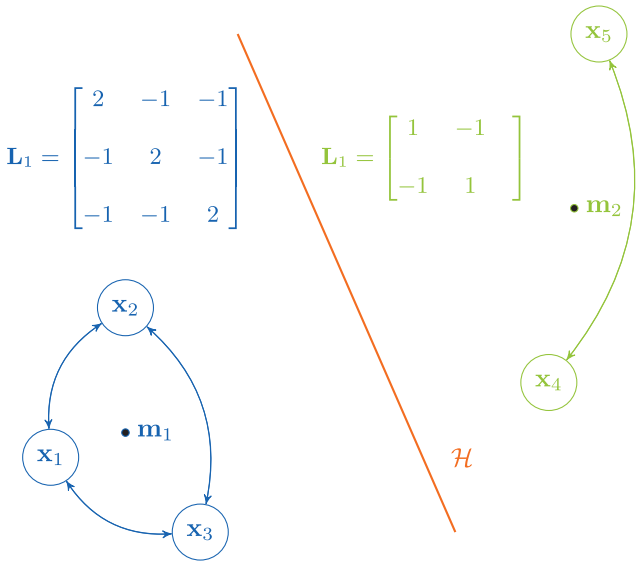

Abbildung 5: Beispiel von zwei vollständig zusammenhängenden Consensus-Graphen $\mathcal{G}_1$ und $\mathcal{G}_2$ mit dazugehörigen Cluster-Laplacematrizen $\mathbf{L}_1$ und $\mathbf{L}_2$ sowie den Clusterzentren $\mathbf{m}_1$ und $\mathbf{m}_2$ und der Hyperebene $\mathcal{H}$.

2.6 Verwandtschaft zu Kernelmethoden

*Kernel*methoden[5] [25, 30] umfassen ein etabliertes und elegantes Verfahren des maschinellen Lernens. Ein *Kernel* bettet Daten aus dem Datenraum $\mathcal{D}$ in einen geeigneten Merkmalsraum $\mathcal{F}$ ein. Dazu wird jedes Datum $\mathbf{x}$ des Datenraumes in den Merkmalsraum abgebildet $\Phi : \mathbf{x} \in \mathcal{D} \longmapsto \Phi(\mathbf{x}) \in \mathcal{F}$, um dann Muster in diesem Merkmalsraum zu finden. Anstatt nichtlineare Zusammenhänge in den Daten über schwierige nichtlineare Zerlegungsmethoden im Datenraum zu finden, können nichtlineare Zusammenhänge in den Daten jetzt über den Umweg einer nichtlinearen Abbildung mit einfachen linearen Zerlegungsmethoden im Merkmalsraum gefunden werden. Eine in der Literatur etablierte Hypothese besagt, dass wenn die Daten in einen *höherdimensionalen* Merkmalsraum $D(\mathcal{F}) > D(\mathcal{D})$ transformiert werden, dann besteht eine bessere Chance die Daten linear zu separieren als im *niederdimensionalen* Datenraum [23]. Kernelmethoden werden auch beim Zerlegen von Daten angewandt, unter anderem beim K-means-Algorithmus und mit *Kernel-K-means* bezeichnet [30].

Die Abbildung $\Phi(\mathbf{x})$ der Koordinaten der Daten in den Merkmalsraum muss nicht explizit bekannt sein, sondern nur das paarweise Skalarprodukt der Merkmalskoordinaten, was über eine Kernelfunktion $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$ bzw. eine Kernelmatrix $\mathbf{K} = \Phi(\mathbf{X})\Phi(\mathbf{X})^T$ mit den Einträgen $\mathbf{K}_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ beschrieben wird[6]. Eine Kernelfunktion ist nur valide, falls die Kernelmatrix für alle Daten positiv semidefinit ist [25].

Um zu zeigen, dass ein Schritt der Datendynamik eine Abbildung der Daten in einen Merkmalsraum mit einer entsprechenden Kernelmatrix beschreibt, wird das

5 *Kernel* ist ein Anglizismus und wird hier nicht mit *Kern* übersetzt, um eine Verwechslung zum Beispiel zum Kern einer Matrix zu vermeiden.

6 Die Berechnung von Distanzen im Merkmalsraum ohne die Transformation zu kennen bzw. auszuführen, wird in der Literatur mit *Kerneltrick* bezeichnet.

Gütekriterium (15) folgendermaßen umformuliert:

$$\begin{aligned} J(\mathbf{X},\mathbf{R}) &= \|(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)\mathbf{X}\|_F^2 \\ &= \mathrm{tr}\{[(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)\mathbf{X}][(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)\mathbf{X}]^T\} \quad (26)\\ &= \mathrm{tr}\{(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)\mathbf{X}\mathbf{X}^T(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)^T\}\,. \quad (27)\end{aligned}$$

Die Matrix $\mathbf{K}=\mathbf{X}\mathbf{X}^T$ entspricht einer Gramschen Matrix und stellt die triviale Kernelmatrix mit der Abbildung des Datenraumes auf sich selbst $\Phi(\mathbf{X})=\mathbf{X}$ dar [25]. Nun wird die Abbildung $\Phi(\mathbf{X})$ durch die Datendynamik (23) ersetzt: $\Phi(\mathbf{X})=\mathbf{P}\mathbf{X}$. Die dazugehörige Gramsche Matrix $\mathbf{K}=\mathbf{P}\mathbf{X}\mathbf{X}^T\mathbf{P}^T$ ist positiv semidefinit [31] und damit eine valide Kernelmatrix.

Ersetzt man die Matrix $\mathbf{P}$ in der Kernelmatrix $\mathbf{K}$ mit ihrer Herleitung $\left[\mathbf{I}-\gamma\cdot(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)\right]$ aus Gleichung (22) und benutzt die Eigenschaften, dass die Matrix $(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)$ symmetrisch ist und folgende Projektion $(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)^2=(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)$ gilt [25, 32], dann ergibt sich

$$\begin{aligned} J(\mathbf{P}\mathbf{X},\mathbf{R}) &= \|(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)\mathbf{P}\mathbf{X}\|_F^2 \quad (28)\\ &= \|(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)[\mathbf{I}-\gamma\cdot(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)]\mathbf{X}\|_F^2 \quad (29)\\ &= (1-\gamma)^2\|(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)\mathbf{X}\|_F^2 \quad (30)\\ &= (1-\gamma)^2 J(\mathbf{X},\mathbf{R})\,. \quad (31)\end{aligned}$$

Jede Datentransformation $\Phi(\mathbf{X})=\mathbf{P}\mathbf{X}$ verringert also das Distanzmaß im Merkmalsraum über den Skalierungsfaktor $(1-\gamma)^2$ im Vergleich zum Distanzmaß im Datenraum.

Die Koordinaten der Mittelwerte ändern sich durch die Transformation nicht $\Phi(\mathbf{M})=\mathbf{M}$, denn unter Berücksichtigung der Projektion $(\mathbf{R}\mathbf{R}^\dagger)^2=\mathbf{R}\mathbf{R}^\dagger$ folgt die Eigenschaft $\mathbf{R}^\dagger\mathbf{R}\mathbf{R}^\dagger=\mathbf{R}^\dagger$ und es gilt:

$$\begin{aligned} \Phi(\mathbf{M}) &= \mathbf{R}^\dagger\Phi(\mathbf{X}) \quad (32)\\ &= \mathbf{R}^\dagger[\mathbf{I}-\gamma(\mathbf{I}-\mathbf{R}\mathbf{R}^\dagger)]\mathbf{X} \quad (33)\\ &= [(1-\gamma)\mathbf{R}^\dagger+\gamma\mathbf{R}^\dagger\mathbf{R}\mathbf{R}^\dagger]\mathbf{X} \quad (34)\\ &= \mathbf{R}^\dagger\mathbf{X}=\mathbf{M}\,. \quad (35)\end{aligned}$$

Da sich die Clusterzentren durch die Transformation der Daten nicht ändern, bleibt auch die Voronoi-Zerlegung des Datenraumes unverändert. Allerdings ändert sich die Lage der Daten, wodurch Daten die Voronoizellen im Datenraum wechseln können und sich damit für jeden Wechsel die Zugehörigkeiten ändern. Im Merkmalsraum muss der Wechsel in eine andere Voronoizelle nicht zwangsläufig erfolgen, obwohl im Datenraum ein Wechsel stattgefunden hat. Je größer die Schrittweite, desto kleiner ist die Wahrscheinlichkeit eines Wechsels im Merkmalsraum, da die Merkmale umso stärker auf die Clusterzentren kontrahieren und sich damit von den Separationsflächen der momentanen Datenzerlegung entfernen (siehe auch Abschnitt 2.3). Durch die Datendynamik ergeben sich also andere Zugehörigkeiten entlang der Iterationen des K-means, als ohne Datendynamik und damit auch andere stabile Clusterlösungen nach Konvergenz des datengeregelten K-means-Algorithmus.

Bei der Anwendung von klassischen Kernelmethoden zur Datenzerlegung wird man mit einem inhärenten Problem konfrontiert: Ein geeigneter Kernel zur Datenzerlegung muss Vorwissen über die nichtlinearen Zusammenhänge der Struktur der Daten beinhalten, damit die Datentransformation eine lineare Separierbarkeit im Merkmalsraum erreicht. Da das Ziel der Datenzerlegung gerade das Finden der unbekannten Struktur der Daten ist, verlagert sich dieses Problem bei Einbettung in einen Kernel also lediglich in das Finden eines geeigneten Kernel, der das Wissen über die nichtlinearen Zusammenhänge in den Daten besitzt.

Im Unterschied zum klassischen Kernel-k-means, der die Daten einmalig in einen Merkmalsraum $\Phi(\mathbf{X})$ transformiert, realisiert die vorgestellte Datendynamik eine rekursive Datentransformation $\Phi^t(\mathbf{X}^t,\mathbf{R}^t)=\mathbf{P}^t\mathbf{X}^t$, wobei kein Vorwissen über die Struktur bekannt sein muss, sondern lediglich das bereits vorhandene Wissen über die Struktur zur Transformation herangezogen wird, was über den bisherigen Verlauf der Datenzerlegung erzeugt wurde und in den Zugehörigkeiten $\mathbf{R}^t$ repräsentiert ist.

Die rekursive Transformation berücksichtigt den kompletten bisherigen Verlauf der Daten $\mathbf{X}^{1:t}$ und der Zugehörigkeiten $\mathbf{R}^{0:t}$, da sowohl die Datendynamik (23) als auch die rekursive Vorschrift der Zugehörigkeiten (15) verkoppelt sind:

$$\begin{aligned} \mathbf{X}^{t+1} &= \Phi^t(\mathbf{X}^t,\mathbf{R}^t(\mathbf{X}^t,\mathbf{R}^{t-1})) = \Phi^t(\mathbf{X}^t,\mathbf{R}^{t-1}) \quad (36)\\ &= \prod_{i=1}^{t}\mathbf{P}^i(\mathbf{R}^i(\mathbf{X}^i,\mathbf{R}^{i-1}))\mathbf{X}^1\,. \quad (37)\end{aligned}$$

Dies entspricht einer neuen Klasse von Kerneln, welche die Daten rekursiv in den Merkmalsraum einbetten, mit dem Ziel die Separierbarkeit in Abhängigkeit von der momentanen Clustergüte schrittweise zu erreichen. Dazu muss die Kernelmatrix nicht explizit ausgerechnet werden. Die rekursive Transformationsvorschrift liegt mit Gleichung (23) vor und die Rücktransformation vom Merkmalsraum in den Datenraum kann jederzeit erfolgen, solange die Matrix $\mathbf{P}^{1:t}:=\prod_{i=t}^{1}\mathbf{P}^i$ invertierbar ist. Jede einzelne Transformation $\mathbf{P}^i$ ist invertierbar, solange noch kein Datenpunkt auf ein Clusterzentrum kollabiert ist. Damit ist auch das Produkt aller Transformationen $[\mathbf{P}^{1:t}]^{-1}=\prod_{i=1}^{t}[\mathbf{P}^i]^{-1}$ invertierbar [33].

3 Evaluation

Das Ziel der folgenden Evaluation ist es festzustellen, ob die Datenregelung Vorteile im Vergleich zum klassischen K-means-Algorithmus im Bezug auf die Iterationsanzahl und die Zerlegungsgüte besitzt und bei welcher Art von Datensätzen Unterschiede deutlich zum Vorschein kommen. Zur Bewertung der Güte der Datenzerlegung wurde der *Adjusted Rand Index* (ARI) verwendet [34]. Der ARI ist ein zwischen $[-1; 1]$ normiertes Maß der Ähnlichkeit von Datenzerlegungen, wobei 1 komplette Übereinstimmung bedeutet.

3.1 Benchmark

Zur Auswertung wurden 48 Datensätze gesammelt [35], für die eine ideale Zerlegung (ground truth), also eine Klassifikation (labels) angegeben ist. Die einzelnen Datensätze variieren in der Anzahl der Daten $N \in [210; 5000]$, der Anzahl der Gruppen $K \in [2; 31]$ und der Anzahl der Dimensionen $D \in [2; 7]$.

Als Clusteranzahl K wurde in allen Durchläufen die angegebene Klassenanzahl gewählt. Für die Parameter des Datenreglers wurde für jeden Datensatz eine Rasterung über 9 bzw. 13 unterschiedliche Werte logarithmisch gleichverteilt über den Wertebereichen $\gamma_0 \in [10^{-6}; 1]$ und $\tau \in [1; 10^4]$ vorgenommen und damit 117 Parameterkonfigurationen η pro Datensatz getestet. Jede Parameterkonfiguration wurde für jeden Datensatz mit zufällig ausgewählten Initialisierungen $\mathbf{M}^0$ hundertmal durchlaufen und die Mittelwerte der benötigten Iterationszahl $ITER_\eta$ und der erreichten Güte ARI_η abgespeichert. Als Referenz wurde für die gleichen Initialisierungen die mittlere Iterationanzahl $ITER_0$ und Güte ARI_0 für den klassischen K-means-Algorithmus berechnet. Danach wurde für jeden Datensatz der Lauf $(ITER, ARI)$ ausgewählt, bei dem das Verhältnis $ITER_\eta/ITER_0$ minimal ist unter der Bedingung, dass die dazugehörige Güte equivalent oder besser ist $ARI_\eta/ARI_0 \geq 1$. Abbildung 6 zeigt das Verhältnis $ITER/ITER_0$ aufgetragen über dem Verhältnis ARI/ARI_0. Es ist deutlich zu sehen, dass es für jeden Datensatz mindestens eine Parameterkonfiguration gibt, bei der sowohl die Iterationsanzahl und die Güte sich nicht verändern. Bei vielen Datensätzen erniedrigt sich die Iterationszahl deutlich und für einige dieser Datensätze erhöht sich zusätzlich noch die Güte der Zerlegung. Dies ist vor allem für Datensätze der Fall, die nicht linear separierbar sind. Die mittlere Reduktion der Iterationen beläuft sich auf 87% der Originaliterationsanzahl, die maximale Reduktion auf 23%. Die mittlere Verbesserung der Güte

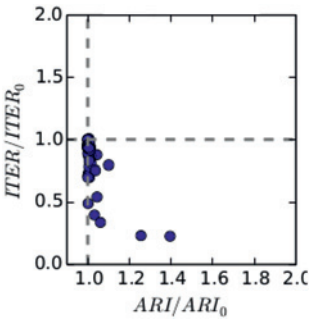

Abbildung 6: Statistische Auswertung bezüglich Iterationsanzahl und Güte der Datenzerlegung.

ergibt eine 1,3%-ige Erhöhung. Das Maximum liegt bei einer Verbesserung von 40% und entsteht bei dem Datensatz, der gleichzeitig die maximale Reduktion an Iterationen auf 23% aufweist. Schaut man sich die Absolutwerte aller ARI an, dann ist eine nahezu optimale Zerlegung von den meisten Datensätzen mit dem klassischen K-means-Algorithmus möglich, wenn genügend Durchläufe mit unterschiedlichen Initialisierungen stattgefunden haben. Das bedeutet, die meisten Datensätze sind linear separierbar. Deswegen fällt die mittlere Verbesserung der Güte so gering aus. Bei den 45 im Benchmark enthaltenen linear separierbaren Datensätzen erzeugt der datengeregelte K-means-Algorithmus einen signifikanten Beschleunigungsvorteil mit einer mittleren Reduktion der Iterationen auf 90%. Bei den 3 deutlich nicht linear separierbaren Datensätzen entsteht eine mittlere Reduktion der Iterationszahl auf 35% bei einer mittleren Gütesteigerung auf 15%. Während hier bei allen drei Datensätzen eine deutliche Reduktion der Iterationsanzahl möglich ist, ist eine Güteverbesserung nur bei zwei Datensätzen zu sehen. Um statistisch signifikante Aussagen zu treffen, sind in Tabelle 1 die Mittelwerte aller ARI μ_{ARI} und die Standardabweichungen σ_{ARI} sowie die mittleren Iterationszahlen μ_{ITER} und deren Standardabweichungen σ_{ITER} über alle Datensätze für den K-means, den Gaussian-Kernel-K-means und den datengeregelten K-means aufgetragen. Hierbei wurden für jeden Lauf die freien Parameter des Gaussian-Kernel-K-means (die Kernelbreite) sowie den datengeregelten K-means (ein festes $\gamma^t = \gamma_0$) optimiert, um kein Verfahren

Tabelle 1: Vergleich zwischen K-means, Gaussian-Kernel-K-means und datengeregeltem K-means.

	K-means	Gaussian-Kernel K-means	datengeregelter K-means
μ_{ARI}	0.55	0.58	0.58
σ_{ARI}	0.32	0.31	0.30
μ_{ITER}	8.50	7.58	5.30
σ_{ITER}	4.73	4.62	2.28

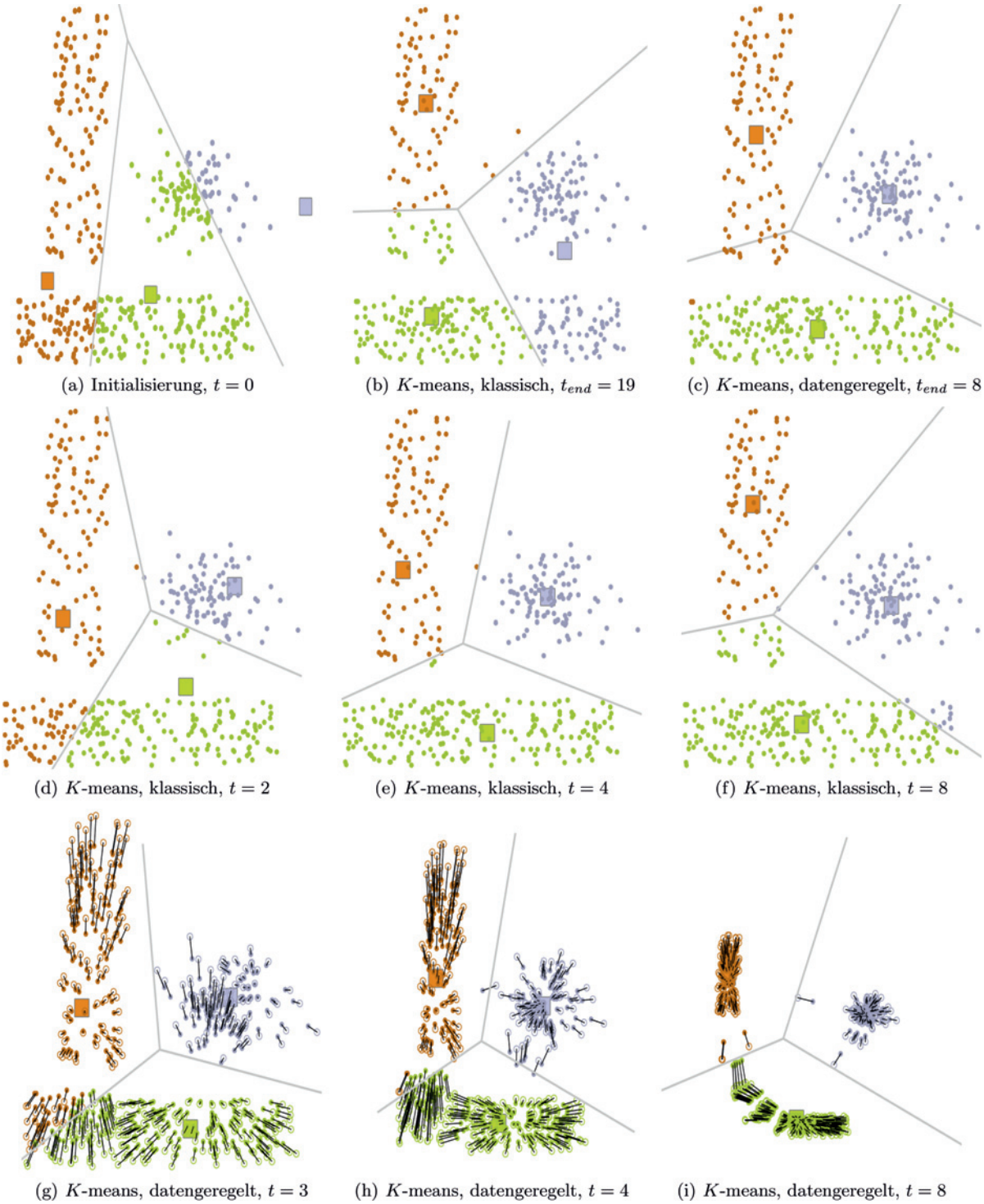

Abbildung 7: Vergleich zwischen dem des klassischen und datengeregelten K-means-Algorithmus. Die Clusterzugehörigkeit ist mit Kreisen der Farben orange, grün und blau gekennzeichnet und in grau ist die Voronoizerlegung über den Clusterzentren, die als gefüllte Quadrate in den gleichen Farben zu sehen sind, visualisiert. Die Daten des momentanen Zeitschritts sind mit gefüllten Kreisen und die des vorangegangenen Zeitschritts mit ungefüllten Kreisen dargestellt. Mit schwarzen Linien von ungefüllten zu gefüllten Kreisen ist die Datendynamik zum nächsten Zeitschritt sichtbar. (a) gibt die equivalente Initialisierung für beide Algorithmen an. (b) zeigt das Ergebnis der Zerlegung für den klassischen K-means, der nach 19 Iterationen beendet ist. (c) zeigt das Ergebnis der Zerlegung für den datengeregelten K-means, der bereits nach 8 Iterationen beendet ist. (d)–(f) zeigt den Zerlegungsfortgang des klassischen K-means für die Iterationsschritte 2, 4, und 8. (g)–(i) zeigt den Zerlegungsfortgang des datengeregelten K-means für die Iterationsschritte 3, 4, und 8. Der Iterationsschritt 2 des datengeregelten K-means entspricht dem Iterationsschritt 2 des klassischen K-means, der in (d) zu sehen ist.

zu bevorzugen. Bezüglich der Güteverbesserung ist der datengeregelte K-means vergleichbar mit dem Kernel-K-means, benötigt jedoch weniger Iterationen und die Varianz in der Konvergenzgeschwindigkeit nimmt ebenfalls ab. Zudem ist die Berechnung der Kernelmatrix rechenintensiv und nicht verteilt implementierbar, während die Datendynamik nur geringen Mehraufwand als der klassische K-means hat und leicht verteilt implementiert werden kann.

Weitere Einzelheiten zu den Datensätzen und deren Quellen sowie detaillierte Auswertungen bezüglich Laufzeit und Rechenzeitersparnis lassen sich in [35] nachlesen.

3.2 Ein Beispiel als Vergleich

Um den positiven Effekt der Datendynamik bei nicht linear separierbaren Datensätzen zu veranschaulichen, wird das Beispiel mit der höchsten Güteverbesserung des Benchmarks herangezogen. Der Datensatz besteht aus 400 zweidimensionalen Datenpunkten und drei Clustern. In Abbildung 7 ist ein Vergleichslauf für eine bestimmte, gleiche Initialisierung (a) mit jeweils drei ausgewählten Iterationsschritten (d)–(f) bzw. (g)–(i) und den beiden Ergebnissen für die K-means Datenzerlegung ohne (b) und mit (c) Datenregelung zu sehen. Während der klassische K-means-Algorithmus erst nach 19 Iterationen konvergiert und mit einem Adjusted Rand Index von $ARI = 0.4$ nicht in der Lage ist eine adäquate Zerlegung zu finden, erreicht die datengeregelte K-means Variante für die Parameter $\gamma_0 = 0.2$ und $\tau = 3$ bereits nach 8 Iterationen eine fast ideale Zerlegung mit nur zwei Fehlklassifikationen und einem Adjusted Rand Index von $ARI = 0.98$. In Iteration (g) und (h) ist deutlich zu sehen, wie durch die Datendynamik Clusterwechsel stattfinden und damit eine andere Zerlegung pro Iteration im Vergleich zum klassischen K-means möglich ist. Betrachtet man das Ergebnis der datengeregelten Zerlegung im originalen Datenraum (c) dann wird auch deutlich, dass die Datenregelung zu einer nichtlinearen Separierbarkeit der Daten führt, denn die Voronoizerlegung über den Clusterzentren entspricht nicht der gefundenen Zerlegung.

4 Diskussion und Ausblick

Als Fazit lässt sich festhalten, dass es mit dem vorgestellten datengeregelten K-means-Algorithmus für alle Beispiele des Benchmark möglich ist einen Parametersatz zu finden, der sowohl eine Beschleunigung der Datenzerlegung als auch eine Verbesserung der Güte erreicht. Diese Vorteile treten besonders deutlich zu Tage, wenn der Datensatz nicht linear separierbar ist. Auch ist keine Sensitivität gegenüber kleineren Parametervariationen von γ^t festzustellen. Für jeden Datensatz gibt es einen Parameterbereich, der fast identische Zerlegungsergebnisse erreicht [35].

Der Zusammenhang zwischen der vorgestellten Datendynamik und der Consensusdynamik lässt viele Erweiterungen zu. Da der klassische K-means auch verteilt berechnet werden kann [9–12] und die Datendynamik ebenfalls verteilt berechenbar ist, da sie einer Consensusdynamik entspricht, kann auch der datengeregelte K-means-Algorithmus komplett verteilt entworfen werden und ist damit auch zur Datenzerlegung von örtlich verteilten und vernetzten Daten einsetzbar. Jeder δ-reguläre Graph konvergiert auf den Mittelwert, wobei die Konvergenzgeschwindigkeit mit kleinerem δ abnimmt. Anstatt den Wert von γ^t zu verringern kann also analog auch der Eingangsgrad δ verringert werden, was in einer verteilten Implementierung zusätzlich Kommunikationsaufwand einspart. Für nicht-δ-reguläre Graphen konvergiert die Consensusdynamik auf einen gewichteten Mittelwert. Die Datendynamik könnte dahingehend verallgemeinert werden, was einen viel größeren Spielraum für mögliche Datentrajektorien zulassen würde. Die Konvergenz ist dann allerdings nicht mehr so leicht zu beweisen, da nicht mehr zwangsläufig ein Gradientenabstieg entlang der K-means Gütefunktion erfolgt. Des Weiteren ist eine einfache Erweiterung zur robusten Datendynamik und damit auch Datenzerlegung bei störbehafteten Daten denkbar, indem die Datendynamik Unsicherheiten in den Daten berücksichtigt, wie das beispielsweise beim Bayes'schen Consensus-Algorithmus möglich ist [36].

Die vorgestellte Arbeit entspricht einer speziellen Abwandlung des sogenannten *N-Consensus* Protokolls [37], welches allerdings nicht Daten zerlegt, sondern Zustände von mobilen Agenten auf mehrere Gruppen synchronisiert, wobei die Gruppenanzahl ebenfalls geschätzt wird und dabei eine weiche Zerlegung in Gruppen mittels eines Bayes'schen Variationsansatzes parallel zur Synchronisierung erfolgt. Damit lässt sich die Idee der Datendynamik auch auf Clusteranalyseverfahren mit weicher Partitionierung und unbekannter Clusteranzahl erweitern [35].

Auch der Bezug zu Kernelmethoden bietet eine Reihe an Erweiterungsmöglichkeiten. Im Prinzip kann die Datendynamik auf alle Zerlegungsmethoden angewandt werden, die auf dem gleichen Gütekriterium beruhen, jedoch andere Nebenbedingungen an die Muster und Zugehörigkeiten stellen. Vorstellbar sind beispielsweise Erweiterungen der Nicht-Negativen Matrixfaktorisierung [38, 39] oder der iterativen Hauptkomponentenanalyse [30].

Nichtzuletzt gehört das vorgestellte Verfahren zur Klasse der verteilten Optimierungsverfahren und kann neue Ideen für gradientenbasierte Reglerentwürfe im Bereich mobiler Multi-Agenten-Systeme [37, 40] liefern, wobei dazu jedes Datum als Zustand eines mobilen Agenten aufgefasst wird.

Danksagung: Wir bedanken uns bei Moritz Schneider für die hilfreichen Diskussionen über die Verwandtschaft zu Kernelmethoden, bei Tatiana Tatarenko, Sebastian Bernhard und Maximillian Löffler für die Diskussionen über die Konvergenzeigenschaften der Datenregelung und bei Dominik Haumann und Stefan Gering für die Vorarbeiten im Bereich der Consensus Regelung und des N-Consensus Problems.

Literatur

1. R. Heinze „Der Produktdatenstandard „teCL@ss“ für die Industrie-4.0-Ontologie“, *etz – Elektrotechnik und Automation, VDE Verlag, 8: 42–47, 2015*.
2. D. Barelmann „Portallösungen für Industrial Big Data“, *etz – Elektrotechnik und Automation, VDE Verlag, 3: 30–31, 2016*.
3. A. Jain, R. Dubes „Algorithms for clustering data“, *Prentice-Hall, Inc., 1988*.
4. A. Jain „Data clustering: 50 years beyond K-means“, *Pattern Recognition Letters, 31(8): 651–666, 2010*.
5. P. D'haeseleer „How does gene expression clustering work?“, *Nature Biotechnology, 23: 1499 –1501, 2005*.
6. V. Kedia, V. Thummala, K. Karlapalem „Time Series Forecasting through Clustering – A Case Study“, *In COMAD, 183–191, 2005*.
7. K.P. Murphy „Machine Learning – A Probabilistic Perspective“, *MIT Press, 2012*.
8. B.J. Frey, D. Dueck „Clustering by Passing Messages Between Data Points“, *Science, 315: 972–976, 2007*.
9. Y. Liang, M.F. Balcan, V. Kanchanapally „Distributed pca and k-means clustering“, *Advances in Neural Information Processing Systems, Big Learning Workshop, 2013*.
10. M.F.F. Balcan, S. Ehrlich, Y. Liang, Y. „Distributed *k*-means and *k*-median Clustering on General Topologies“, *Advances in Neural Information Processing Systems: 1995–2003, 2013*.
11. R. Zhang, I.A. Rudnicky „A large scale clustering scheme for kernel k-means“, *IEEE International Conference on Pattern Recognition, 4: 289–292, 2002*.
12. S. Datta, C. Giannella, H. Kargupta „K-Means Clustering Over a Large“, *Dynamic Network, SDM: 153–164, 2006*.
13. L. Jing, M.K. Ng, J.Z. Huang „An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data“, *IEEE Transactions on Knowledge and Data Engineering, 19(8): 1026–1041, 2007*.
14. J. Wang, Q. Ke, G. Zeng, S. Li, S. „Fast approximate *k*-means via cluster closures“, *Multimedia Data Mining and Analytics: 373–395, 2015*.
15. M. Bishop „Pattern Recognition and Machine Learning“, *Springer Verlag, 2006*.
16. J.Z.C. Lai, T.J. Huang, Y.C. Liaw „A fast *k*-means clustering algorithm using cluster center displacement“, *pattern Recognition, 42: 2551–2556, 2009*.
17. M. Mézard „Where are the Exemplars?“, *Science, 315: 949–951, 2007*.
18. A. Cichocki, R. Zdunek, A.H. Phan, S.I. Amari „Nonnegative Matrix and Tensor Factorizations“, *John Wiley & Sons, Ltd, 2009*.
19. J. MacQueen „Some methods for classification and analysis of multivariate observations“, *In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, 1(14): 281–297, 1967*.
20. S.Z. Selim, M.A. Ismail „K-Means-Type Algorithms: A Generalized Convergence Theorem and Characterization of Local Optimality“, *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6(1): 81–87, 1984*.
21. J. Shawe-Taylor, N. Cristianini „Kernel methods for pattern analysis“, *Cambridge university press, 2004*.
22. K.L. Wu, M.S. Yang „Alternative c-means clustering algorithms“, *Pattern recognition, 35(10): 2267–2278, 2002*.
23. M. Girolami „Mercer kernel-based clustering in feature space“, *IEEE Transactions on Neural Networks, 13(3): 780–784, 2002*.
24. S.R. Kulkarni, G. Harman „Statistical learning theory: a tutorial“ *Wiley Interdisciplinary Reviews: Computational Statistics, 3(6): 543–556, 2011*.
25. C. Campbell „An Introduction to Kernel Methods“, *In: Radial Basis Function Networks: Design and Applications, Springer Verlag, Berlin: 31–69 , 2000*.
26. R. Olfati-Saber, R. Fax, R.M. Murray „Consensus and Cooperation in Networked Multi-Agent Systems“, *Proceedings of the IEEE, 95(1): 215–233, 2007*.
27. W. Ren, R.W. Beard, E.M. Atkins „A Survey of Consensus Problems in Multi-Agent Coordination“, *Proceedings of the American Control Conference: 1859–1864, 2005*.
28. L. Moreau, L. „Stability of multiagent systems with time-dependent communication links“, *IEEE Transactions on Automatic Control, 50(2): 169–182, 2005*.
29. J.L. Stuart, J.R. Weaver „Matrices that commute with a permutation matrix“, *Linear Algebra and Its Applications, 150: 255–265, 1991*.
30. B. Schölkopf, A. Smola, K.-R. Müller „Nonlinear component analysis as a kernel eigenvalue problem“, *Max-Planck-Institut für biologische Kybernetik, Tübingen, Tech. Rep. No. 44, 1996*.
31. S. Lipschutz „Lineare Algebra – Theorie und Anwendung“, *McGraw-Hill Book Company Europe, 1977*.
32. I.S. Dhillon, Y. Guan, B. Kulis „Kernel k-means: spectral clustering and normalized cuts“, *In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining: 551–556, 2004*.
33. C. Voigt, J. Adamy „Formelsammlung der Matrizenrechnung“, *Oldenbourg Wissenschaftsverlag GmbH, 2007*.
34. L. Hubert, P. Arabie „Comparing partitions“, *Journal of classification, 2(1): 193–218, 1985*.
35. M. Schnaubelt „Accelerated data clustering using internal data dynamics“, *Masterarbeit, TU Darmstadt, 2013*.
36. V. Willert, D. Haumann, S. Gering „Decentralized Bayesian Consensus Over Networks“, *Proceedings of the 13th European Control Conference: 1600–1606, 2014*.

37. S. Gering, V. Willert „Solving the N-Consensus Problem: Combining Clustering and Synchronization“, *Proceedings of the 3rd IFAC Workshop on Estimation and Control of Networked Systems, 3(1): 216–221, 2012*.
38. T. Guthier, J. Eggert, V. Willert „Unsupervised Learning of Motion Patterns“, *20th European Symposium on Artificial Neural Networks, 2012*.
39. T. Guthier, V. Willert, J. Eggert „Topological Sparse Learning of Dynamic Form Patterns“, *Neural Computation, 27 (1): 42–73, 2015*.
40. D. Haumann, V. Willert, K. Listmann „DisCoverage: From Coverage to Distributed Multi-Robot Exploration“, *Proceedings of the 4th IFAC Workshop on Distributed Estimation and Control in Networked Systems, 4(1): 328–335, 2013*.

Autoreninformationen

Dr.-Ing. Volker Willert
Technische Universität Darmstadt,
Institut für Automatisierungstechnik
und Mechatronik, Fachgebiet
Regelungsmethoden und Robotik,
Landgraf-Georg-Str. 4, 64283 Darmstadt
vwillert@rtr.tu-darmstadt.de

Dr.-Ing. Volker Willert ist Gruppenleiter der Forschergruppe *Autonome Systeme und Mobile Robotik* des Fachgebietes Regelungstheorie und Robotik unter der Leitung von Prof. Dr.-Ing. J. Adamy am Institut für Automatisierungstechnik und Mechatronik im Fachbereich Elektrotechnik und Informationstechnik der Technischen Universität Darmstadt. Hauptarbeitsgebiete: Inferenzmethoden in der Regelungstechnik, maschinelles Sehen, Mustererkennung, Multi-Agenten-Systeme.

Matthias Schnaubelt, M.Sc.
Rudolf-Bultmann-Str. 9, D-35039 Marburg
matthias.schnaubelt@gmail.com

Matthias Schnaubelt hält einen Master und Bachelor in Physik sowie einen Bachelor in Informationssystemtechnik. Während seiner Abschlussarbeit am Fachgebiet Regelungstheorie und Robotik am Institut für Automatisierungstechnik und Mechatronik im Fachbereich Elektrotechnik und Informationstechnik der Technischen Universität Darmstadt beschäftigte er sich mit Clustering-Algorithmen. Hauptarbeitsgebiete: Datenanalyse, Variational Bayes Clustering.