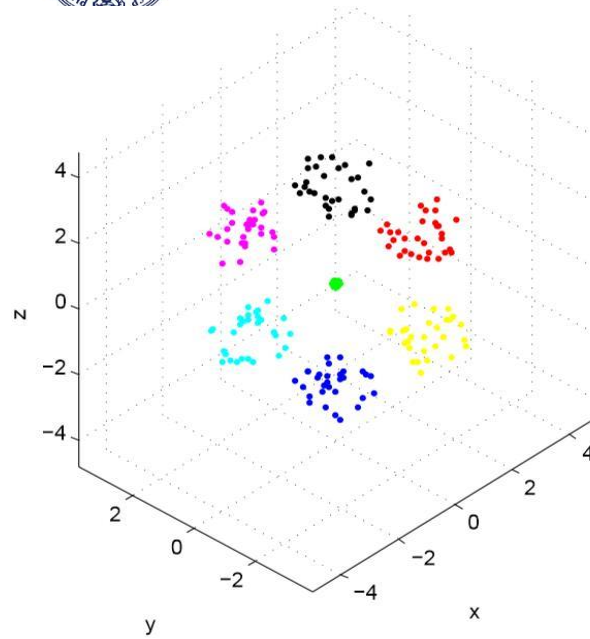


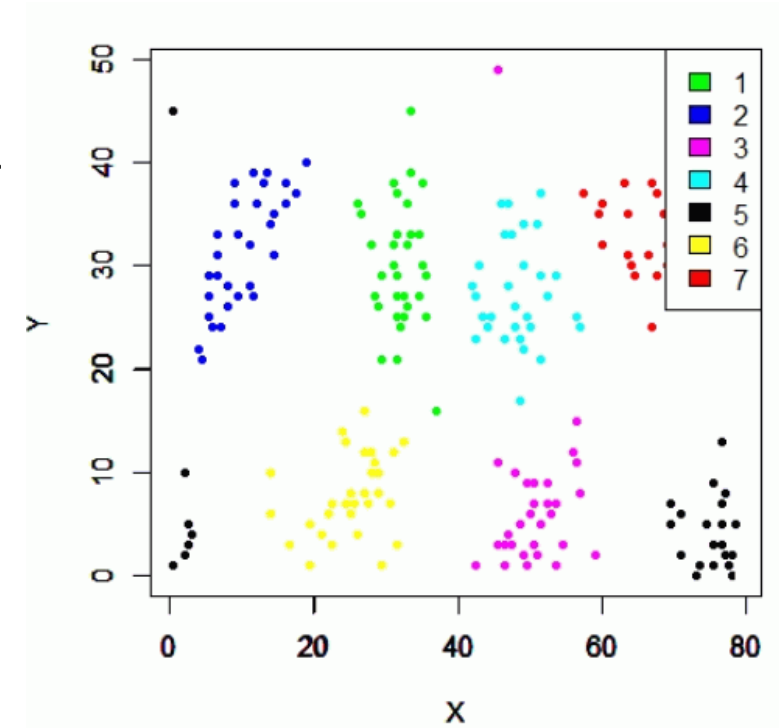
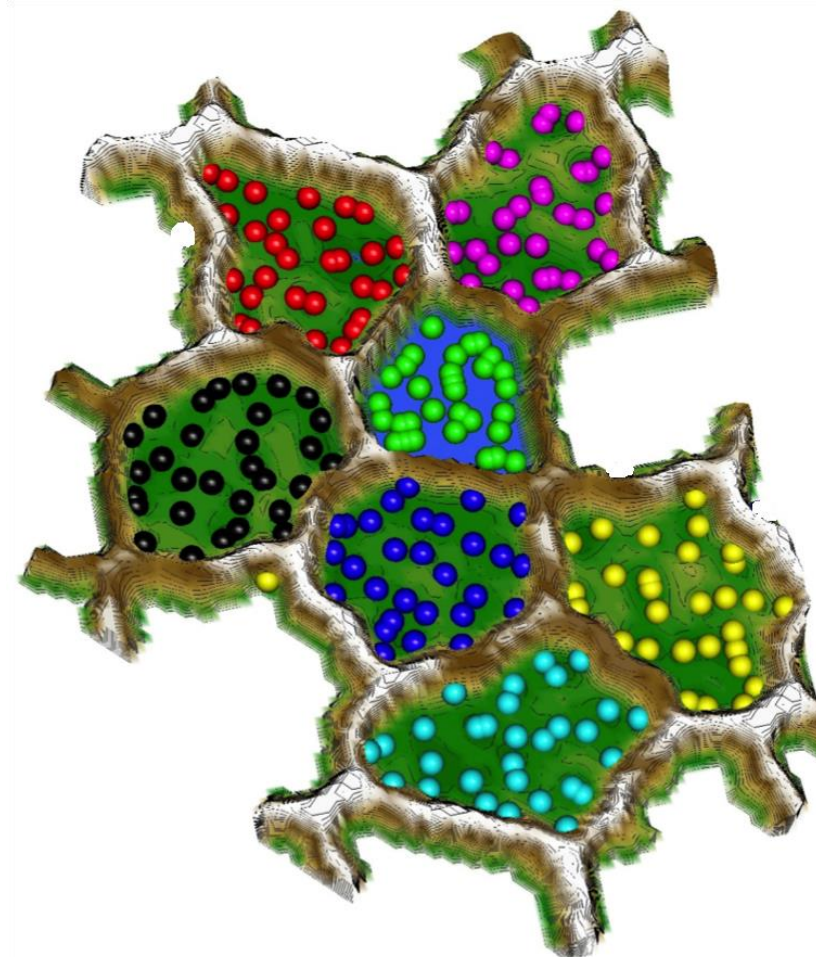
Databionic Swarm: Swarm-based Projection and Cluster Analysis

Dr. Michael Thrun

Based on: Thrun, M. C., & Ultsch, A.: Swarm Intelligence for Self-Organized Clustering, Journal of Artificial Intelligence, accepted, 2020.



Application:
Dimensionality Reduction for
Data Visualization
and Clustering

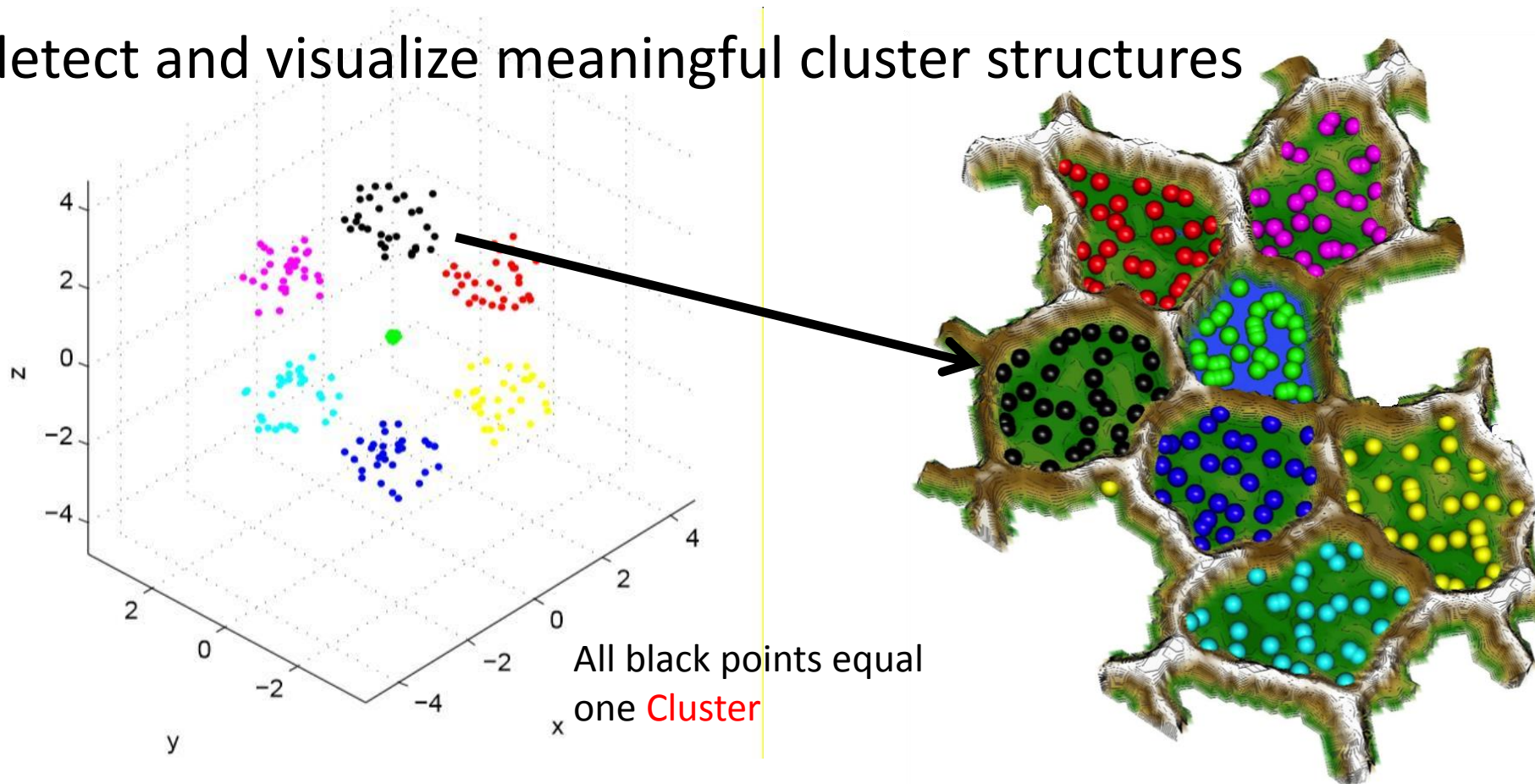


Projection-based Clustering

Problem: separate data into similar groups -> **Clustering**

Solution: Visualize data in two dimensions as landscape -> **Projection**

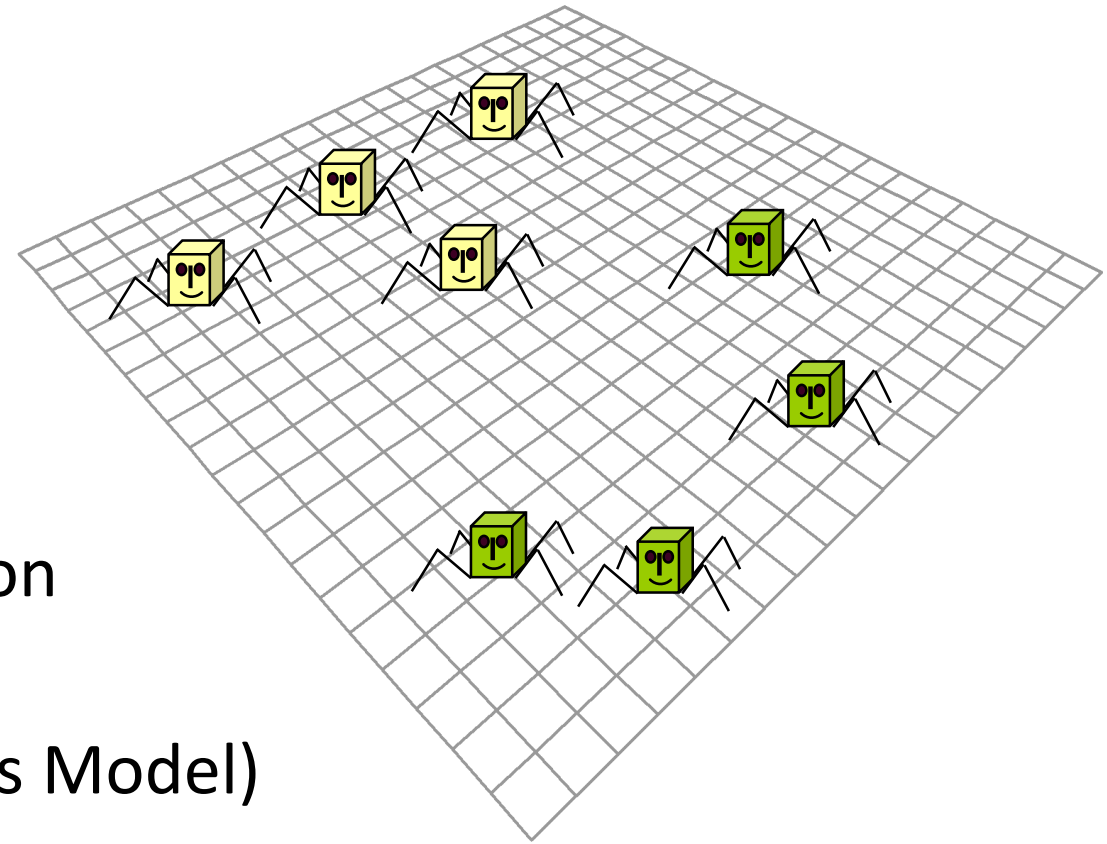
Goal: detect and visualize meaningful cluster structures



Prior: DataBots [Ultsch 1999]

Each DataBot

- Represents exactly one data record
- Sits on a grid position (initially random)
- Evaluates his environment
- Has motion strategies for each grid direction
- Moves towards “improvement” (Schelling’s Model)



Common Concepts of Swarms

1. Swarm intelligence

[Beni 1989], [Cao et al., 1997], [Grosan et al., 2006]

“the **collective behavior** of **many** simple entities called agents”

1. Self-Organization

- Swarms: [Bonabeau/Dorigo et al., 1999]
- SOM: [Ultsch 1992]

“spontaneous pattern formation by a system itself, without responsibility of any determinate inside agent”

2. Bionics

[Deneubourg 1991, Reynolds, 1987]

“the application of biological methods and systems found in nature”

My Ideas [Thrun, 2018]

- Combine these three concepts
- Make use of Emergence
 - the arising of novel and coherent properties during the process of self-organization
- Make use of Game theory by realizing the similarity to swarm intelligence

Introduction into Game Theory I

- Decision-making situations are modeled in which several parties influence each other
- Derive rational decision-making behavior in social conflict situations by using mathematical models
- Used in economics
 - (Operations Research)
 - political science
 - Psychology
 - Biology
 - Poker
 -
- 8 Nobel Memorial Prices

Introduction into Game Theory II

Important representatives:

- John von Neumann and Oskar Morgenstern (“Theory of Games and Economic Behavior”)
- John Forbes Nash Jr
- Thomas Schelling
- Daniel Kahneman
- RAND Corporation (offer research and analysis to the United States Armed Forces -> space race, nuclear deterrence in Cold war)

Topics

⇒ A solution concept always exists for egoistic agents in a non cooperative game

- *What does this mean?*
 - > watering garden during drought
- Definitions
- What would be an equilibrium for a game?
 - > Prisoner's dilemma

Watering Garden During Drought

- Should the garden be watered?
 - self-interest (**payoff**: living garden) $\leftrightarrow$ restraint (conserving water for community)
- No matter what the other persons do, a person is always better off watering his garden:
 - Unnecessary for one person to exercise restraint if the most other persons are restraining as well
 - Even if the rest of the community doesn't exercise restraint, it is futile for just one person to do so since one person does not have that big of an impact on the whole water supply
- Cooperative game: e.g. fine for watering the garden

Game and Players in Game Theory

- A game is defined by $i=1...n$ **players**, where each player makes a **choice**
- The choice of each player determines the outcome for each player
 - In general the outcome will be different for different players

$\Pi_i = \{\pi_1^i, \dots, \pi_\alpha^i, \dots, \pi_k^i\}$ represent the set of the k choices made by player i

- where π_α^i indicates the i^{th} player's α^{th} choice

Strategies in Game theory

- the choices Π_i $i = 1, \dots, n$ are defined as mixed strategies $s_j(i) \in S_i$ for each player i

$$s_j(i) = \sum_{\alpha=1}^{k(i)} c_{\alpha}(i) \pi_{\alpha}(i)$$

$$\sum_{\alpha=1}^{k(i)} c_{\alpha}(i) = 1 \quad \text{and all} \quad c_{\alpha}(i) \geq 0$$

Payoff in Game Theory

⇒ In a game, the **payoff** p of each **player** i depends not only on his own choice, but also on the choice of all other players

$$p = (p_1, \dots, p_n): \Pi_1 \times \dots \times \Pi_n \rightarrow \mathbb{R}^n$$

⇒ models situations in which multiple players interact or affect each other's outcomes

Equilibrium in Game Theory

- For non-cooperative games [J. Nash, 1951] proved the existence of at least one equilibrium point

Let $t_j(i) \in S_i$ be the mixed strategy that maximizes the payoff for the player i ; then, the Nash equilibrium is defined as

$$p_i(s(1), \dots, s(i-1), t_j(i), s(i+1), \dots, s(n)) = \max_{t_j(i) \in S_i} p_i(s(1), \dots, s(n))$$

- if and only if this equation holds for every i [Nash, 1951].

Types Nash Equilibria

$$p_i \left(s(1), \dots, s(i-1), t_j(i), s(i+1), \dots, s(n) \right) = \max_{t_j(i) \in S_i} p_i \left(s(1), \dots, s(n) \right) \quad (2)$$

⇒ The mixed strategy $t_j(i) \in S_i$ is the equilibrium point if no deviation in strategy by any single person results in a greater profit for that person

- Nash equilibrium is called **weak** if multiple mixed strategies $t_j(i) \in S_i$ for the same person exist in equation (2) that result in the same maximal payoff p_i
- Whereas in a **strong** Nash equilibrium, even a coalition of players cannot further increase their payoffs by simultaneously changing their strategies $t_j(i) \in S_i, i = 1 \dots m \leq n$, in (2)

Prisoner's dilemma - requirements

- Two persons A and B imprisoned for a crime.
- Each prisoner is in solitary confinement with no means of communicating with the other
- The prosecutors lack sufficient evidence to convict the pair on the principal charge
- Decision of each person does not affect future reward/punishment

Prisoner's dilemma - choices

1. If A and B each betray the other, each of them serves 2 years in prison
2. If A betrays B but B remains silent, A will be set free and B will serve 3 years in prison (and vice versa)
3. If A and B both remain silent, both of them will only serve 1 year in prison (on the lesser charge)

<i>A/B</i>	<i>Betray</i>	<i>Silent</i>
<i>Betray</i>	$[-2, -2]$	$[0, -3]$
<i>Silent</i>	$[-3, 0]$	$[-1, -1]$

- Details see: ../Subversion\WGR\Hausarbeiten\16Spieltheorie

Prisoner's dilemma payoff matrix

<i>A/B</i>	<i>Betray</i>	<i>Silent</i>
<i>Betray</i>	$[-2, -2]$	$[0, -3]$
<i>Silent</i>	$[-3, 0]$	$[-1, -1]$

- Rational solution: $[-1, -1]$
- BUT “mixed strategy $t_j(i) \in S_i$ is the equilibrium point if no deviation in strategy by **any** single person results in a greater profit for that person”

Prisoner's dilemma payoff matrix

<i>A/B</i>	<i>Betray</i>	<i>Silent</i>
<i>Betray</i>	$[-2, -2]$	$[0, -3]$
<i>Silent</i>	$[-3, 0]$	$[-1, -1]$

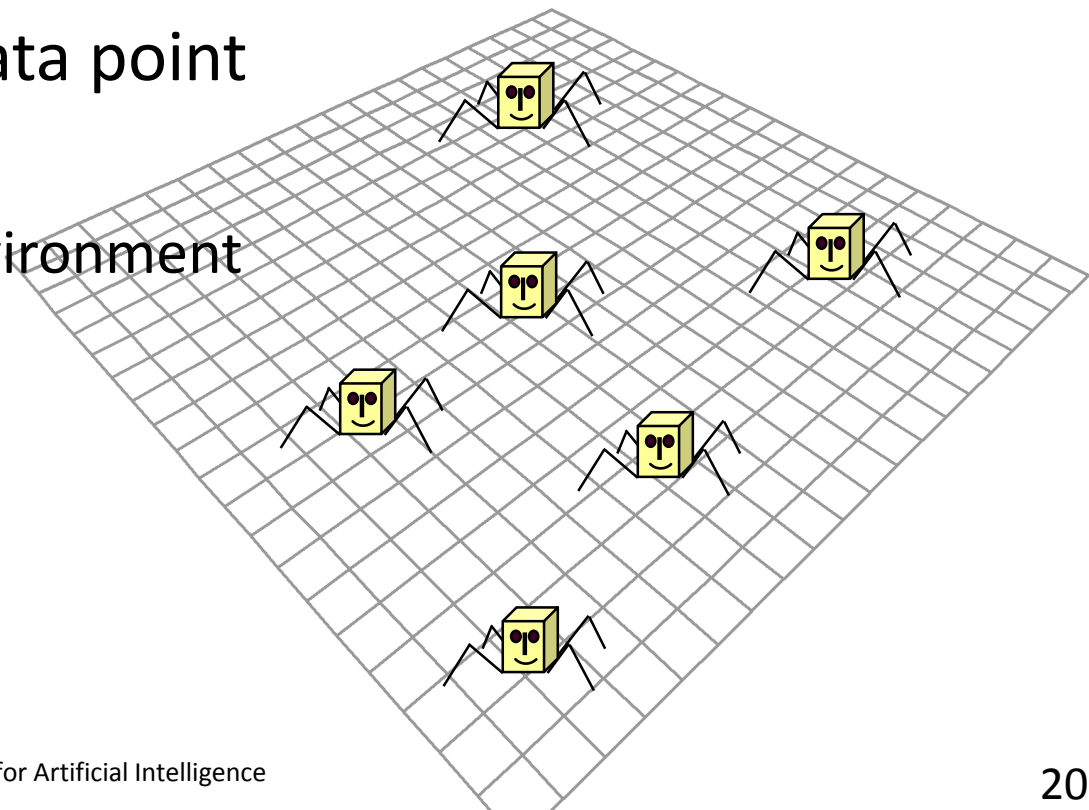
=> Nash equilibrium: $[-2, -2]$, because otherwise player b betrays player a (can maximize his/her payoff more)

If is implied that

- Prisoners will have no opportunity to reward or punish their partner other than the prison sentences they get
- Their decision will not affect their reputation in the future.

Databionic Swarm Principle

- A projection based clustering method based on emergent, self-organizing, egoistic, artificial life forms
- The collective is called DataBionic Swarm (DBS)
 - Every entity is defined as a DataBot
- Every DataBot is represented by exact one data point
- DataBots communicate through stigmergy:
 1. Live on a two-dimensional toroid grid as an environment
 2. Are able to smell a position and emit scent



Concepts for Databionic swarm (DBS)

- Swarm intelligence
- Self-Organization and Emergence
- Bionics
 1. Communication: Scent
 2. Living in and moving on a flat surface
 3. “Ants” wear their colony’s dead -> *will wear data points*
 4. Preference: Smelling the surroundings of ones place
 5. “Ant” moves to a free positions, if it prefers the scent of the new position
- Application of Game Theory
 - Non-cooperative game -> Nash equilibrium

How to describe the preference for having neighbors of their own type?

- Requirements for *preference*
 - Topological neighborhood and distance on grid
 - Correct position determination
- Requirements for projection method
 - Focussing scheme with proper reduction of neighborhood -> Nash equilibrium

physics:

- Given problem -> Search first for symmetry
- Solution results in parameter reduction
 - Rmin, Rmax, epochs, ...
 - Neighborfunction
 - Grid form and grid distance

DBS has three interchangeable modules

I. Polar swarm (Pswarm)

- symmetry considerations
 - Polar Coordinates on a Hexadiagonal toroidal grid
 - Adept definition of distance in two dimensions
- Application of game theory: Scent -> payoff
- Annealing scheme based on game theory
 - Searches for Nash equilibrium in each epoch
- Emergence: No Objective function or parameters (Except R^n distance)

II. Visualization: topographic map with hypsometric tints

- Simplified ESOM-Algorithmus for Interpolation, parameter- free
- Automated U^* matrix computation
- 3D landscape generation

III. Semi-automated Clustering

Scent of DataBots == payoff in game theory

Hermann 2009 (SOP):

Let $D(l,j)$ be the distance of $x_l, x_j \in I$, let $d(l,j)$ be the distance in the Output space O , let $F_R: R \rightarrow [0,1]$ be an arbitrary but continuous and monotone decreasing function, then the scent $\lambda(b_j, R): \mathbb{R}_0^+ \times O \rightarrow \mathbb{R}_0^+$ is defined with

$$\text{Preference/Scent: } \lambda(b_j, R) = \frac{\sum_{l \in I} F_R(d(j,l)) * D(j,l)}{\sum_{l \in I} F_R(d(j,l))}$$

DataBionicSwarm

$$\text{Payoff: } \lambda(b_j, R, S_0) = \begin{cases} S_0 - \frac{\sum_{l \in I} H_R(r(j,l)) * D(j,l)}{\sum_{l \in I} H_R(r(j,l))}, & \text{if } \sum_{l \in W} H_R(r(j,l)) > 0 \\ S_0, & \text{else} \end{cases}$$

Scent of DataBots == payoff in game theory

Hermann 2009 (SOP):

Let $D(l,j)$ be the distance of $x_l, x_j \in I$, let $d(l,j)$ be the distance in the Output space O , let $F_R: R \rightarrow [0,1]$ be an arbitrary but continuous and monotone decreasing function, then the scent $\lambda(b_j, R): \mathbb{R}_0^+ \times O \rightarrow \mathbb{R}_0^+$ is defined with

**Maximizes
structure preservation**

$$\text{Preference/Scent: } \lambda(b_j, R) = \frac{\sum_{l \in I} F_R(d(j,l)) * D(j,l)}{\sum_{l \in I} F_R(d(j,l))}$$

DataBionicSwarm

$$\text{Payoff: } \lambda(b_j, R, S_0) = \begin{cases} S_0 - \frac{\sum_{l \in I} H_R(r(j,l)) * D(j,l)}{\sum_{l \in I} H_R(r(j,l))}, & \text{if } \sum_{l \in W} H_R(r(j,l)) > 0 \\ S_0, & \text{else} \end{cases}$$

Scent of DataBots == payoff in game theory

Hermann 2009 (SOP):

Let $D(l,j)$ be the distance of $x_l, x_j \in I$, let $d(l,j)$ be the distance in the Output space O , let $F_R: R \rightarrow [0,1]$ be an arbitrary but continuous and monotone decreasing function, then the scent $\lambda(b_j, R): \mathbb{R}_0^+ \times O \rightarrow \mathbb{R}_0^+$ is defined with

$$\text{Preference/Scent: } \lambda(b_j, R) = \frac{\sum_{l \in I} F_R(d(j,l)) * D(j,l)}{\sum_{l \in I} F_R(d(j,l))}$$

DataBionicSwarm

$$\text{Payoff: } \lambda(b_j, R, S_0) = \begin{cases} S_0 - \frac{\sum_{l \in I} H_R(r(j,l)) * D(j,l)}{\sum_{l \in I} H_R(r(j,l))}, & \text{if } \sum_{l \in W} H_R(r(j,l)) > 0 \\ S_0, & \text{else} \end{cases}$$

Algorithm

```

function Positions O=pswarm(matrix D(l,j))
  for all  $z_i \in I$ : assign an initial random polar position  $i_\phi(r) \in O$  on the grid
    to generate DataBots  $b_i \in B$ 
  for  $R = \{R_{max} = Lines/2, \dots, R_{min}\}$  do
    calculate chance  $c(R)$ 
    Repeat for each iteration
       $c = \text{sample}(c(R), B)$ 
       $m_{\tilde{i}}(c) = \text{uniform}(1, R_{max}), \tilde{i} = 1, \dots, \alpha$ 
       $l(c) = \text{argmax}_j (\lambda(b_j, R))$  with  $j = \{i, m_{\tilde{i}}(c)\} \in O$ 
       $l(!c) = i$ 
       $S = \sum_l \lambda_l(b_l, R)$ 
    until  $\frac{\partial S(e, \lambda(R))}{\partial e} = 0$ 
  return O in cartesian coordinates
end function pswarm

```

Algorithm

function Positions O=pswarm(matrix D(l,j))

for all $z_i \in I$: assign an initial random polar position $i_\phi(r) \in O$ on the grid

to generate DataBots $b_i \in B$

for $R=\{Rmax=Lines/2,...,Rmin\}$ do

calculate chance $c(R)$

Repeat for each iteration

$c = \text{sample}(c(R), B)$

$m_{\tilde{i}}(c) = \text{uniform}(1, Rmax), \tilde{i}=1,...,\alpha$

$l(c) = \text{argmax}_j(\lambda(b_j, R))$ with $j = \{i, m_{\tilde{i}}(c)\} \in O$

$l(!c) = i$

$S = \sum_l \lambda_l(b_l, R)$

until $\frac{\partial S(e, \lambda(R))}{\partial e} = 0$

return O in cartesian coordinates

end function pswarm

*Grid size
automatically
chosen,
contrary to
ESOM*

*Iterations
defined by
annealing
Rmax defined
by grid size*

Details: Define Gridsize

Let the size of the grid defined by Lines L and Columns C , N number of DataBots, α be the number of possible jumping positions, $\beta \in (0.5, 1]$, $p_{01}(D)$ robust minum, $p_{99}(D)$ robust maximum

$$\text{I.} \cdot \frac{\sqrt{C^2 + L^2}}{1} \geq \frac{p_{01}(D)}{p_{99}(D)} =: A \quad \text{shortest/longest distances assignable to grid units}$$

$$\text{II.} \cdot L * C \geq \alpha * N \quad \text{enough free positions for a DataBot to jump}$$

$$\text{III.} \cdot \frac{L}{C} = \frac{\beta}{1} \quad \text{rectangular grid}$$

Resulting in a bi-quadratic equation of

$$C^4 - A^2 * C^2 + \alpha^2 * N^2 = 0$$

$$z_{1/2} = A^2 \pm \frac{1}{2} \sqrt{A^4 - \frac{\alpha^2}{4} N^2}$$

$$\Rightarrow C = \begin{cases} \frac{1}{\sqrt{2}} \sqrt{A^2 + \sqrt{A^4 - \frac{\alpha^2}{4} N^2}}, & \text{If } A^4 > \frac{\alpha^2}{4} N^2 \\ \text{approximation, else} \end{cases}$$



Algorithm

function Positions $O = \text{pswarm}(\text{matrix } D(l,j))$

for all $z_i \in I$: assign an initial random polar position $i_\phi(r) \in O$ on the grid

to generate DataBots $b_i \in B$

for $R = \{R_{\max} = \text{Lines}/2, \dots, R_{\min}\}$ do

calculate chance $c(R)$

Repeat for each iteration

$c = \text{sample}(c(R), B)$

$m_{\tilde{i}}(c) = \text{uniform}(1, R_{\max}), \tilde{i} = 1, \dots, \alpha$

$l(c) = \text{argmax}_j (\lambda(b_j, R))$ with $j = \{i, m_{\tilde{i}}(c)\} \in O$

$l(!c) = i$

$S = \sum_l \lambda_l(b_l, R)$

until $\frac{\partial S(e, \lambda(R))}{\partial e} = 0$

return O in cartesian coordinates

end function pswarm

*Parameter
free annealing
process*

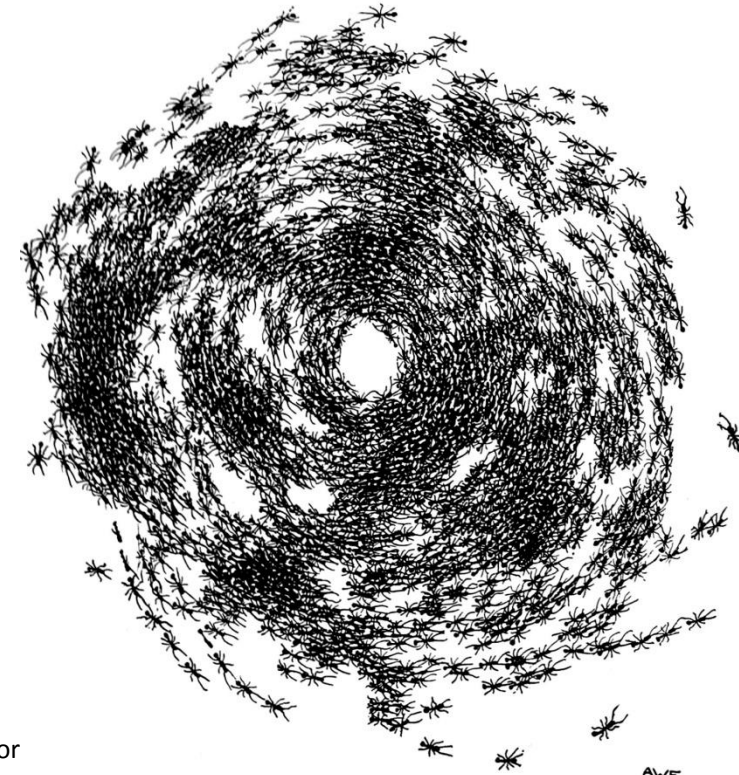
Details: *annealing process*

- Precise neighborhood function (polar (r,phi) coordinates)

Symmetry considerations (physics): $d(j, l) \rightarrow$ radius between l and j and

$$F_R = H_R = \begin{cases} 1 - \frac{r(j, l)^2}{\pi R^2}, & \text{if } \frac{r(j, l)^2}{\pi R^2} < 1 \\ 0, & \text{else} \end{cases}$$

- $R_{\max} = \text{Lines}/2$ else self interaction
- $R_{\min} \gg 1$ else „ant mill“ [Schneirla 1971]
 - Fixed
 - Determined by grid size,
 - Number of DataBots



Algorithm

function Positions O=pswarm(matrix D(l,j))

for all $z_i \in I$: assign an initial random polar position $i_\phi(r) \in O$ on the grid

to generate DataBots $b_i \in B$

for $R=\{Rmax=Lines/2,...,Rmin\}$

do calculate chance $c(R)$

Repeat for each iteration

$c = \text{sample}(c(R), B)$

$m_{\tilde{i}}(c) = \text{uniform}(1, Rmax), \tilde{i}=1,...,\alpha$

$l(c) = \text{argmax}_j(\lambda(b_j, R))$ with $j = \{i, m_{\tilde{i}}(c)\} \in O$

$l(!c) = i$

$S = \sum_l \lambda_l(b_l, R)$

until $\frac{\partial S(e, \lambda(R))}{\partial e} = 0$

return O in cartesian coordinates

end function pswarm

TradeOff

Batch vs

Online

(one Databot

moves versus

all Databot

move)

Searching for

Nash

equilibrium

Algorithm

function Positions O=pswarm(matrix D(l,j))

for all $z_i \in I$: assign an initial random polar position $i_\phi(r) \in O$ on the grid

to generate DataBots $b_i \in B$

for $R=\{Rmax=Lines/2,...,Rmin\}$ do

calculate chance $c(R)$

Repeat for each iteration

$c = \text{sample}(c(R), B)$

$m_{\tilde{i}}(c) = \text{uniform}(1, Rmax), \tilde{i}=1,...,\alpha$

$l(c) = \text{argmax}_j(\lambda(b_j, R))$ with $j = \{i, m_{\tilde{i}}(c)\} \in O$

$l(!c) = i$

$S = \sum_l \lambda_l(b_l, R)$

until $\frac{\partial S(e, \lambda(R))}{\partial e} = 0$

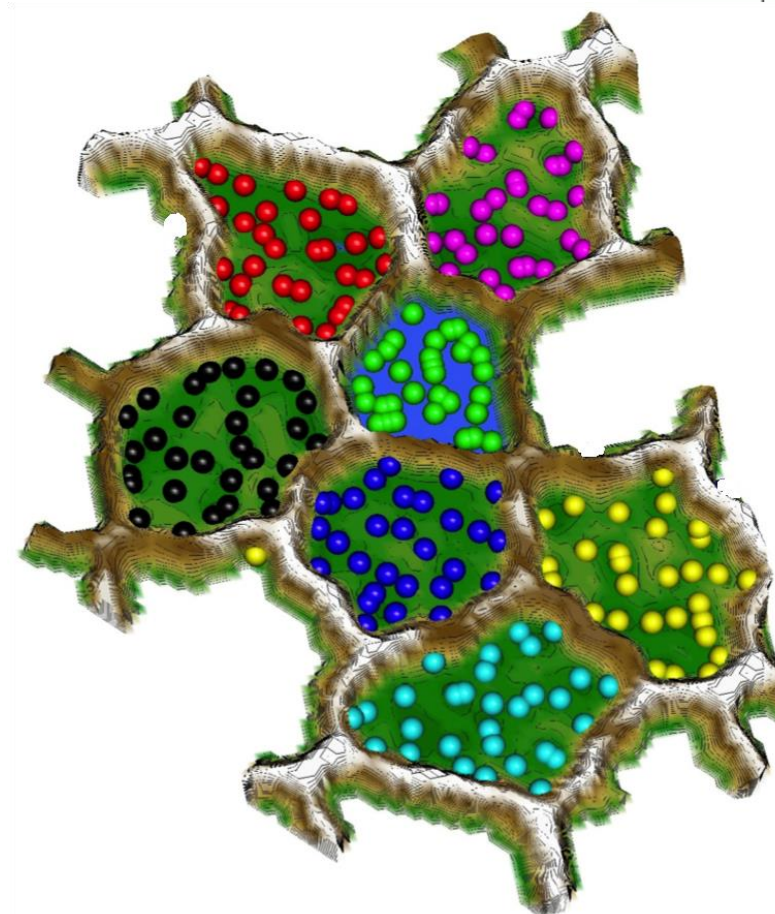
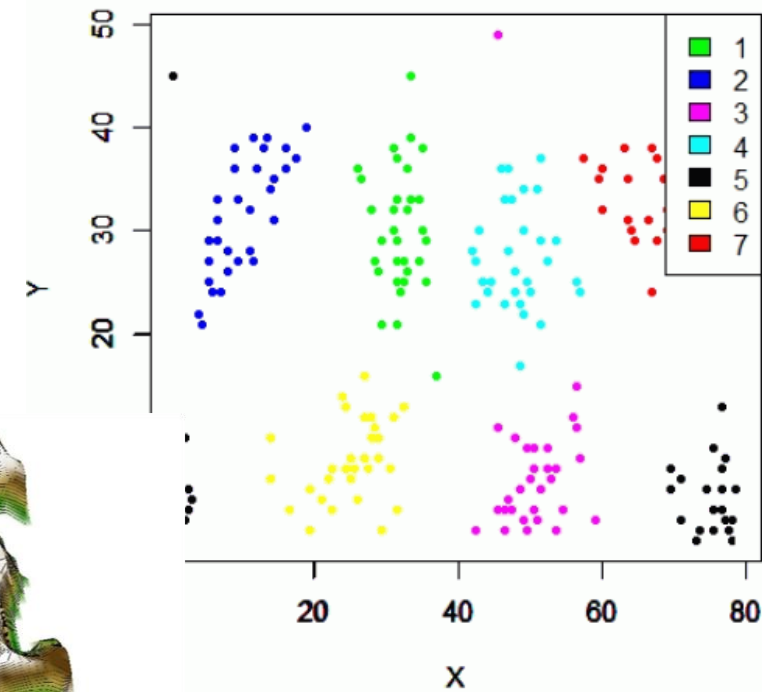
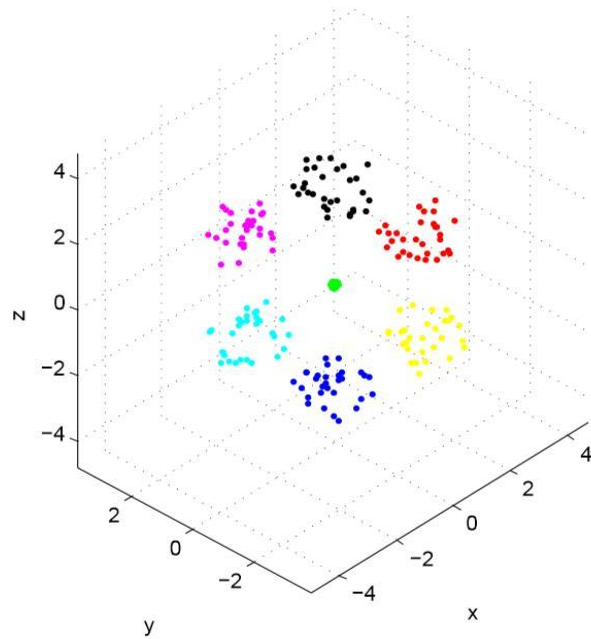
return O in cartesian coordinates

end function pswarm

*Choice is set of
mixed
strategies -
>select
positions to
jump to*

*Egoistic, non-
cooperative game
Jump only if new
position 'as higher
payoff*

Result Pswarm:



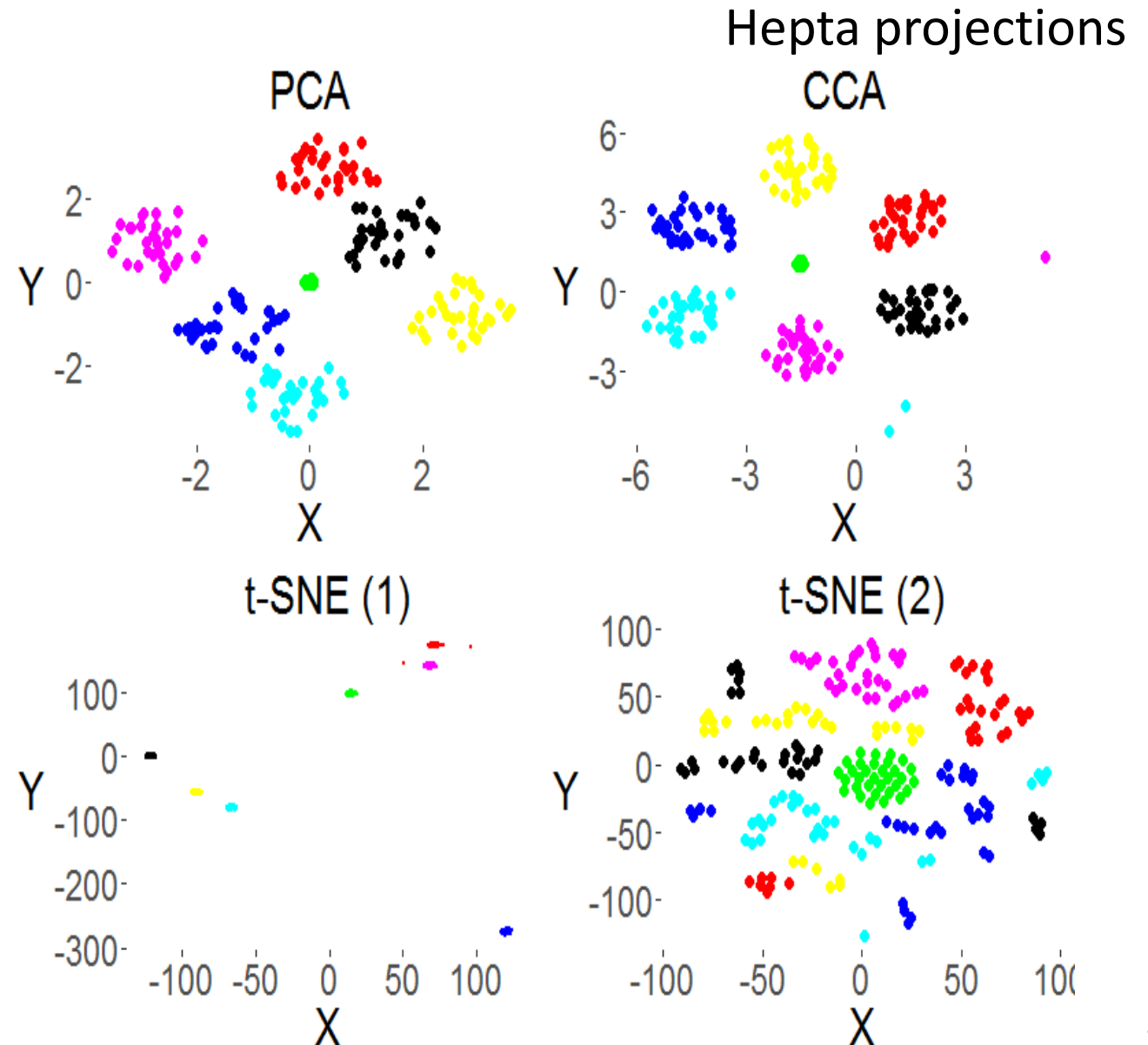
Why do we want this?

DBS Advantage: Parameter-Free!

- Choice of projection method
- Choice of parameters for the projection

Structure preservation:
projection

$\mathbf{R}^N \rightarrow \mathbf{R}^m$ with $N > m$ cannot
preserve all distances!



DataBionic Swarm II

Content:

I. Pswarm

II. Visualization: topographic map with hypsometric tints

1. Transformation from hexagonal grid to BestMatches
2. Simplified ESOM-Algorithmus for Interpolation
3. Automated estimation for P-Matrix Radius [Thrun et al. 2016]
4. U*matrix computation [Ultsch et al. 2016]
5. 3D landscape generation [Thrun et al. 2016]
 1. Normalization of U*-Matrix + automatic rectangular Island generation

III. Semi-automated Clustering

- Uses abstract U-distances [Lötsch/Ultsch, 2014]
- Hierarchical Clustering [Ultsch et al. 2016]

Visualization technique: topographic map with hypsometric tints

U*-Matrix Visualization of Pswarm:

- considers distance and density based structures of data

Reason:

- take the structures seen on the U-matrix into account, s.

Lötsch, J., Ultsch, A.: Exploiting the structures of the U-matrix, 2014

-> Clustering:

- Formalization of the structures through a U-matrix such that
 - distance calculations between best-matching units
 - the height structures of a U-matrix (U-cell distance).
- This enables
 - the assessment of structure preservation and the implementation of clustering algorithms.

Simplified ESOM-Algorithmus for Interpolation

Two Problems:

1.)

ESOM algorithm is about a Best Matching Units (BMUs) search

But Pswarm results in given BMUs

2.)

In the normal ESOM algorithm following parameters have to be set

- Neighborfunction
- maxEppochs
- AnfangsLernrate
- EndLernrate
- CoolingMethod
- AnfangsRadius
- EndRadius
- Lines and Columns

Detail1: ProjectedPoints to BestMatches

Let the Grid begin on the x-axis at 1 and end with a maximal number defined as Columns, similar for Lines, then

$$\frac{Lines - 1}{Columns - 1} \approx \frac{|\max(y) - \min(y)|}{|\max(x) - \min(x)|} = \frac{dy}{dx} = \Delta$$

- describes, that the grid size should equal the size of the coordinate system as accurate as possible

The grid size should be around 4096:

$$Lines * Columns \geq 4096$$

- The resulting equation to be solved is

$$Lines^2 + Lines(1 + \Delta) - 4096\Delta \geq 0$$

- Resulting in

$$Lines \geq -\frac{1 + \Delta}{2} + \sqrt{\left(\frac{1 + \Delta}{2}\right)^2 + 4096\Delta}$$

sESOM: Special Case of ESOM

- Idea: Let the bestmatches be fix [Mörchen/Uttsch 2005]
- My idea: with good neighborhood function the algorithmus can be simplified

The neighborhood function **h** is defined by

$$h = \begin{cases} H_R: R \rightarrow [0,1]: \\ 1 - \frac{d(j,l)^2}{\pi R^2}, \text{ iff } \frac{d(j,l)^2}{\pi R^2} < 1 \\ 0, \text{ else} \end{cases}$$

- maxEpochs=10
- Learning rate not required (==1)
- **annealingScheme**='linear' (and only for Radius)
 - CurrentRadius = max(AnfangsRadius-CurrentEpoch,1)
 - AnfangsRadius, EndRadius well and autimatically defined

Simplified ESOM (sESOM)

Initialisierung

Setze An BMU Positionen der Karte die Gewichtsvektoren = Datensätze

Für jede Epoche

Führe ESOM-Algorithmus mit oberen Parametern ohne Suche nach bestmatches
aus

Ersetze an BMU Positionen der Karte die Gewichtsvektoren= Datensätze

U-Matrix: Standard über Moore-Nachbarschaft

Details: P-Matrix Radius Abschätzung

- Schätze minimalen, korrekten Radius und maximal möglichen Radius über ABCanalyse[Ultsch/Lötsch 2014] der Distanzen ab.
- u.U. Korrigiere Interaktiv an `iPmatrix()` nach
- $P = \text{InterClusterdistanz} / \text{InnerClusterdistanz}$
- $\text{InterClusterdistanz} = \min(\text{Distanz}(\text{GruppeA}))$
- $\text{InnerClusterdistanz} = \max(\text{Distanz}(\text{GruppeC}))$

U*matrix

- 3D landscape defined by U*matrix [Ultsch et al., 2016]
 - Combines Umatrix and Pmatrix
- Umatrix: folding of high dimensional space [Ultsch/Siemon, 1990]
 - (-) In literature cited as grey-scaled 2D vizualization (e.g. [Kadim Tasdemir/Merényi, 2012])
 - > Precise colored definition required
- Pmatrix: high dimensional density estimation technique [Ultsch, 2003]
 - (-) Estimation for hypersphere of radius is trying
 - > use ABCanalsys [Ultsch/Lötsch, 2015]
- => U*matrix represents distance and density based structures!

U*matrix -> 3D landscape

- use colors proposed in [Ultsch, 2003]

=> hypsometric tints: surface colors which depict ranges of elevation

- Calculate contour lines
 - Normalization of U*heights, define height intervalls, ...
- Combine specific color scale with contour lines
- Create island of toroid map (rectangular)

=> topographic map with hypsometric tints

(DBS) DataBionic Swarm III

I. Pswarm

II. Visualization: topographic map with hypsometric tints

III. Semi-automated Clustering

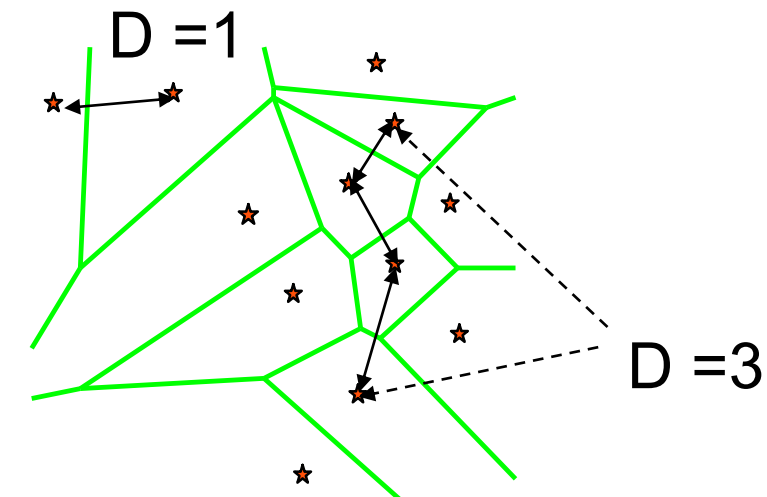
- Uses abstract U-distances [Lötsch/Ultsch, 2014]
- My idea: Calculate Shortest Paths between these distances
-> new distance matrix $\hat{D}$
- Apply Hierarchical Clustering [Ultsch et al. 2016] to $\hat{D}$
- My Idea: Only single linkage (SL) or ward relevant:
 - See Compact versus connected structures.
-> Cluster analysis requires the choice of compact or connected as a parameter
- Second parameter: number of clusters
 - Estimate from dendrogram oder visualization

Abstract U-distances

- Delaunay Graph: Graph of Voronoi cells
 - A region corresponds to each BestMatch consisting of all points closer to that BestMatch than to any other
- Delaunay Path: Number of edges between two BestMatches
- Dijkstra Shortest Paths of Delaunay graph weighted with highdimensional Distances

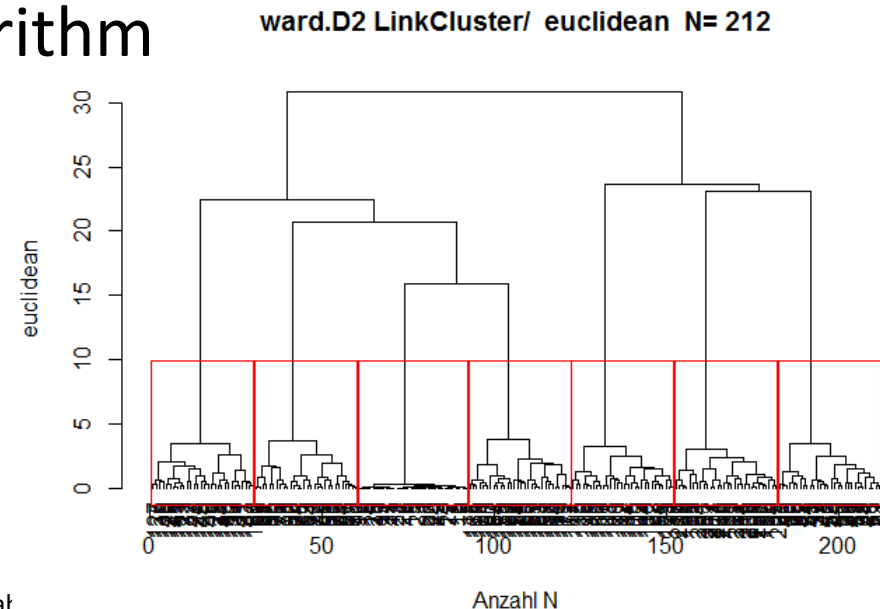
Reason:

- U-matrix is approximation of
Abstract Umatrix [Lötsch/Ultsch 2014]
which is based on Voronoi cells
- Height of Voronoi-Borders=Euclidean
High-dimensional distance of data



Hierarchical Clustering

- Clustering distance= shortest paths
- Aggloremative:
 - beginning: every point is one cluster
 - Calculate „similarity“ between clusters
 - Unite the two most „similar“ clusters to new cluster
 - Do this until only one cluster exists
- Similarity of clusters: depends on algorithm
- Evaluation of number of clusters:
 - Dendrogram -> „ultrametric“



Verification of DBS

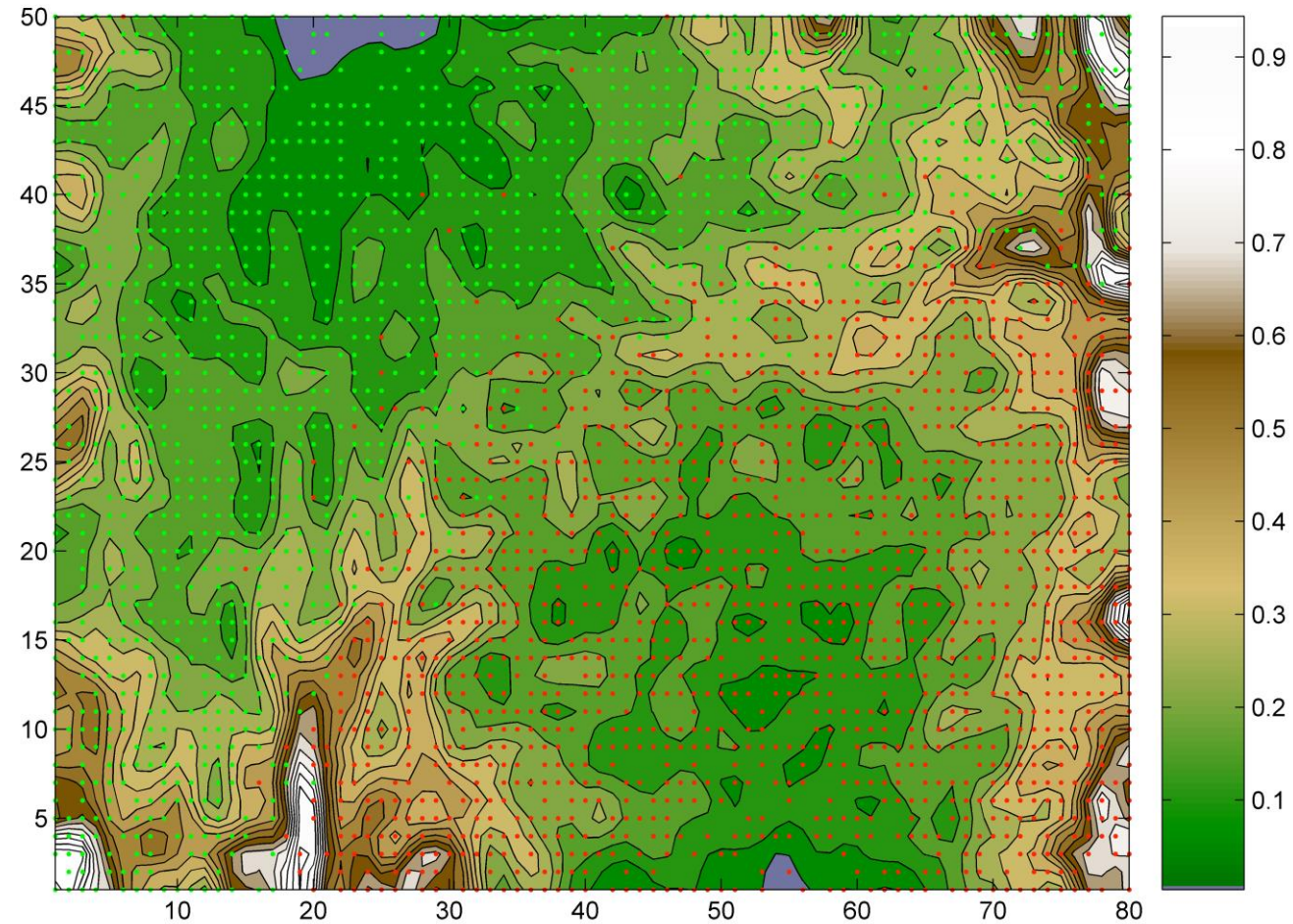
- No general error for projections exists (Thesis)
- Statistical: Only through accuracy of given Classification possible
 - Natural datasets: may have different clusters depending on goal
- =>Artificial data sets: Fundamental Clustering Problem Suite (FCPS)
- Visual: 3D Landscape of known datasets
- Application based: Searching for new Knowledge

Visual Benchmarking on FCPS

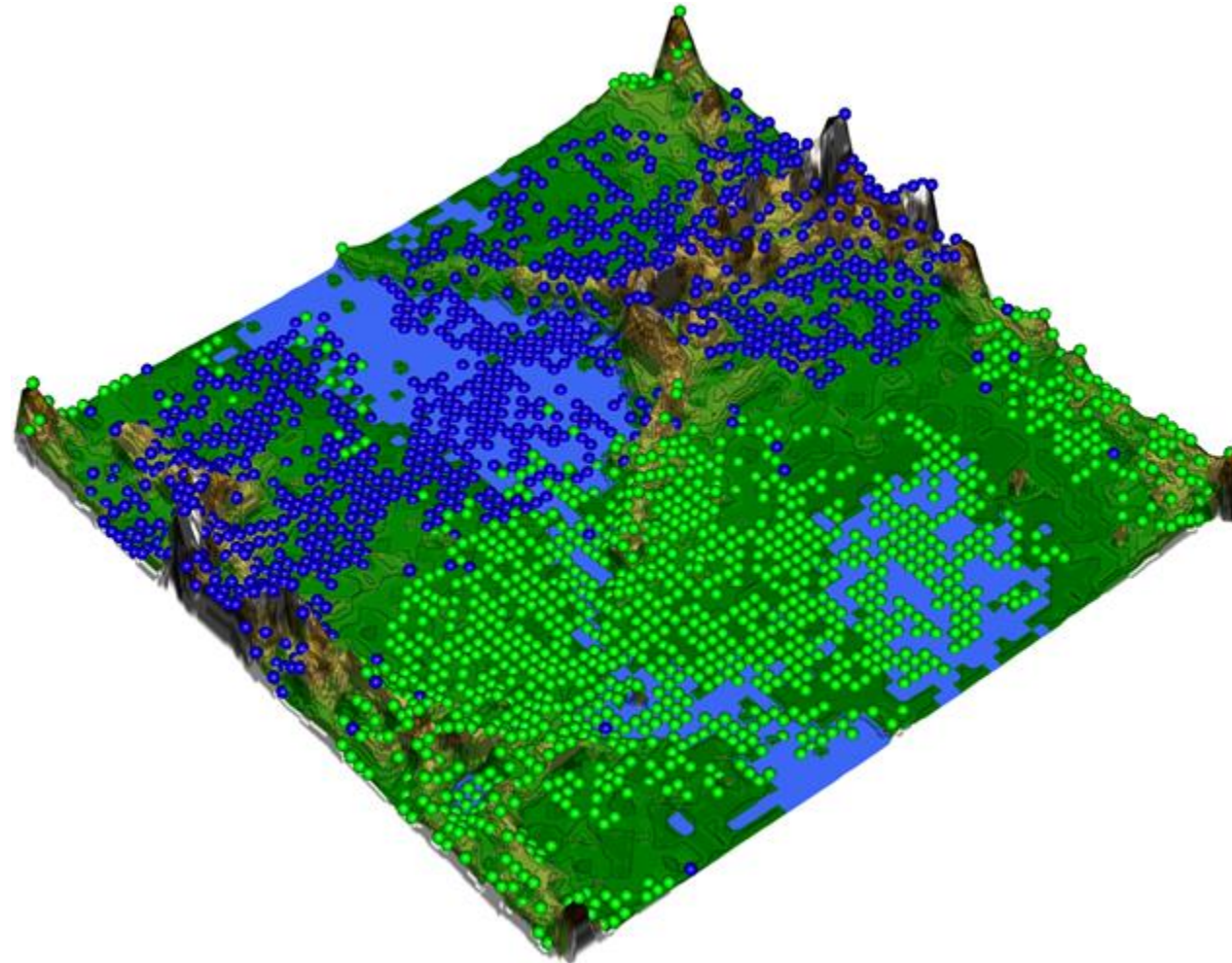
Using 3d landscape: Topographic map with hypsometric tints

Data/Projection	ESOM	NerV	SOP	DBS
Atom (k=2)	Trial and parameter depended	OK	3 Clusters	OK
Chainlink (k=2)	OK	OK	(OK)	OK
EngyTime (k=2)	Clusters invisible	PCA init	Intermixed Clusters	OK
Lsun (k=3)	(OK)	PCA init	Ok?	OK
Golfball (k=1)	OK	Clusters visible	OK	OK
WingNut	Too many substructures	PCA init	Ok?	OK
Other FCPS data sets	OK	OK	OK	OK

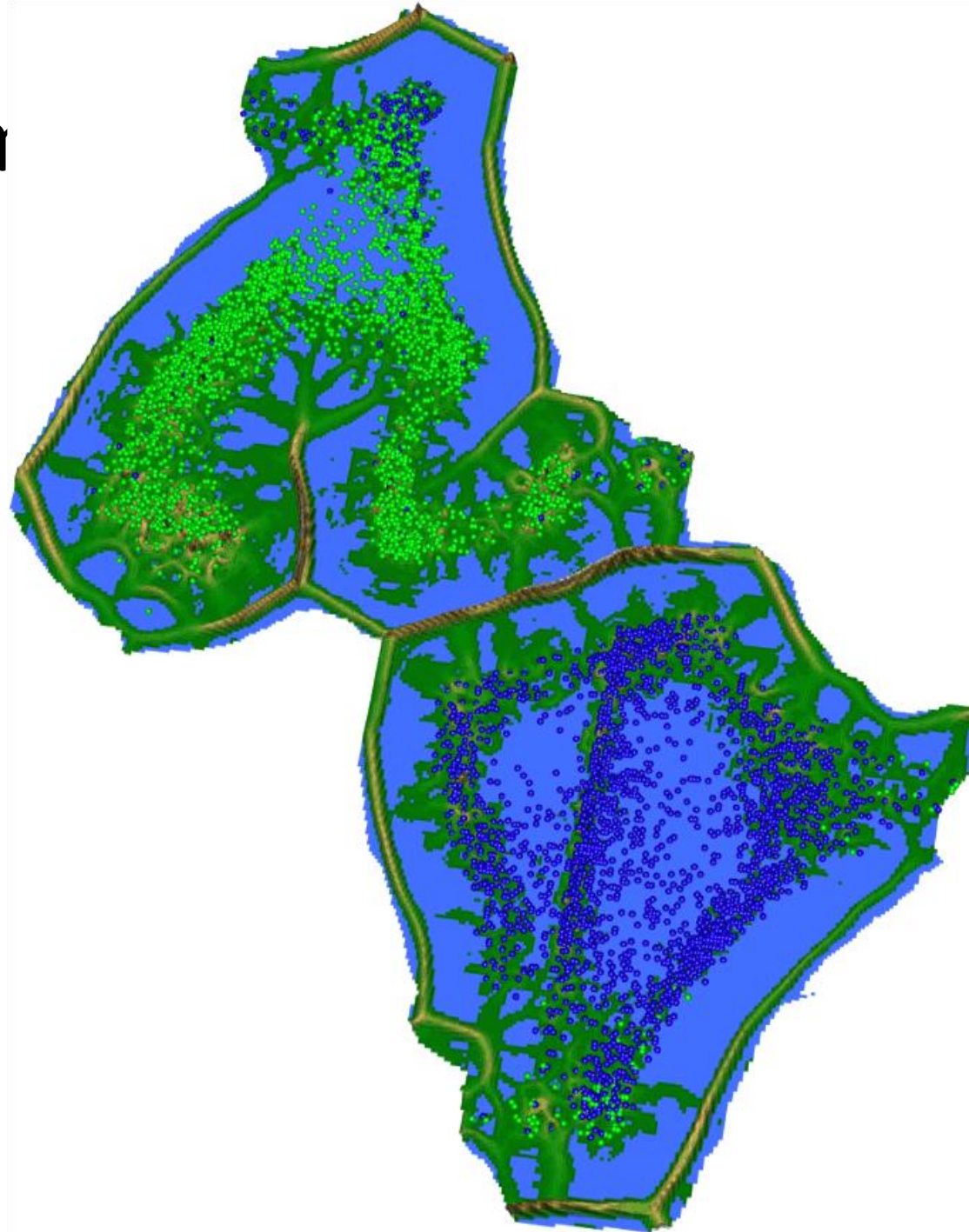
Example: EngyTime ESOM

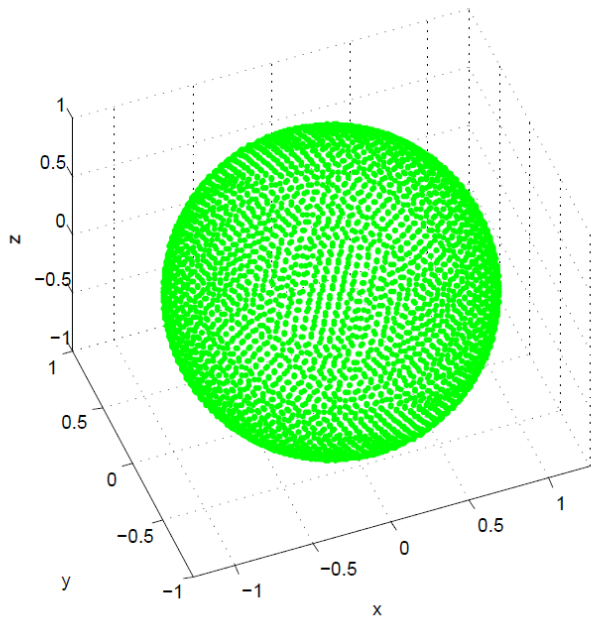


Example: EngyTime SOP



DBS EngyTin





State-of-The-Art: Ready-Made Method: e.g. t-SNE
Data Visualization by Dimensionality Reduction

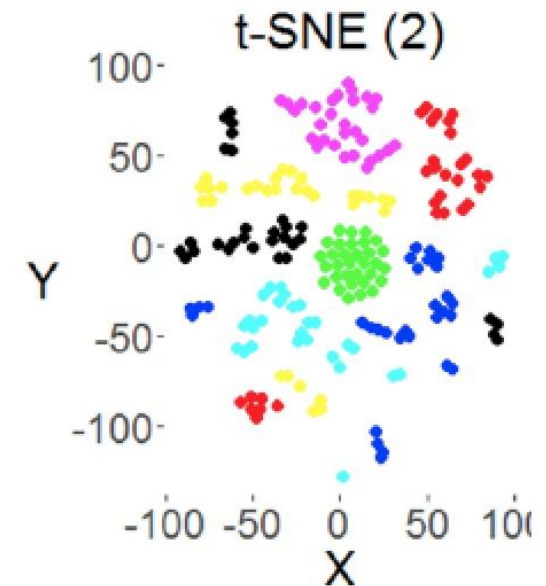
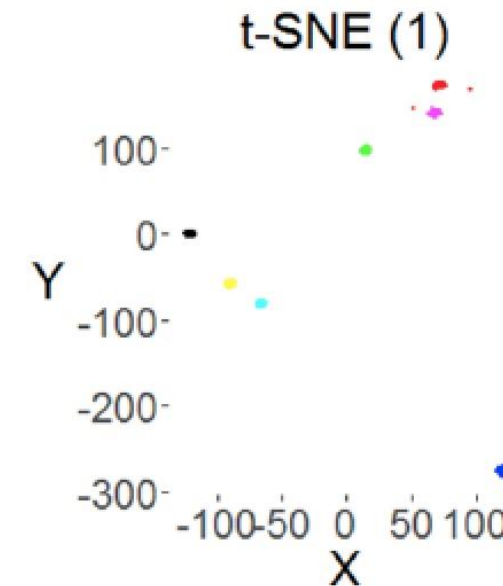
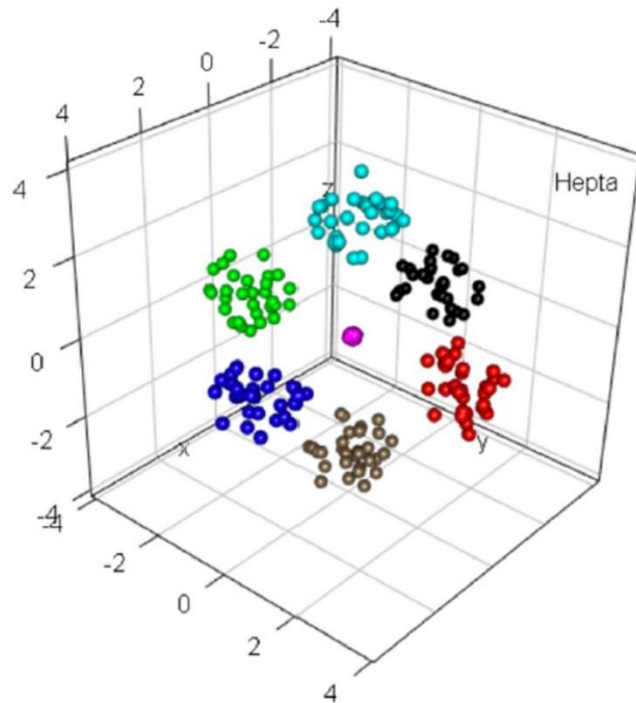
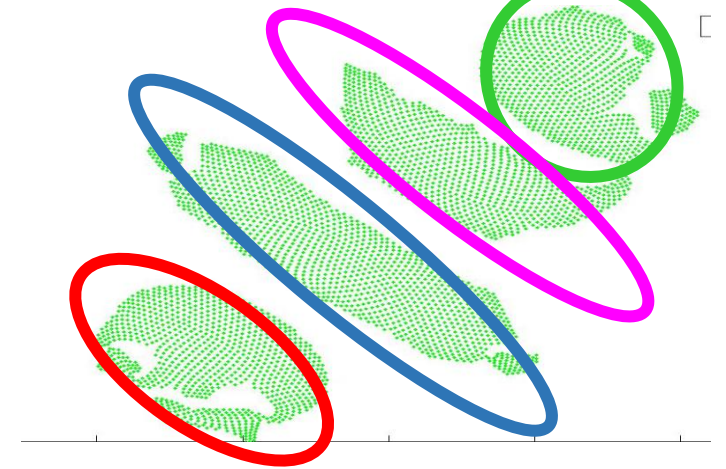
[Ultsch/Lötsch, 2019, IJMS]

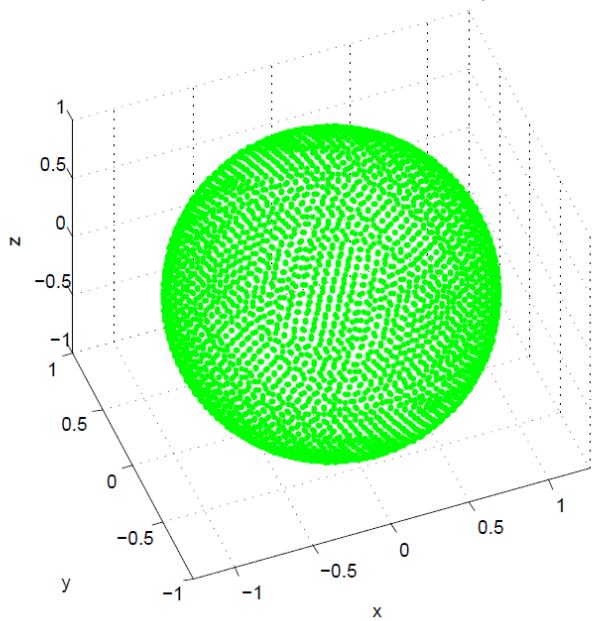
Dimensionality Reduction often results in biases at subgroup detection
=> Incorrect diagnoses

[Thrun, 2018,

ISBN

9783658205393]

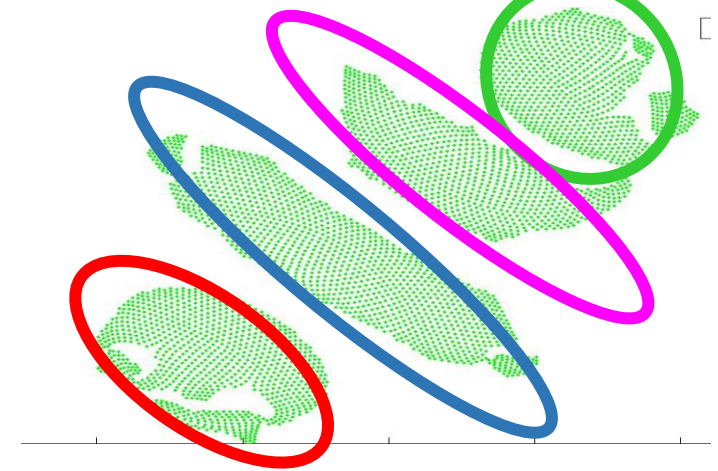




State-of-The-Art: Ready-Made Method: e.g. t-SNE
Data Visualization by Dimensionality Reduction

[Ultsch/Lötsch, 2019, in press]

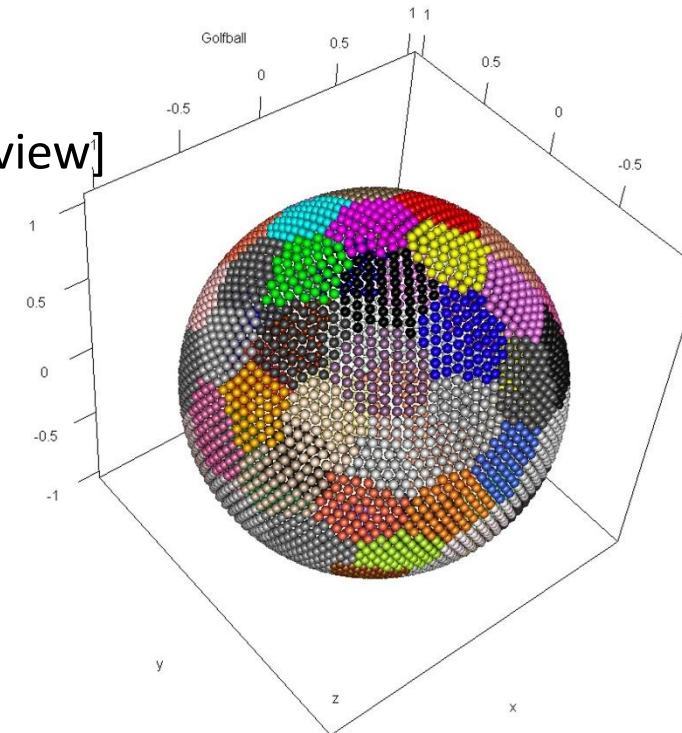
Chapter 4: Behavior-Based Systems



Optimization does not work if it does not fit the sought cluster structures!!!

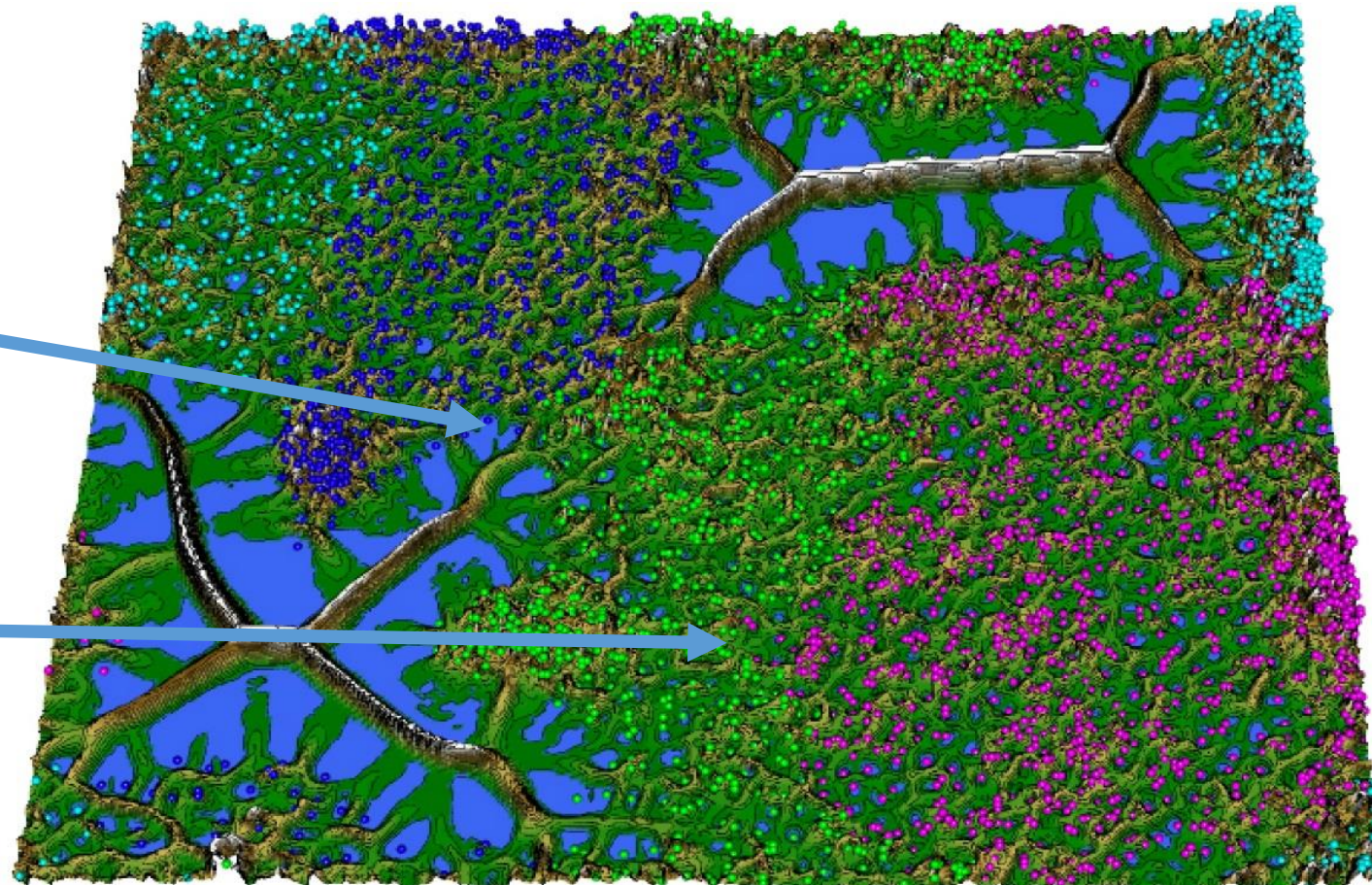
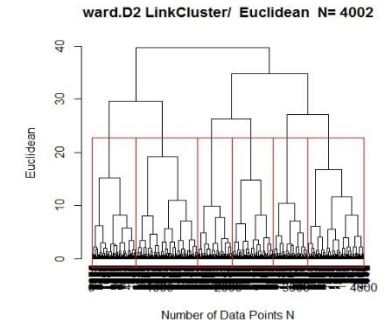
State-of-The-Art: Ready-Made Methods:
SOM Clustering (Coloring of Points)
[Thrun, 2019, under review]

Clustering of non-structured data results irrelevant diagnoses because within the cluster there are no more similar diagnosis than outside the cluster



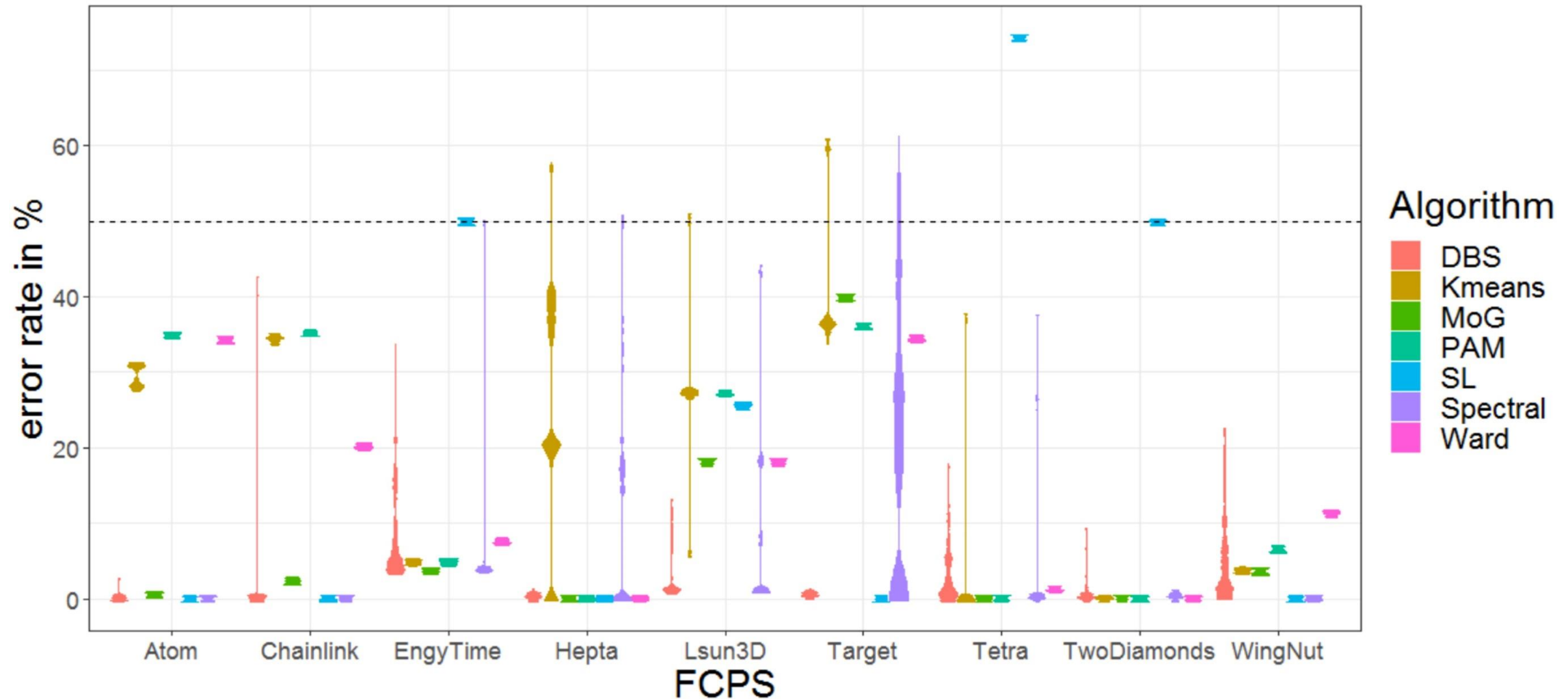
DBS Advantage: Shows the Absence of Clusters

- Topographic map of the structures of the Golfball dataset where the DBS projection and (compact) clustering are included
- Visualization does not indicate a cluster structure because no valleys are visible and the mountain ranges are not closed
- DBS clustering generates clusters that are not separated by mountains
- No island can be extracted from the toroidal visualization.



Accuracy of DBS Clustering via MDplot

MD Plot: Error Probability of Common Clustering Algorithms



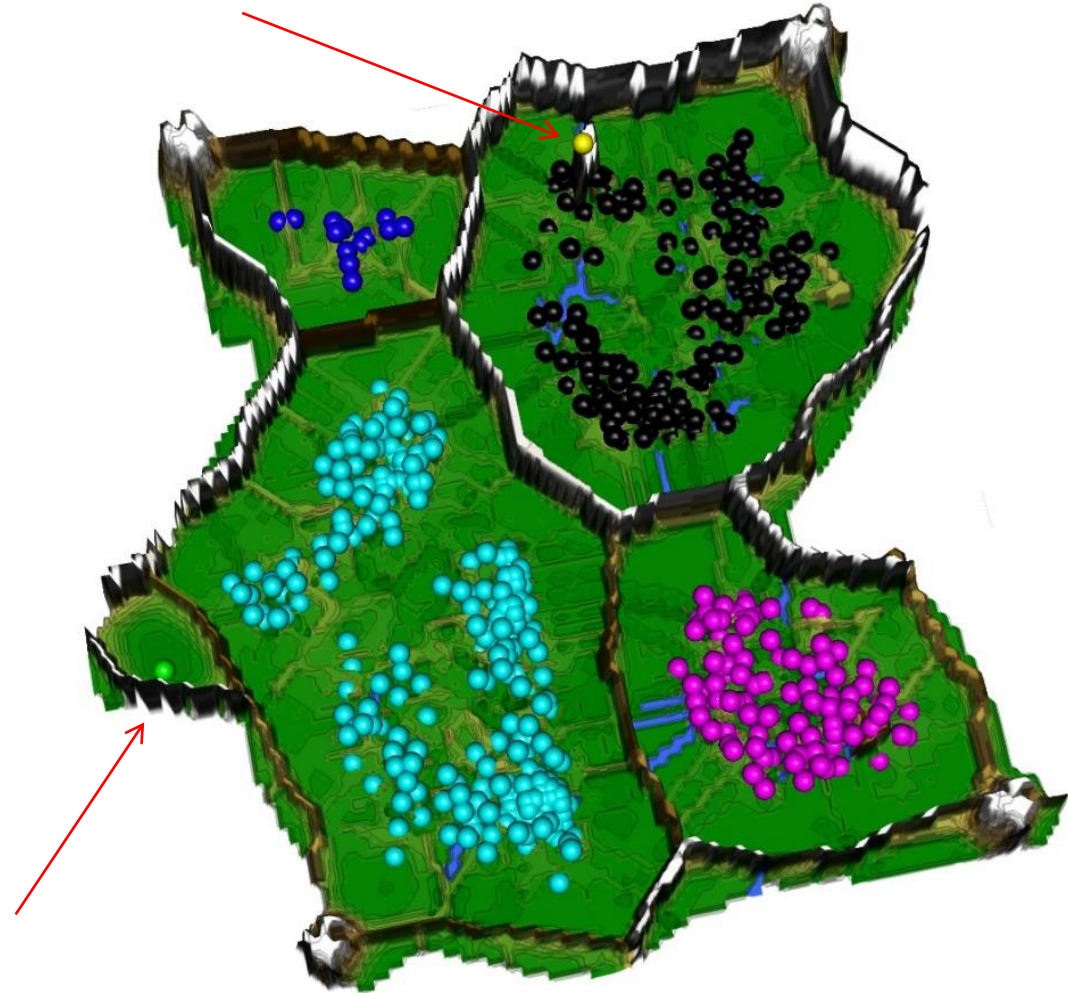
Application I: High-dimensional Data

Leucemia

- 554 Patients with 7447 active Genes
- Four diagnosis
- 6 Klassen reproduzieren
- Die Vorklassifizierung mit einer Accuracy von 99.6%

Two Outliers:

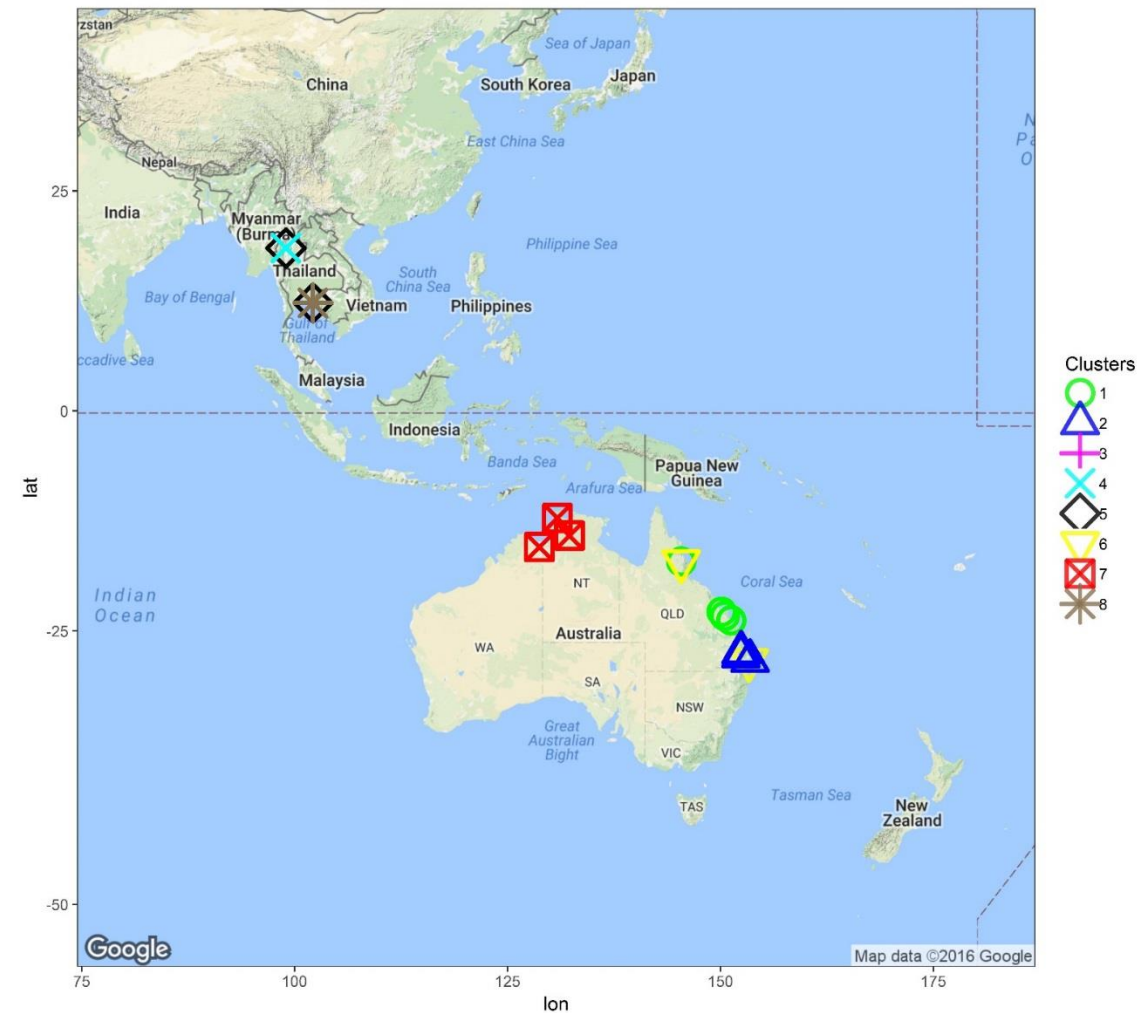
- -> problem in diagnosis
- -> see our future publication
- [Brendel, et al.,2017]



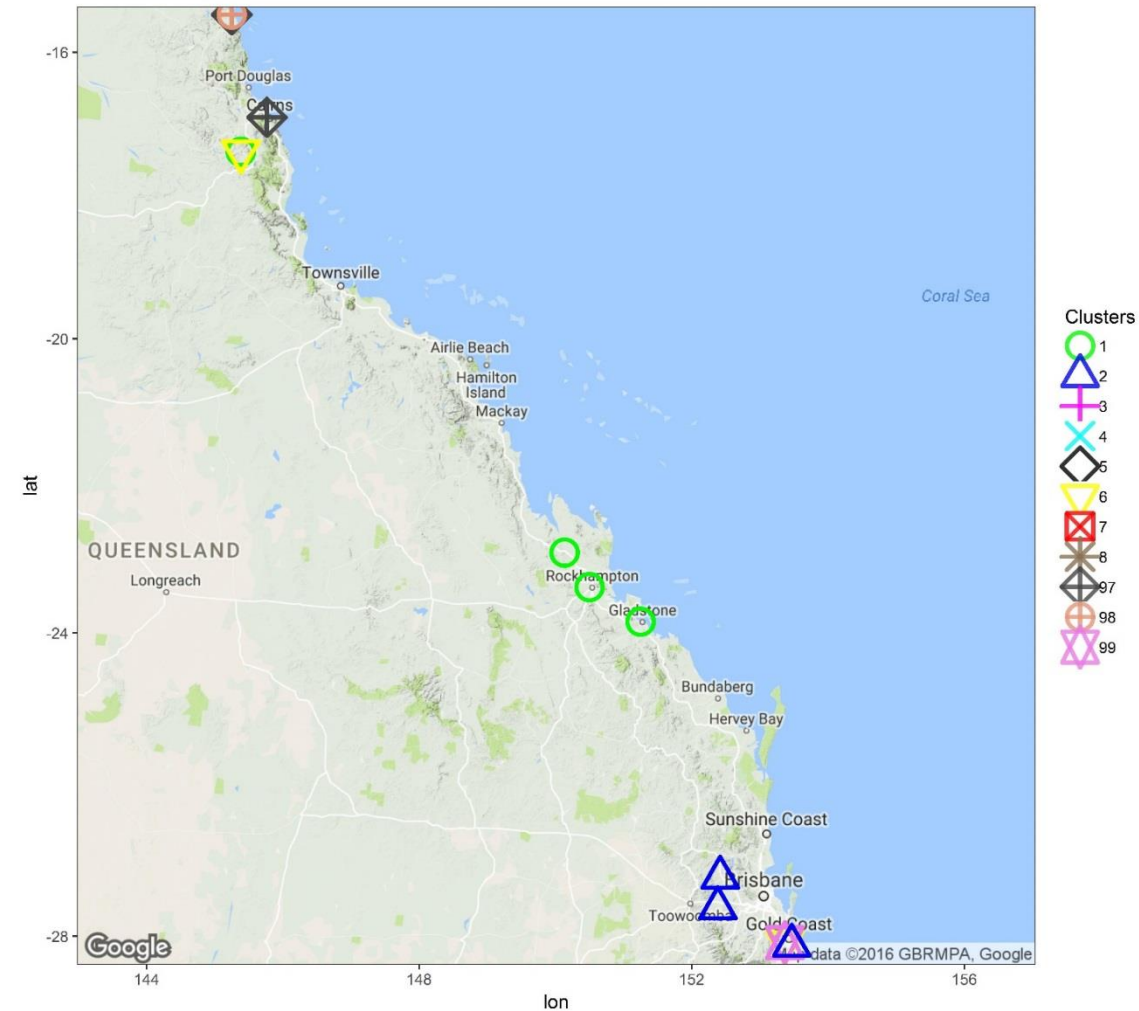
Application 2: Genetic Data of Bees



Main Clusters depends on Location of Bees



Outliers are in Queensland



Published Applications

- Reproduction of high-dimensional structures for pain genes [Thrun, 2018, p. 145] in Fig. A.2
- Clustering Genes [Thrun, M. C., Lippmann, C., & Ultsch, A]
- For noxious cold stimuli [Weyer-Menkhoff et al., 2018]
- Using high-dimensional structures of clusters new knowledge was generated in the case of the world gross domestic product [Thrun, 2019]
- For of next-generation sequencing regarding the *TRPA1/TRPV1* NGS genotypes [Kringel et al., 2018] depicted in Fig. A7
- Hydrology [Thrun et al., 2018], [Thrun et al., 2019, under review]
- Stock Picking in the German stock market [Thrun, 2019], [Thrun/Ultsch, 2020, under review]

- What do you think the disadvantages are?

Disadvantages

- Computation time due to complexity
 - 4000 Datarows require 4000 Databots => 24hours of computation¹
- Variance of results because DBS is a stochastic algorithm (see slide 56)
- Island generation is not automatic and toroidal visualization is not that easy to interpret
 - Same disadvantage as ESOM
- Density structures are being reproduced in Pswarm but only visible in P-Matrix or U*-Matrix (which requires) P-matrix
 - P-Matrix requires one parameter to be set which is challenging!
 - Same disadvantage as ESOM
- Clusters are generated but not automatically explained with DBS (the same is true for any other clustering algorithm)

Summary

- DataBionicSwarm (DBS) is an interactive Visualization and Clustering approach consting of 3 modules
 - Parameter free
 - Independ of number of variables
 - Self-organizing and emergent
- DBS uses projection and high-dimensional data for clustering
- Every module is interchangeable with another state of the art application of each respective field, e.g.
 - Projection: NeRV instead of Pswarm
 - Clustering: Spectral clustering instead of DBSclustering

Future Work

- Find a strong Nash equilibrium
- Abstract U^* -distances instead of abstract U -distances
 - How to integrate Pmatrix information?
- Add new data incrementally to given projection
 - Efficiency: Project more than 4000 points in less than 24h
- Better Knowledge Extraction from Cluster structures [Florian Lerch, Diss]
- apply, connections to solid-state physics
 - E.g Use Bravais lattice or apply a Fourier transformation to the reciprocal lattice (do calculations in the reciprocal space)